MEASURING FRONTIER CODE AGENTS AT DIGITAL-TWIN SCENE CONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Simulation-based robot learning is gated by a manual step that repeats at every deployment site: constructing a physical digital twin of the scene. This paper measures how far unmodified, off-the-shelf code agents can automate that step, a setting referred to as Frontier Agent as Scene Constructor (FASC). Frontier code agents such as Claude Code and Codex, each used as shipped, rebuild eight BEHAVIOR-1K rooms from one photograph, an asset catalog and the simulator. Each reconstruction is scored against the room's known 3D layout on ACDC's sim-to-sim protocol, with memorisation and file-access contamination controlled. The strongest agent on the rooms, Codex with GPT-6 Astra, retrieves the exact asset model for 66% of the scored objects and places objects within 14 cm of ground truth across repeated runs. Claude Code with Opus 5, released a little over one month earlier, retrieves the exact asset model for 52% of the scored objects, against 6% for the same agent with Opus 4.6, released under six months earlier, and places objects within 39 cm against 153 cm, a 4× reduction in placement error over that predecessor. A gap to ACDC's curated pipeline, at 6 cm, remains. Next, the two strongest agents on the rooms both extend to outdoor farm scenes, where no calibrated camera or 3D ground truth exists, using a procedural plant model and common asset libraries. Validated against their photographs in DINOv2 embedding space, 24 of the 46 GPT-6 Astra twins and 22 of the 46 Opus 5 twins rank their own photograph first among the 46 reference photographs in cosine similarity. Together with prior results on trajectory generation, these findings indicate that a generic code-agent harness covers a growing share of embodied tasks, and that its capabilities improve with each frontier-model generation while requiring no embodiment-specific harness engineering.

1 INTRODUCTION

Simulation-based policy learning and evaluation for robot manipulation still begins with a manual step: constructing a digital twin, or at least a visually matched approximation, of the deployment scene by human modeling, scanning, or annotation of the site (Torne et al., 2024; Li et al., 2024b). The cost repeats with every deployment site, since each site requires its own in-distribution scene, and policy performance degrades under variation in object and table textures and camera place-

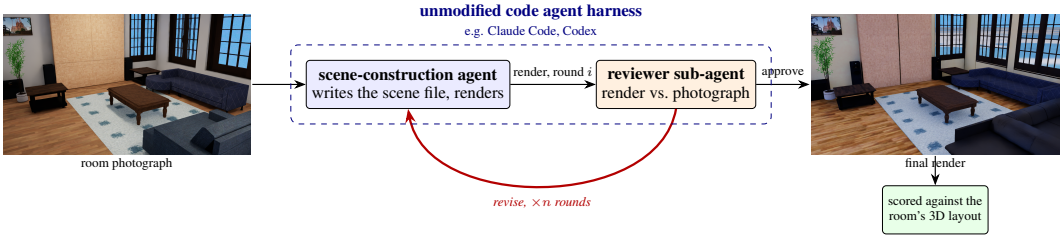

Figure 1: Room reconstruction from one photograph. The agent writes and renders a scene file, and a reviewer sub-agent it spawns compares each render to the photograph until it approves; the final scene is re-rendered once and scored against the room's 3D layout (Section 3).

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.

ment (Li et al., 2024b), and under lighting and clutter changes (Pumacay et al., 2024). Prior work automates twin construction with purpose-built systems of three kinds. Reconstruction pipelines recover a scene from a photograph through a fixed perception chain of object detection, depth estimation, asset retrieval or generation, and pose estimation, or reconstruct individual objects into simulation-ready assets (Dai et al., 2024; Zhao et al., 2025; Gu et al., 2026; Le et al., 2025; Cao et al., 2026). Language-model composers assemble scenes from asset libraries, prompted by text rather than a photograph (Wang et al., 2024b; Yang et al., 2024). Agentic builders wrap a language model in a custom scaffold of roles, skills, and verification loops that steers it through scene construction. None of these systems measures what an unmodified general-purpose code agent achieves on the same task, with no custom harness built around it. The hypothesis of this paper is that purpose-built harnesses arose to compensate for capabilities that earlier models lacked, and that the gap they compensate for narrows with every frontier model and their agent harness release.

This paper measures the gap that remains between an unmodified code agent and purpose-built pipelines at scene reconstruction, a setting referred to here as Frontier Agent as Scene Constructor (FASC), by analogy with FAEA (Tsui et al., 2026). FASC names the setting rather than a method: the agent is used as shipped, and the only inputs are the photograph, the simulator, and one prompt. Seven frontier code agents, each used as shipped through Claude Code or OpenAI Codex, rebuild eight BEHAVIOR-1K rooms from one photograph each and are scored against the rooms' known 3D layout on ACDC's sim-to-sim protocol (Dai et al., 2024), with memorisation and file-access contamination controlled (Figure 1; Section 3). From the photograph the agent estimates the object layout (and, when no calibrated camera is supplied, the camera pose), then selects, places, and configures assets from a catalog. The two strongest of them, Codex with GPT-6 Astra and Claude Code with Opus 5, are then tested for generality on farm photographs, a domain with no calibrated camera and no 3D reference, where each agent places plants pre-grown with a procedural botanical model and each twin is validated against its photograph, and where the two perform comparably (Section 5). On the rooms Astra places objects within 14 cm of ground truth and retrieves the exact asset model for 66% of the scored objects. The gain from one model generation to the next is large within each provider as well. Claude Opus 5 places objects within 39 cm, a 4$\times$ reduction in placement error over Opus 4.6, released under six months earlier, whose reconstructions recover too few objects to be practical. Within Codex, GPT-5.6 Sol cuts GPT-5.5's placement error by a third and nearly triples its exact-asset recovery, and GPT-6 Astra, released eight weeks after GPT-5.6 Sol, cuts its placement error from 76 to 14 cm and doubles its exact-asset recovery, suggesting that reconstruction performance scales with frontier-model capability independent of harness engineering.

2 Related Work

Twins from images. Simulation-ready scenes have been constructed in several distinct ways. Most scenes and assets are still modeled manually (Szot et al., 2021; Jin et al., 2026b; Kolve et al., 2017; Khanna et al., 2024; Li et al., 2024a); language-guided systems compose them from asset libraries without a photograph (Yang et al., 2024; Wang et al., 2024b); and real-to-sim pipelines rebuild one specific real setup, by manually matching a simulated scene's textures and background to the real workspace (Li et al., 2024b), by scanning the environment and manually annotating the joints of its articulated objects, such as drawers and cabinets (Torne et al., 2024), or by fitting the physical parameters of deformable objects to video of a person manipulating them (Jiang et al., 2025; Zhang et al., 2026b). A growing line of work instead turns visual observation directly into simulation content: recovering geometry, physics, and camera pose from one image (Zhao et al., 2025), articulating assets (Chen et al., 2024; Cao et al., 2026; Le et al., 2025; He et al., 2025; Wu et al., 2026b), and composing scenes from images and panoramas (Zhang et al., 2026c; Lee et al., 2026b; Lu et al., 2026a; Wu et al., 2026a) or video (Wang et al., 2026c; Jain et al., 2026; Han et al., 2026; Dan et al., 2025; Xia et al., 2026); IGen (Gu et al., 2026) likewise lifts a single image into a point-cloud workspace but renders observations from point clouds rather than a physics-based reconstruction. ACDC composes cousin scenes from one image (Dai et al., 2024), and PRISM generates demonstrations from an image and an instruction (Ko et al., 2026). Both vary the object instances rather than reproduce the photographed site, and both retrieve assets rather than generate them. SimFoundry and GRS instead require a video scan or depth sensing (Ranawaka et al., 2026; Zook et al., 2025). Concurrent work reaches simulator-usable scenes from real observations through purpose-built pipelines. Lucida parses multi-view video into a scene graph, generates an asset for each instance and places it with a trained placement policy (Qin et al., 2026), and NeoWorld-Pro has a multimodal model write

scene programs from a single image and refines them with physics-in-the-loop validation (He et al., 2026). Each pipeline fixes its strategy—reconstruct, retrieve, or generate—at design time.

Agentic scene builders. A newer family wraps the model in role or skill harnesses (Pfaff et al., 2026; Yang et al., 2026; Chen et al., 2026). Several build from a description rather than a photograph (Ning et al., 2026; Wang et al., 2026a;b), and those that take an image score 3D geometry only against synthetic ground truth or pseudo-labelled scans (Kim et al., 2026; Liu et al., 2025), or report no 3D-ground-truth metric (Kang et al., 2026). The nearest, SceneConductor, orchestrates specialized agent roles across a three-stage pipeline from a single image to a Blender scene (Kim et al., 2026), and EmbodiedGen V2 connects off-the-shelf coding agents to a skill harness with policy training in its generated worlds (Wang et al., 2026b). Code agents in robot pipelines are wrapped the same way, calling authored primitives, writing task and reward code, or generating integration glue inside designer-built loops (Liang et al., 2023; Singh et al., 2023; Wang et al., 2024a; Hua et al., 2024; Ma et al., 2024; Wang et al., 2024b; Katara et al., 2024; Chen et al., 2025; Lee et al., 2026a; Jing et al., 2025; Liu et al., 2026; Fu et al., 2026; Berman et al., 2026; Xiao et al., 2026; Li et al., 2026; Jin et al., 2026a; Zhang et al., 2026a; Lu et al., 2026b; Elmaaroufi et al., 2026); unmodified agents have acted from privileged state or written algorithmic policies (Tsui et al., 2026; Xie et al., 2026), in neither case referenced to a real scene. This work instead gives one unmodified code agent the simulator and domain-specific generators as tools, so retrieval, generation, and reconstruction are choices the agent makes. It recovers the object layout from the photograph, and the camera pose when none is given, then selects, places, and configures assets. Each twin is validated against 3D references when available and against the photograph otherwise.

3 Reconstructing each B1K room from one photograph

This paper first tests whether one unmodified code agent, given a photograph, references the simulator library and composes a physical 3D twin of the photographed scene. BEHAVIOR-1K (B1K) rooms with known 3D ground truth are scored on ACDC's sim-to-sim protocol (Dai et al., 2024) (Figure 1); in this paper's setup the camera pose is given and every object must be retrieved from the B1K asset catalog (departures from ACDC's published setup are listed in Appendix A). ACDC is the reference because it publishes scene-reconstruction fidelity under a defined protocol; the nearest comparable system, PRISM, reports policy success rather than reconstruction fidelity and had released no code at the time of writing (Ko et al., 2026).

In this experiment, each agent rebuilds one B1K room (Li et al., 2024a) from one photograph of it, and the reconstruction is measured against the room's known 3D layout.

Agents and the task. Seven code agents run the same task: Claude Opus 5, Claude Fable 5.1, Claude Fable 5 and Claude Opus 4.6 through the Claude Agent SDK, the programmatic harness of Claude Code, and GPT-6 Astra, GPT-5.6 Sol and GPT-5.5 through OpenAI Codex. The SDK is used rather than the interactive client so that every run's transcript is persisted and can be audited for instruction adherence and potential cheating (Appendix C). Each agent is used as shipped with default tools. Ground truth can leak through two channels, and both are controlled. First, the model's weights: B1K is public, so a model may have memorised its rooms. Rooms are therefore referred to by obscure ids only. A recitation probe, which gives the model a room's real B1K name with no image and no tools and asks it to list the room's assets, scores zero on every reported room for Opus 5 and for GPT-6 Astra; for Fable 5.1 it names one scored asset in each of two rooms, both assets that agents with a zero probe also retrieved from the photograph (Appendix C). Second, the filesystem: the ground-truth files sit beside the workspace. During a session these files, the library's scene directories, and every other run's directory are unreadable, and the agent's web tools are disabled. Afterwards, every tool input in the transcript, sub-agents included, is searched for attempts to read them. Seven reported runs made such attempts, all denied, and one further run (Fable 5.1, room J) read the benchmark's own README, a file outside the lock that holds no room data, and is kept with the audit flag it received (Appendix C); none obtained ground-truth content. In every arm the whole task is the following prompt:

```
Input:  scene.png in this directory is a photo of one room built from
BEHAVIOR-1K assets.  The asset catalog is catalog.jsonl (one line per model:
category, model, bbox_size in metres).  render.py is an OmniGibson example
that renders a scene json from the photo's camera (`source env.sh && og_py
render.py --pred pred.json`, about a minute; the json format is in its
```

```
header).  Modify it or write your own scripts as needed:  walls, floor,
lighting and anything else in the background are yours to build; furniture
must come from the catalog.  cam.json holds that camera:  {{CAMERA}}
Code up a digital twin of the room as pred.json.
Regenerate and rerender until reviewer agent approves for similarity.
Reviewer compares similar objects and scene background and make sure to use
the most similar ones, also Check whether object detected is indeed sensible
for the placement
Do not read anything under behavior-1k-assets/scenes.
```

The scene file the agent writes lists each asset's position, orientation and size. The prompt instructs the agent to revise until a reviewer agent approves, but no reviewer prompt is provided: the agent must create the reviewer itself, as a sub-agent whose instructions and model it chooses. In every arm the session ends only when the agent ends it, since no turn, time or cost cap is imposed. The settings of each agent arm and the room list are given in Appendix C, and Appendix B compares Opus 5 runs on five rooms with and without the prompt sentence "Modify it or write your own scripts as needed".

Results. ACDC's exact evaluation rooms were never released, so its published main-paper table is quoted as is. The agents are scored on eight rooms assembled here, a bedroom, a living room, an office, a kitchen, a dining room, a bar and two bathrooms, spanning 6–14 scored objects and 3.5–9.0 m. Room selection, the camera procedure and the scoring rules are given in Appendix C.

GPT-6 Astra performs best of the seven agents on the eight rooms, no run having obtained ground-truth content (Table 3 in Appendix C, with ACDC's published values (Dai et al., 2024) for reference). It assigns the correct category to 72% of the scored objects and retrieves the exact asset model for 66% of them. Its placement error, the distance between the centre of a placed object and the centre of its matched ground-truth object, is 14 cm in the median run. It is the best agent on 5 of the 8 rooms by placement error and on 5 of the 8 by centred IoU. GPT-6 Astra places fewer scored objects than the rooms contain (123 of 150); an unplaced object counts as wrong for category, model and both IoU metrics but does not enter placement error. A gap to ACDC's curated pipeline, whose main-paper table reports a median placement error of 6 cm on its four rooms, remains. Figure 2 shows the input photograph beside each agent's final render for the best, median and worst room by mean placement error, and Figure 6 in Appendix C shows all eight rooms.



Figure 2: Input photograph and each agent's final render from its first run, from the same fixed camera. Rows are the best, median and worst of the eight rooms by mean $\mathcal{L}_2$ over the seven agents (rooms G, B and F).

More strikingly, reconstruction performance improves substantially from one model generation to the next within each harness provider (Figure 3, top). Opus 4.6, released under six months before Opus 5, retrieves the exact asset model for 6% of the scored objects where Opus 5 retrieves it for 52%, and Opus 4.6 median-run placement error is 153 cm where Opus 5's is 39 cm. In the Codex harness, GPT-5.6 Sol retrieves the exact asset model for 32% of the scored objects where GPT-5.5 retrieves it for 11%, and GPT-5.6 Sol median-run placement error of 76 cm is a third lower than GPT-5.5's 120 cm. The same direction holds at the next Codex step: GPT-6 Astra, released eight weeks after GPT-5.6 Sol, retrieves the exact asset model for 66% of the scored objects where GPT-5.6 Sol retrieves it for 32% and places objects within 14 cm in the median run where GPT-5.6 Sol places them within 76 cm. Two caveats apply across all agents. The protocol does not penalise extra objects, and GPT-5.5 places nearly twice as many scored-category objects as the rooms contain. Orientation is recovered poorly by every agent, at about one radian of raw error, and

ACDC's 0.08 rad is reported after a symmetry restriction it does not define, so the two values are not comparable.

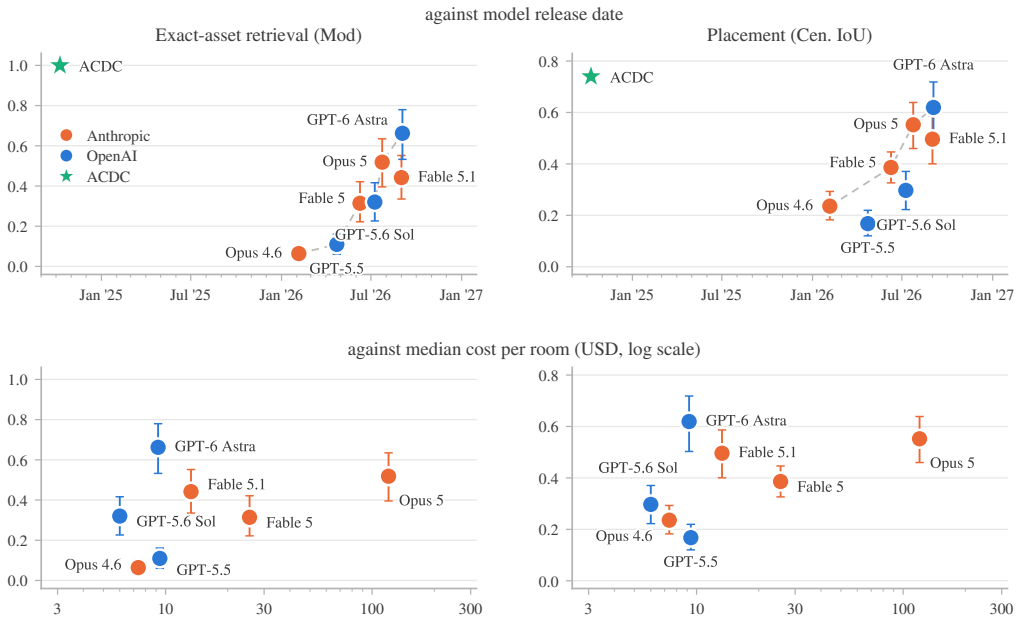

Figure 3: Room reconstruction performance against model release date (top row) and against median cost per room in US dollars on a logarithmic axis (bottom row), for exact-asset retrieval (Mod., left column) and IoU after centre alignment (Cen. IoU, right column). Higher is better. Each point is one agent's mean over the eight rooms with initial and replication run pooled, at the model's release date or at the median cost of its 16 runs, and bars are 95% bootstrap intervals of that mean. In the top row the star is ACDC's published result (Dai et al., 2024) and the dashed grey line traces the running best agent.

GPT-6 Astra gives the highest performance per unit cost of the seven agents and Fable 5.1 the highest of the four Claude agents, at median costs of $9 and $13 per room against $120 for Opus 5, which reaches its performance through far more work per room, 335 turns and four reviewer sub-agents at the median where Fable 5.1 takes about 90 turns and spawns one (Figure 3, bottom). Codex costs are list-price estimates, since those runs were billed to a subscription. Discussions in online forums such as Reddit also report that Opus 5 tends to use more thinking turns, but the why remains a mystery.

4 Code Reasoning as a Generic Harness

Sections 3 and 5 establish what the agents build; this section examines how. FAEA analysed the traces of a code agent writing manipulation trajectories (Tsui et al., 2026). The same analysis, applied to every Opus 5 room run of Section 3, finds the same loop—write a script, execute it, inspect the output, revise—retargeted from trajectories to scenes with no change to the harness.

Tool orchestration. The 16 runs, two per room, contain 7,690 tool calls. Shell execution dominates at 65.0%, file reading follows at 32.6%, and the remaining tools share 2.4%; each run spawns 3.9 reviewer sub-agents and launches the simulator 56.4 times on average. The counts describe a write–render–inspect cycle, and their shape matches the Bash-dominant distribution FAEA reports on manipulation benchmarks, with one departure: where FAEA's agent created scripts with the dedicated file-writing tool, this agent creates them from within shell commands, so file writes are counted under shell execution rather than under the file-writing tools. Appendix G reports the per-tool counts.

Analytic bootstrap. A recurring first step in the traces is analytic rather than iterative: before placing any object, the agent writes a projection script from the supplied camera and back-projects image edges into world coordinates. The main agent's final summary in the room J round-1 run reads as follows (transcript quotes are verbatim, with code spans set in typewriter type):

> *"every number in `layout.py` comes from back-projecting image edges through `cam.json` — leg contacts on the floor plane gave the partition plane `x = -6.50`; the wall/floor line gave `y = -47.87`; the tile top edge gave `z = 2.174`"*

Wall planes, footprints and heights enter the scene file as solved quantities, and iteration then corrects what the analytic solution does not reach.

Hypothesis-driven revision. When a render disagrees with the photograph, the agent proceeds as FAEA describes: it measures the mismatch, states a cause, and tests the cause, rather than perturbing coordinates. A reviewer sub-agent in the room F round-1 run, judging a mid-iteration render of a panel whose silhouette would not fit, writes:

> *"Unprojecting the photo's edge onto horizontal planes gives x = −4.96 @ y=4.8 down to −4.11 @ y=0.1 — i.e. it is a **raking eave**, not a flat panel cut at constant x, which is why `LATH_X = -4.55` cannot reproduce it."*

The hypothesis names the geometry and explains why the current constant cannot fit it; the fix follows from the measurement. The same procedure produces tooling: one run diagnoses that asset origins are offset from bounding-box centres and writes a calibration script that corrects each position against the renderer's reported bounding boxes until the rendered box matches the target.

5 Extending to a new domain: farm twins

In the room experiment, the camera pose is supplied and reconstruction fidelity is scored against known 3D ground truth. The question here is whether the two strongest of the same unmodified agents, Codex with GPT-6 Astra and Claude Code with Opus 5, extend to a domain with neither: farm fields with no calibrated camera and no 3D reference. Each agent must instead read the photograph—horizon, row geometry, plant habit, scale cues—and author the scene from it. Specialty-crop farming is chosen as the domain because it carries major sustainability and economic impact and still faces major automation challenges (Vougioukas, 2019; Bac et al., 2014). A photograph is also the input a grower can supply with a phone camera, without specialized instruments or calibration, which makes it the natural entry point for a farm twin. Procedural crop-field generators such as CropCraft (Bertoglio et al., 2025) assemble randomized fields from a configuration file that no photograph constrains, and language models have been used to author agricultural simulation scenes from text (Yoncalik et al., 2026). Recovering procedural plant descriptions from imagery has been studied directly, by training a vision-language model to emit plant-architecture representations from simulated images and by prompting foundation models to fill plant simulation configurations in context (Yun et al., 2026a;b). Here each agent instead places the generator's plants and composes the whole scene against the photograph.

The test involves 46 twins from each agent across ten crops, apple, strawberry, lettuce, broccoli, blueberry, table grape, peach, sweet cherry, tomato and sweet potato, five photographs per crop except broccoli with two and blueberry with four, each built from one online photograph and validated against it (Figure 4). The 46 photographs are openly licensed images from Wikimedia Commons, Flickr and rawpixel. The first is a canonical example of harvest automation in agriculture (Silwal et al., 2017; Zhang et al., 2024). The other nine are among the most labor-intensive fruit and vegetable crops per acre in national data (Martin, 2020).

Whereas the room experiment isolates the agent for controlled measurement, the farm experiment measures how much each agent automates from a single top-level prompt covering all the photographs, mimicking how a real user would ask the agent. In this testbed, both agents have access to Isaac Sim (NVIDIA, 2026b), a procedural plant generator, Helios (Bailey, 2019), and a library of supporting assets absent from the room testbed. The library provides plant models for each crop, sky domes covering clear to overcast conditions, and tileable ground materials. The plant models are grown with Helios, using the nearest species in its library for the four crops it lacks. The sky domes come from the openly licensed Poly Haven collection (Poly Haven, 2026). The ground materials are



Figure 4: Farm twins of apple, strawberry, lettuce and broccoli. Each column is one crop's reference photograph above the final path-traced render of the twin each of the two agents authored from it. Photograph credits are given in Appendix F.

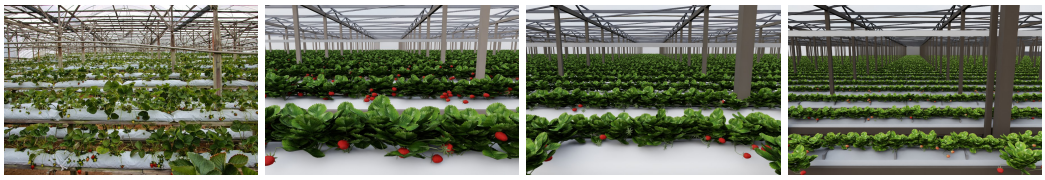

Figure 5: Camera correction across rounds in the greenhouse strawberry run. The Opus 5 agent first solved a low viewpoint from the bag-row heights, then raised the camera twice until renders showed the bench edges as the photograph requires. Renders are the real-time previews used during iteration.

grass from the NVIDIA Omniverse Base Materials library (NVIDIA, 2026a), with leafy grass and brown mud soil from Poly Haven. The prompt is reproduced in Appendix D. Each agent then wrote its own generator and reviewer sub-agent prompts and ran them.

During the iterations, each agent estimates and corrects the camera itself from the horizon and the crop rows' vanishing point (Figure 5, with camera solutions and correction traces for the first eight Opus 5 runs in Appendix D), places plants pre-grown with the procedural plant-architecture model of the Helios framework (Bailey, 2025) and fits their scale, spacing and colour to the photograph (plant-configuration traces in Appendix D), and authors the remaining objects as primitives at photo-measured dimensions (structure traces in Appendix D). Validation is render-to-photograph similarity under pretrained vision encoders (Oquab et al., 2024; Radford et al., 2021), ranked among all the twins' reference photographs (the metric is specified in Appendix E).

Astra and Opus 5 are comparable at constructing these novel scenes. Under DINOv2, 24 of the 46 Astra renders and 22 of the 46 Opus 5 renders rank their own photograph first among the 46 references, the mean similarity of matched render–photograph pairs is 0.52 and 0.57 against 0.20 for mismatched pairs, and reference photographs of the same crop average 0.49, so both matched-pair means sit above the level that real photographs of the same crop reach against each other. Under the same encoder, 30 of the 46 twins of each agent rank their own photograph first among the two to five photographs of their own crop, so the embedding discriminates between scenes of one crop and not only between crops. CLIP compresses the range, with matched pairs at 0.76 for Astra and 0.80

for Opus 5 against 0.67 and 0.69 in the same order for mismatched pairs and same-crop photographs at 0.79, and ranks the own photograph first for 22 of the Astra renders and 25 of the Opus 5 renders. Neither agent ranks more twins first under both encoders. Astra built its 46 twins in one Codex session of two hours, against over 23 hours across two campaigns for Opus 5. The weakest twin under DINOv2 by own-photograph similarity is the fifth peach twin for Astra and the third sweet potato twin for Opus 5, and CLIP ranks those photographs seventh and first, so the two encoders disagree on which twin is weakest; per-twin values are in Table 5.

6 TASK FORMALIZATION

The internals of Claude Code and Codex, and the way the frontier models behind them are trained, are not public, so nothing in this section describes how these systems are built. The formalization below is a mental model of the observed behaviour, assembled from the run transcripts of Sections 3 and 5 and from the structure of open-source code agents.

The FASC setting of Sections 3 and 5 shares one structure, stated here formally: scene construction as iterative program synthesis, following the framework of FAEA (Tsui et al., 2026). Given a task instruction ℓ, a reference photograph I, and a tool set $\mathcal{T}$ (the agent's shipped file, shell, and sub-agent tools, through which it invokes a renderer), the agent generates a sequence of script attempts $\{\sigma_1, \sigma_2, \ldots, \sigma_N\}$. Each attempt σ_i is a sequence of tool invocations that writes or revises a scene program, whose output is a scene file listing each object's asset, position, orientation, and size. The agent maintains accumulated context

$$\mathcal{C}_i = \{(\sigma_j, o_j) \mid j < i\}, \tag{1}$$

where o_j denotes the observations from attempt j: the render, error messages, and any reviewer verdict. Each new attempt is generated conditioned on instruction, photograph, tools, and history,

$$\sigma_i \sim \mathrm{LLM}(\ell, I, \mathcal{T}, \mathcal{C}_i). \tag{2}$$

Two departures from FAEA follow from the task. First, an attempt produces a scene rather than a trajectory: where FAEA's attempts end in a terminal simulator state, an attempt here ends in a render $R(\sigma_i)$ of its scene file. Second, the evaluator splits in two. FAEA's evaluator $\mathcal{E}$ returns success inside the loop; here the in-loop judgment comes from a reviewer the agent itself creates and instructs—an internal evaluator comparing $R(\sigma_i)$ to I—while the external evaluator $\mathcal{E}$ is withheld until the session ends. On rooms, $\mathcal{E}$ is ACDC's metric suite against the room's known 3D layout (Section 3); on farms, where no 3D reference exists, it is encoder similarity between the final render and the photograph (Section 5). The agent never observes $\mathcal{E}$, so a run terminates only when the internal reviewer approves or the agent ends its session; no gradient updates occur.

7 CONCLUSION

This paper measured what unmodified frontier code agents achieve at digital-twin scene construction from one photograph, the FASC setting, on eight BEHAVIOR-1K rooms with 3D ground truth and 46 farm photographs without it. Performance rises steeply from one frontier model generation to the next within each harness provider, which is consistent with the hypothesis that purpose-built harnesses and specialized models compensated for capability that newer general models now supply. The twins are certified on geometry and appearance, not on dynamics or on policy transfer.

Language processing moved from specialised models to a single large language model, and a capable general agent emerged on top of it. FAEA (Tsui et al., 2026) and ENPIRE (Xiao et al., 2026) showed that generic frontier agents can produce embodied trajectories and autonomously improve manipulation policies, and this paper shows that the same class of agent can reconstruct scenes. Multimodal models are taking on more of the tasks that only specialised models or tools could perform. A generic frontier agent built on top of them, accessible as an ordinary program such as Claude Code or Codex, can be expected to bring that intelligence, still largely confined to text today, into embodied settings. The emergent physics understanding capabilities reported for large video models (Wiedemer et al., 2025) point the same way. The trend indicates that embodied capability will keep improving while requiring less embodiment-specific engineering.

This paper has several limitations. The training of the latest frontier models is opaque, so for the models used here it is not clear whether this behaviour emerged from progress in the underlying model or was specifically trained for. The sample sizes are limited by available resources.

REPRODUCIBILITY STATEMENT

The code and its execution environment are attached as supplementary material. Nonetheless, readers are encouraged to run their own code agent with a standard Isaac Sim installation and ask it, in whatever wording they like, with a photo, to create a digital twin. A native agent setup adapted without changes on two friends' machines.

AI USE STATEMENT

This paper results from many conversations between the author and Claude Code with a Max 20X subscription. The author initially drafted the idea. Each step iterated between a token-heavy agent team and manual review. The steps span experiment planning and execution, main text and figure generation, and in-depth reviewing. The author reviewed all AI-assisted work and is responsible for the content.

REFERENCES

C. Wouter Bac, Eldert J. van Henten, Jochen Hemming, and Yael Edan. Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, 31(6): 888–911, 2014. doi: 10.1002/rob.21525.

Brian N. Bailey. Helios: A scalable 3d plant and environmental biophysical modeling framework. *Frontiers in Plant Science*, 10:1185, 2019. doi: 10.3389/fpls.2019.01185.

Brian N. Bailey. A generalized framework for procedural generation of three-dimensional static and dynamic plant model geometries, 2025. URL https://arxiv.org/abs/2512.17966.

Shmuel Berman, Michael Ilie, Jia Deng, and Daniel Freeman. Claude plays robotics. https://www.anthropic.com/research/claude-plays-robotics, 2026. Anthropic Frontier Red Team research report, 9 July 2026. Accessed 20 August 2026.

Riccardo Bertoglio, Cyrille Pierre, Johann Laconte, and Roland Lenain. CropCraft: A procedural world generator for robotic simulation of agricultural tasks, 2025. URL https://arxiv.org/abs/2511.02417.

Ziang Cao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. PhysX-Anything: Simulation-ready physical 3D assets from single image. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. arXiv:2511.13648.

Guanxiong Chen, Qianjun Xia, Jiawei Peng, Heng Zhang, Bole Ma, Justin Qian, Ziyi Jiao, Bingyang Zhou, Luoxin Ye, Kaifeng Zhang, Kunyi Wang, Weijia Zeng, Yunuo Chen, Pengzhi Yang, Ziqiu Zeng, Siyuan Luo, Huamin Wang, Chao Liu, Alan Yuille, Fan Shi, Changxi Zheng, Yunzhu Li, Chenfanfu Jiang, and Peter Yichen Chen. Agentic Real2Sim: Physics-based world modeling with vision-language agents, 2026. URL https://arxiv.org/abs/2607.19190.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation, 2025. URL https://arxiv.org/abs/2506.18088.

Zoey Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Karthikeya Vemuri, Alan Wu, Dieter Fox, and Abhishek Gupta. URDFormer: A pipeline for constructing articulated simulation environments from real-world images. In *Robotics: Science and Systems (RSS)*, 2024. URL https://arxiv.org/abs/2405.11656. arXiv:2405.11656.

Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. In *Conference on Robot Learning (CoRL)*, 2024. URL https://arxiv.org/abs/2410.07408. ACDC; arXiv:2410.07408.

Prithwish Dan, Kushal Kedia, Angela Chao, Edward Duan, Maximus Adrian Pace, Wei-Chiu Ma, and Sanjiban Choudhury. X-Sim: Cross-embodiment learning via real-to-sim-to-real. In *Conference on Robot Learning (CoRL)*, 2025. URL https://arxiv.org/abs/2505.07096. PMLR 305:816–833; arXiv:2505.07096.

Karim Elmaaroufi, Justin Svegliato, Sarunas Kalade, Graham Schelle, Sanjit A. Seshia, and Matei Zaharia. RHO: Your coding agent is secretly a roboticist, 2026. URL https://arxiv.org/abs/2606.16458.

Letian Fu, Justin Yu, Karim El-Refai, Ethan Kou, Haoru Xue, Huang Huang, Wenli Xiao, Guanzhi Wang, Dantong Niu, Fei-Fei Li, Guanya Shi, Jiajun Wu, Shankar Sastry, Yuke Zhu, Ken Goldberg, and Linxi Fan. CaP-X: A framework for benchmarking and improving coding agents for robot manipulation. In *International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2603.22435. arXiv:2603.22435.

Chenghao Gu, Haolan Kang, Junchao Lin, Jinghe Wang, Duo Wu, Shuzhao Xie, Fanding Huang, Junchen Ge, Ziyang Gong, Letian Li, Hongying Zheng, Changwei Lv, and Zhi Wang. IGen: Scalable data generation for robot learning from open-world images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. arXiv:2512.01773.

Yunhai Han, Jianuo Qiu, Linhao Bai, Ziyu Xiao, Zihang Zeng, Yangcen Liu, Zhaodong Yang, Shalin Jain, Wenrui Ma, Jiaqi Fu, Yuqian Zheng, Manisha Natarajan, Muhammad Zubair Irshad, Kenneth Shaw, Matthew Gombolay, Zsolt Kira, and Harish Ravichandar. Video2Sim2Real: Full-stack autonomous dexterous skill acquisition from a single human video, 2026. URL https://arxiv.org/abs/2606.08828.

Yumeng He, Ying Jiang, Jiayin Lu, Yin Yang, and Chenfanfu Jiang. SPARK: Sim-ready part-level articulated reconstruction with VLM knowledge, 2025. URL https://arxiv.org/abs/2512.01629.

Yumeng He, Yichen Song, Xiaotian Yang, Weijia Zhang, Zanwei Zhou, Junru Gong, Xiaokang Yang, and Yunbo Wang. NeoWorld-Pro: Programming interactive scenes from monocular images for embodied simulation, 2026. arXiv:2608.24212.

Pu Hua, Minghuan Liu, Annabella Macaluso, Yunfeng Lin, Weinan Zhang, Huazhe Xu, and Lirui Wang. GenSim2: Scaling robot data generation with multi-modal and reasoning LLMs. In *Conference on Robot Learning (CoRL)*, 2024. URL https://arxiv.org/abs/2410.03645. arXiv:2410.03645.

Arhan Jain, Mingtong Zhang, Kanav Arora, William Chen, Marcel Torne, Muhammad Zubair Irshad, Sergey Zakharov, Yue Wang, Sergey Levine, Chelsea Finn, Wei-Chiu Ma, Dhruv Shah, Abhishek Gupta, and Karl Pertsch. PolaRiS: Scalable real-to-sim evaluations for generalist robot policies. In *Robotics: Science and Systems (RSS)*, 2026. URL https://arxiv.org/abs/2512.16881. arXiv:2512.16881.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. PhysTwin: Physics-informed reconstruction and simulation of deformable objects from videos. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. URL https://arxiv.org/abs/2503.17973. arXiv:2503.17973.

Yufeng Jin, Jianfei Guo, Xiaogang Jia, Yu Deng, Zechu Li, Han Liu, Weiran Liao, Vignesh Prasad, Mathias Franzius, Gerhard Neumann, and Georgia Chalvatzaki. Nautilus: From one prompt to plug-and-play robot learning, 2026a. URL https://arxiv.org/abs/2605.11665.

Zhao Jin, Zhengping Che, Tao Li, Zhen Zhao, Kun Wu, Yuheng Zhang, Yinuo Zhao, Zehui Liu, Qiang Zhang, Xiaozhu Ju, Jing Tian, Yousong Xue, and Jian Tang. ArtVIP: Articulated digital assets of visual realism, modular interaction, and physical fidelity for robot learning. In *International Conference on Learning Representations (ICLR)*, 2026b. URL https://arxiv.org/abs/2506.04941. arXiv:2506.04941.

Zhi Jing, Siyuan Yang, Jicong Ao, Ting Xiao, Yu-Gang Jiang, and Chenjia Bai. HumanoidGen: Data generation for bimanual dexterous manipulation via LLM reasoning, 2025. URL https://arxiv.org/abs/2507.00833.

Haoqiang Kang, Xiaokang Ye, Yuhan Liu, Siddhant Hitesh Mantri, Lingjun Mao, James Fleming, Drishti Regmi, and Lianhui Qin. SimWorld Studio: Automatic environment generation with evolving coding agent for embodied agent learning, 2026. URL https://arxiv.org/abs/2605.09423.

Pushkal Katara, Zhou Xian, and Katerina Fragkiadaki. Gen2Sim: Scaling up robot learning in simulation with generative models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6672–6679, 2024. URL https://arxiv.org/abs/2310.18308. DOI 10.1109/ICRA57147.2024.10610566; arXiv:2310.18308.

Mukul Khanna, Yongsen Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X. Chang, and Manolis Savva. Habitat synthetic scenes dataset (HSSD-200): An analysis of 3D scene scale and realism tradeoffs for ObjectGoal navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16384–16393, 2024. doi: 10.1109/CVPR52733.2024.01550. URL https://arxiv.org/abs/2306.11290.

Jeonghwan Kim, Yushi Lan, Yongwei Chen, Hieu Trung Nguyen, Chuanyu Pan, and Xingang Pan. SceneConductor: 3D scene generation from a single image with multi-agent orchestration, 2026. URL https://arxiv.org/abs/2606.08402.

Dogyu Ko, Haneul Kim, Chanyoung Yeo, Dowoon Lee, Taeho Park, and Hyoseok Hwang. PRISM: Personalized robotic dataset generation via image-based scene and motion synthesis, 2026. URL https://arxiv.org/abs/2607.04880.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An interactive 3D environment for visual AI, 2017. URL https://arxiv.org/abs/1712.05474. Revised through v4, 2022.

Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://arxiv.org/abs/2410.13882. arXiv:2410.13882.

Hojoon Lee, Ajay Subramanian, Ben Abbatematteo, Vijay Veerabadran, Pedro Matias, Karl Ridgeway, and Nitin Kamra. RDA: Reward design agent for reinforcement learning. In *Reinforcement Learning Conference (RLC)*, 2026a. URL https://arxiv.org/abs/2606.01672. arXiv:2606.01672.

Inhee Lee, Sangwon Baik, Sungjoo Kim, Hyeonwoo Kim, Hyunsoo Cha, and Hanbyul Joo. SimuScene: Simulation-ready compositional 3D scene reconstruction from a single image, 2026b. URL https://arxiv.org/abs/2606.03994.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Wensi Ai, Benjamin Martinez, et al. BEHAVIOR-1K: A human-centered, embodied AI benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024a. URL https://arxiv.org/abs/2403.09227. A preliminary version appeared at CoRL 2022.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Conference on Robot Learning (CoRL)*, 2024b.

Zechu Li, Yufeng Jin, Xiaoyang Liu, Puze Liu, Vignesh Prasad, Carlo D'Eramo, and Georgia Chalvatzaki. HARBOR: A harness framework for agentic robot reinforcement learning, 2026. URL https://arxiv.org/abs/2606.08610.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500, 2023. doi: 10.1109/ICRA48891.2023.10160591. URL https://arxiv.org/abs/2209.07753.

Parker Liu, Chenxin Li, Zhengxin Li, Yipeng Wu, Wuyang Li, Zhiqin Yang, Zhenyuan Zhang, Yunlong Lin, Sirui Han, and Brandon Y. Feng. IR3D-Bench: Evaluating vision-language model scene understanding as agentic inverse rendering, 2025. URL https://arxiv.org/abs/2506.23329.

Xiang Liu, Sen Cui, Guocai Yao, Zhong Cao, Jingheng Ma, Min Zhang, and Changshui Zhang. Scene2Demo: Self-evolving embodied data generation via object-action graph, 2026. URL https://arxiv.org/abs/2602.12065.

Jasper Lu, Zhenhao Shen, Yuanfei Wang, Shugao Liu, Shengqiang Xu, Shawn Xie, Jingkai Xu, Feng Jiang, Jade Yang, Chen Xie, and Ruihai Wu. From seeing to simulating: Generative high-fidelity simulation with digital cousins for generalizable robot learning and evaluation, 2026a. URL https://arxiv.org/abs/2604.15805.

Runyu Lu, Yubo Wu, Ethan Kou, Letian Fu, Wenli Xiao, Ajay Mandlekar, Yinzhen Xu, Guanya Shi, Ken Goldberg, Ang Chen, Mosharaf Chowdhury, Yuke Zhu, Linxi Fan, and Guanzhi Wang. ASPIRE: Agentic /Skills discovery for robotics, 2026b. URL https://arxiv.org/abs/2607.00272.

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. URL https://arxiv.org/abs/2310.12931. arXiv:2310.12931.

Philip Martin. Labor in fruit and vegetable agriculture. Rural Migration News Blog, University of California, Davis, November 2020. URL https://migration.ucdavis.edu/rmn/blog/post/?id=2497.

Yansong Ning, Jingwen Ye, Zhongkai Wu, Yang Sun, Yiqin Zhu, Xingyi Li, Weidong Zhang, and Hao Liu. VibeWorlding: Can multimodal agents construct 3D open worlds end-to-end?, 2026. URL https://arxiv.org/abs/2608.15265.

NVIDIA. Base Materials Pack. Downloadable Asset Packs, NVIDIA Omniverse USD documentation, https://docs.omniverse.nvidia.com/usd/latest/usd_content_samples/downloadable_packs.html, 2026a. 161 drag-and-drop MDL materials built on OmniPBR, distributed with NVIDIA Omniverse and Isaac Sim as Materials/Base on the Omniverse content server; the Natural/Grass_Cut and Natural/Grass_Countryside sets are used here. Accessed September 2026.

NVIDIA. NVIDIA Isaac Sim: Robotics simulation and synthetic data generation. https://developer.nvidia.com/isaac/sim, 2026b. Open source reference framework built on NVIDIA Omniverse libraries, version 5.1. Accessed September 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt.

Nicholas Pfaff, Thomas Cohn, Sergey Zakharov, Rick Cory, and Russ Tedrake. SceneSmith: Agentic generation of simulation-ready indoor scenes. In *International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2602.09153. Spotlight; arXiv:2602.09153.

Poly Haven. Poly Haven: The public 3D asset library. https://polyhaven.com/, 2026. All assets (HDRIs, textures and 3D models) released under CC0 (https://polyhaven.com/license). Accessed September 2026.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. THE COLOSSEUM: A benchmark for evaluating generalization for robotic manipulation. In *Robotics: Science and Systems (RSS)*, 2024.

Minghan Qin, Yuang Wang, Xiuyu Yang, Yushi Long, Yujian Zhang, Ruihuan Wang, Kai Ye, Yangang Zhang, and Hang Li. Lucida: Parse, generate, and place for composable real-to-sim scene modeling, 2026. arXiv:2608.30821.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Nadun Ranawaka, Josiah Wong, Wei-Lin Pai, Wei-Teng Chu, Tianyuan Dai, Masoud Moghani, Hang Yin, Yunfan Jiang, Wesley Durbano, Brandon Huynh, Yu Fang, Danfei Xu, Ruohan Zhang, Li Fei-Fei, Linxi Fan, Bowen Wen, Ajay Mandlekar, and Yuke Zhu. SimFoundry: Modular and automated scene generation for policy learning and evaluation, 2026. URL https://arxiv.org/abs/2606.28276.

Abhisesh Silwal, Joseph R. Davidson, Manoj Karkee, Changki Mo, Qin Zhang, and Karen Lewis. Design, integration, and field evaluation of a robotic apple harvester. *Journal of Field Robotics*, 34(6):1140–1159, 2017. doi: 10.1002/rob.21715.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating situated robot task plans using large language models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11523–11530, 2023. doi: 10.1109/ICRA48891.2023.10161317. URL https://arxiv.org/abs/2209.11302.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 251–266, 2021. URL https://arxiv.org/abs/2106.14405. ReplicaCAD; arXiv:2106.14405.

Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. In *Robotics: Science and Systems (RSS)*, 2024.

Brian Y. Tsui, Alan Y. Fang, and Tiffany J. Hwu. Demonstration-free robotic control via LLM agents. *arXiv preprint arXiv:2601.20334*, 2026. URL https://arxiv.org/abs/2601.20334. Accepted to IEEE/RSJ IROS 2026.

Stavros G. Vougioukas. Agricultural robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:365–392, 2019. doi: 10.1146/annurev-control-053018-023617. URL https://www.annualreviews.org/doi/10.1146/annurev-control-053018-023617.

Hongyu Wang, Jingquan Wang, Ashvin Anilkumar, Bocheng Zou, Radu Serban, and Dan Negrut. ChronoAgentic: A code-based multi-agent world simulator for physically grounded simulation construction, 2026a. URL https://arxiv.org/abs/2605.14398.

Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. GenSim: Generating robotic simulation tasks via large language models. In *International Conference on Learning Representations (ICLR)*, 2024a. URL https://arxiv.org/abs/2310.01361. arXiv:2310.01361.

Xinjie Wang, Liu Liu, Taojun Ding, Andrew Choi, Chaodong Huang, Mengao Zhao, Ziang Li, Jackson Jiang, Chunlei Yu, Shengxiang Liu, Wei Xu, and Zhizhong Su. EmbodiedGen V2: An agentic, simulation-ready 3D world engine for embodied AI, 2026b. URL https://arxiv.org/abs/2607.07459.

Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. RoboGen: Towards unleashing infinite data for automated robot learning via generative simulation. In *International Conference on Machine Learning (ICML)*, 2024b. URL https://arxiv.org/abs/2311.01455. arXiv:2311.01455.

Zihan Wang, Jiashun Wang, Jeff Tan, Yiwen Zhao, Jessica Hodgins, Shubham Tulsiani, and Deva Ramanan. CRISP: Contact-guided Real2Sim from monocular video with planar scene primitives. In *International Conference on Learning Representations (ICLR)*, 2026c. URL https://arxiv.org/abs/2512.14696. arXiv:2512.14696.

Thaddäus Wiedemer, Yuxuan Li, Paul Vicol, et al. Video models are zero-shot learners and reasoners. *arXiv preprint arXiv:2509.20328*, 2025.

Xinyuan Wu, Jingrao Zhang, Mengdi Xu, Henry K. Chu, Mingguang He, and Danli Shi. Operational digital twin clinics enable task-based evaluation of embodied AI, 2026a. URL https://arxiv.org/abs/2608.21416.

Zhuangzhe Wu, Yue Xin, Chengkai Hou, Minghao Chen, Yaoxu Lyu, Jieyu Zhang, and Shanghang Zhang. URDF-Anything+: End-to-end generation for simulation-ready articulated assets, 2026b. URL https://arxiv.org/abs/2603.14010.

Chong Xia, Kai Zhu, Zizhuo Wang, Fangfu Liu, Zhizheng Zhang, and Yueqi Duan. SimRecon: Simready compositional scene reconstruction from real videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. URL https://arxiv.org/abs/2603.02133. arXiv:2603.02133.

Wenli Xiao, Jia Xie, Tonghe Zhang, Haotian Lin, Letian Fu, Haoru Xue, Jalen Lu, Yi Yang, Cunxi Dai, Zi Wang, Jimmy Wu, Guanzhi Wang, S. Shankar Sastry, Ken Goldberg, Linxi Fan, Yuke Zhu, and Guanya Shi. ENPIRE: Agentic robot policy self-improvement in the real world, 2026. URL https://arxiv.org/abs/2606.19980.

Shuangyu Xie, Kaiyuan Chen, and Ken Goldberg. Revisiting the "Push-T" robot manipulation task with agentic robotics, 2026. URL https://arxiv.org/abs/2608.18227.

Yixuan Yang, Zhen Luo, Wanshui Gan, Jinkun Hao, Junru Lu, Jinghao Yan, Zhaoyang Lyu, and Xudong Xu. Code-as-Room: Generating 3D rooms from top-down view images via agentic code synthesis, 2026. URL https://arxiv.org/abs/2605.18451.

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. Holodeck: Language guided generation of 3D embodied AI environments. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://arxiv.org/abs/2312.09067. arXiv:2312.09067.

Arafa Yoncalik, Wouter Jansen, Nico Huebel, Mohammad Hasan Rahmani, and Jan Steckel. LLM-driven 3D scene generation of agricultural simulation environments. In *IEEE Conference on Artificial Intelligence (CAI)*, 2026. URL https://arxiv.org/abs/2602.11706. arXiv:2602.11706.

Heesup Yun, Isaac Kazuo Uyehara, Ioannis Droutsas, Earl Ranario, Christine H. Diepenbrock, Brian N. Bailey, and J. Mason Earles. A vision language model for generating XML-based organ-level plant architecture representations of cowpea from simulated images. *Frontiers in Artificial Intelligence*, 9:1844338, 2026a. doi: 10.3389/frai.2026.1844338. arXiv:2603.22622.

Heesup Yun, Isaac Kazuo Uyehara, Earl Ranario, Lars Lundqvist, Christine H. Diepenbrock, Brian N. Bailey, and J. Mason Earles. Using vision language foundation models to generate plant simulation configurations via in-context learning, 2026b. URL https://arxiv.org/abs/2603.08930.

Junyi Zhang, Jiaxin Ge, Hanjun Yoo, Letian Fu, Zihan Yang, Yaowei Liu, Raj Saravanan, Shaofeng Yin, Justin Yu, Dantong Niu, Zirui Wang, Roei Herzig, Ken Goldberg, Yutong Bai, David M. Chan, Ion Stoica, Angjoo Kanazawa, Jiahui Lei, Haiwen Feng, and Trevor Darrell. Playful agentic robot learning, 2026a. URL https://arxiv.org/abs/2606.19419.

Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-sim robot policy evaluation with Gaussian Splatting simulation of soft-body interactions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026b. arXiv:2511.04665.

Kaixiang Zhang, Kyle Lammers, Pengyu Chu, Zhaojian Li, and Renfu Lu. An automated apple harvesting robot—from system design to field evaluation. *Journal of Field Robotics*, 41(7):2384–2400, 2024. doi: 10.1002/rob.22268. URL https://onlinelibrary.wiley.com/doi/full/10.1002/rob.22268.

Shujie Zhang, Jingkun Yi, Weipeng Zhong, Zirui Zhou, Yangkun Zhu, Hanqing Wang, Xudong Xu, Weinan Zhang, and Chunhua Shen. RoboSnap: One-shot real-to-sim scene generation for generalizable robot learning and evaluation, 2026c. URL https://arxiv.org/abs/2607.06699.

Siheng Zhao, Jiageng Mao, Wei Chow, Zeyu Shangguan, Tianheng Shi, Rong Xue, Yuxi Zheng, Yijia Weng, Yang You, Daniel Seita, Leonidas Guibas, Sergey Zakharov, Vitor Guizilini, and Yue Wang. Robot learning from any images. In *Conference on Robot Learning (CoRL)*, 2025. URL https://arxiv.org/abs/2509.22970. RoLA; arXiv:2509.22970.

Alex Zook, Fan-Yun Sun, Josef Spjut, Valts Blukis, Stan Birchfield, and Jonathan Tremblay. GRS: Generating robotic simulation tasks from real-world images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 594–603, 2025. doi: 10.1109/CVPRW67362.2025.00064. URL https://arxiv.org/abs/2410.15536. arXiv:2410.15536.

A Departures from ACDC's protocol

In this paper's setup the agent is given the camera pose and floor height, whereas ACDC, by its published description, is given the intrinsics only and fits the floor plane from monocular depth; the asset catalog here has no thumbnails, whereas ACDC matches against rendered asset snapshots. ACDC does not state how predictions are paired with ground truth, and its sim-to-sim evaluation code was never released. The pairing rule used here, one Hungarian assignment on box centres as described in Appendix C, is therefore this paper's own choice rather than ACDC's. ACDC reports its orientation error only over objects outside a symmetry class, but its captions name that class without defining which objects fall in it. The protocol here excludes a fixed list of rotationally ambiguous objects instead, written down per object before any run, and reports the orientation error over every other object. Axis-aligned boxes are used, as ACDC's supplementary tables specify for Bbox IoU.

B Script-writing ablation

One sentence of the prompt in Section 3, placed after the description of the render script, changes what the agent builds: *"Modify it or write your own scripts as needed: walls, floor, lighting and anything else in the background are yours to build; furniture must come from the catalog."* It lets the agent edit or replace the render script and build the background by any means, and the background is rendered but never scored. The sentence was absent from the prompt file for part of the campaign, so rooms E, F, G, I and J each have one Opus 5 run without it and one with the full prompt, under the same harness and reviewer instruction. Every run in Table 3 used the full prompt. The runs without it appear only here, as the (–) rows of Table 1.

Behaviour. Without the sentence no run edited the render script, and every run built its background from catalog wall, floor and ceiling assets placed in the scene file. With the sentence three runs edited the render script and one replaced it, and two rooms built their background from primitive boxes and lights instead.

Table 1: Opus 5 on five rooms with the prompt lacking (–) or containing (+) the script-writing sentence, one run per cell. Cat., Mod.: fraction of the N ground-truth objects with correct category / asset model; $\mathcal{L}_2$: mean centre error of matched objects; Ori.: mean rotation error of matched non-symmetric objects, with flips (within 2° of 90°/180°) over scored orientations; Bbox, Cen.: 3D box IoU before and after centre alignment; N_{p}: scored objects placed (background excluded); USD: session cost; min: wall time. Mean rows: room means, $\mathcal{L}_2$ median in parentheses, flips and N_{p}/N pooled.

Room	Prompt	Cat.	Mod.	$\mathcal{L}_2$ cm	Ori. rad	Bbox	Cen.	N_{p}/N	USD	min
E bath	–	9/11	9/11	33	2.79 (8/9)	0.03	0.77	9/11	73	183
	+	5/11	5/11	8	3.14 (5/5)	0.27	0.39	5/11	59	121
F dining	–	2/6	0/6	403	1.58 (3/6)	0.04	0.13	7/6	65	107
	+	1/6	1/6	11	0.00 (0/1)	0.13	0.16	1/6	185	169
G office	–	5/8	4/8	25	0.55 (1/7)	0.60	0.69	10/8	186	127
	+	6/8	6/8	10	0.08 (0/7)	0.63	0.77	9/8	252	136
I bar	–	6/14	5/14	59	0.67 (2/7)	0.27	0.59	17/14	43	84
	+	4/14	4/14	108	1.35 (3/7)	0.10	0.46	15/14	117	156
J bath	–	7/9	7/9	17	0.00 (0/8)	0.58	0.76	8/9	64	228
	+	7/9	7/9	21	0.00 (0/9)	0.65	0.83	16/9	60	130
Mean	–	0.60	0.49	107 (33)	1.12 (14/37)	0.30	0.59	51/48	86	146
	+	0.49	0.49	32 (11)	0.91 (8/29)	0.36	0.52	46/48	135	142

Reading. Across the five rooms the mean centre error fell from 107 to 32 cm, mean Cat. fell from 0.60 to 0.49, mean Mod. was unchanged and cost rose about 1.6×. The centre error fell in E, F and G and rose in I and J. Each cell is one run, so room-level differences are indicative only. The G run with the sentence ended on a server-side API error and is scored on its final scene file.

C ROOM-EXPERIMENT DETAILS

Rooms. Table 2 lists the eight evaluation rooms by their obscure ids. Rooms are chosen by a fixed rule rather than manually: a list of desired room types (living room, bedroom, kitchen, bathroom, dining room, office, bar) is fixed, and at most one room per B1K house is taken. In each room only the eight largest assets are kept, with every instance of a kept model counting as a separate object; a model used in one room is excluded from all others. A room is accepted if it has at least five assets kept, at least five kept objects visible from the camera, and a scale (ACDC's definition: the largest distance between two objects) inside ACDC's range. All other furniture is deleted before the photograph is rendered, so the image shows exactly the scored objects and the background. ACDC does not describe how its rooms were prepared; only large furniture is kept here, so that object counts fall in the range of its table and every scored object is resolvable in one photograph. The photograph is rendered in OmniGibson from the footprint corner that sees the most kept objects, at a fixed inset and height, aimed at their centroid, with no retakes. Ground truth is every kept object visible in the photograph. Two candidate rooms were replaced before any run (one failed the recitation probe, one had too few scored objects), and no spare remains.

Scoring. Predictions are paired with ground truth by one category-agnostic Hungarian assignment on box centres with no distance gate. Unmatched ground truth counts as wrong for Cat. and Mod., scores zero IoU and is left out of $\mathcal{L}_2$ and Ori.; extra predictions are not penalised, so the predicted count is reported beside the scored count. The background (walls, floors, windows, doors, carpets) is rendered but never scored: predicted objects in those categories are dropped before pairing. Ori. skips a per-object list of rotationally ambiguous objects written before any run (Appendix A) and is reported raw with the count of exact quarter- or half-turn flips (fixed tolerance).

Claude arms. The four Claude agents ran through the Claude Agent SDK over the shipped Claude Code binary with its stock tool suite (file read, write and edit, shell, search, and sub-agent spawning). Permission prompts were bypassed, the two web tools were disabled so a model cannot look up B1K scene files online, and no MCP servers, hooks, or tools of any kind were added. One SDK default matters for interpretation: when no system prompt is supplied, the SDK launches the binary with an *empty* system prompt—so the Claude arms ran with Claude Code's tools but no system prompt at all, rather than under Claude Code's default prompt. When an agent created a reviewer, it wrote the brief and chose the reviewer's model itself. Of the 64 Claude runs, 7 spawned no reviewer (Fable 5 in rooms B and C and in both runs of room E; Opus 4.6 in one run of room E; Fable 5.1 in one run

Table 2: The eight evaluation rooms.

Room	Type	Scored objects	Scale (m)
B	bedroom	8	5.3
C	commercial kitchen	9	6.1
D	living room	10	4.3
E	bathroom	11	4.9
F	restaurant dining room	6	9.0
G	private office	8	3.5
I	bar	14	6.1
J	station bathroom	9	6.1

each of rooms D and J): each launched a background render, ended its turn while waiting for it, and was scored on the scene file at that point.

Fable 5.1 arm. Claude Fable 5.1 (`claude-fable-5-1`, released 2026-09-01) was added after the other arms under the same settings, prompt and harness code; its two runs per room are r7 and r17, with every r17 run started after the last r7 run had finished. This arm ran on claude-agent-sdk 0.2.151 with Claude Code CLI 2.1.258, because the 0.2.144 / 2.1.239 pair the other Claude arms ran on rejects the model; the harness code is identical. The recitation probe, rerun for this model, lists one of the scored assets of room D (a sofa, 1 of 10) and one of room G (a bookcase, 1 of 8) among mostly wrong guesses, and none in the other six rooms; both assets were also placed by agents whose probe scores zero on those rooms, so the probe does not separate recall from retrieval there, and both rooms are kept. One run (room J, r7) read the first 80 lines of the benchmark's README, a file outside the lock that describes the protocol and file layout and holds no room, object or camera data; the harness audit flags the run for a file name that text contains, the run is kept and pooled, and without it the arm's column in Table 3 would read 0.57 / 0.46 / 60 (40) / 0.94 / 0.25 / 0.51 for Cat., Mod., $\mathcal{L}_2$, Ori., Bbox IoU and Cen. IoU. Three runs (rooms E and F, r7; room G, r17) listed path names under an unlocked pre-migration copy of earlier run records while locating their workspace with `find`; no tool input read a file there. The first r7 attempts of rooms C and G ended after 12 to 14 minutes with no render, each waiting on a background render, and were rerun by decision of the author; the first attempts are retained beside the run records but not reported.

Codex arms. GPT-6 Astra, GPT-5.6 Sol and GPT-5.5 ran through OpenAI Codex in non-interactive mode with approvals and sandboxing bypassed—the analogue of the Claude arms' permission bypass, needed to launch the simulator—and with web search, browser, computer use, image generation, apps, and plugins disabled, ignoring any user configuration. Codex offers no empty-prompt option, so these arms ran under its vendor system prompt plus a per-turn multi-agent policy message. Its tool surface is a single shell tool with patch-apply and image-view commands rather than Claude Code's separate file tools, and its reviewer starts from a copy of the agent's prompt messages but none of its turns, followed by the agent's brief, where Claude Code's sub-agent receives the brief alone; its policy discourages sub-agents unless the user asks for them, which the prompt's "until reviewer agent approves" clause satisfied in every room. Reasoning effort was raised to high for all three models; the Claude arms ran at their default reasoning settings.

GPT-6 Astra arm. GPT-6 Astra (`gpt-6-astra`, released 2026-09-03) was added after the other arms under the same settings, prompt and harness code apart from the model's list-price entry; its two runs per room are r8 and r18, with every r18 run started after the last r8 run had finished. This arm ran on Codex CLI 0.153.4 where the other Codex arms ran on 0.150.1. The recitation probe was not run for the other Codex arms. For this model it was run through Codex with every tool

Table 3: Room reconstruction on ACDC's protocol: ACDC quoted on its own rooms; seven agents under one prompt on this paper's rooms

	ACDC	Claude Agent SDK				Codex		
	(quoted)	Opus 5	Fable 5.1	Fable 5	Opus 4.6	GPT-6 Astra	GPT-5.6 Sol	GPT-5.5
rooms / scored objects	4/39	8/75	8/75	8/75	8/75	8/75	8/75	8/75
objects placed / in the rooms	–	170/150	135/150	143/150	147/150	123/150	110/150	282/150
Cat. ↑	1.00	0.58	0.55	0.48	0.25	0.72	0.46	0.27
Mod. ↑	1.00	0.52	0.44	0.31	0.06	0.66	0.32	0.11
$\mathcal{L}_2$ cm ↓, mean (median)	8 (6)	54 (39)	59 (46)	75 (54)	147 (153)	24 (14)	101 (76)	123 (120)
Ori. rad ↓	0.08[†]	0.94[‡]	0.93[‡]	1.01[‡]	1.22[‡]	0.48[‡]	1.05[‡]	1.44[‡]
flips / scored	–	39/108	36/105	32/101	36/106	19/96	30/89	25/95
Bbox IoU ↑	0.66	0.40	0.24	0.14	0.04	0.36	0.09	0.02
Cen. IoU ↑	0.74	0.55	0.50	0.39	0.24	0.62	0.30	0.17
cost / room ($), median	–	120	13	26	7	9[§]	6[§]	9[§]
wall / room (min), median	–[¶]	132	28	43	38	17	26	23

Cat. and Mod.: fraction of ground-truth objects with the correct category and asset model; $\mathcal{L}_2$: distance between matched box centres (cm); Ori.: rotation error (rad) over objects not excluded as symmetric; Bbox IoU: 3D bounding-box IoU; Cen. IoU: IoU after centre alignment. Agents may place more scored-category objects than a room contains; extra predictions are unpenalised, and the objects-placed row exposes them. ACDC's column is its published sim-to-sim table, averaged here over its four rooms (one run each); the agent columns are this paper's rooms under the same metric definitions, two runs per room, both pooled. Metric rows are means over rooms ($\mathcal{L}_2$ also the median run); cost and wall time are the median run. One Opus 5 room finished within an earlier cap. One Fable 5.1 run is kept with a harness audit flag for reading the benchmark's README, a file outside the lock with no room data (Appendix C). Two GPT-6 Astra runs are kept with a harness audit flag for a reviewer note that names the locked directory to state it was not read (Appendix C). Departures from ACDC's published description are listed in Appendix A. [†]after ACDC's symmetry restriction, which its captions name but do not define. [‡]raw; the next row counts exact quarter- or half-turn flips (fixed tolerance). [§]list-price estimate (runs billed to a subscription). [¶]ACDC publishes per-object timings, no per-room time.

disabled, since the probe script issues its question through the Claude Agent SDK and this setup reaches OpenAI models only through that CLI, and scores zero on every reported room: on seven rooms the model lists no asset, and on room J it lists six pairs none of whose ids is a catalog model. Two runs (room J, r8; room D, r18) are flagged by the harness audit because the reviewer's approval note, written into the workspace with Codex's patch tool, ends with a sentence stating that no file under the locked scene directory was read; the harness exempts the prompt's rule as the agent relays it to a reviewer but not this past-tense statement in a written file, so the flag stood. The hit is the class the audit of the recorded Codex arms also raised on three GPT-5.5 records, where the audit words appear in the rule relayed to a reviewer, in listing commands that exclude the directory and in one workspace edit of `render.py`, none a read of a locked file; the campaign re-audit found no ground-truth access and kept those records with their labels. No shell command in the arm names that directory except as a negated filter in `rg` file listings, the opening listing of 13 of the 16 runs and four later listings in two of them, a form the audit already exempts, and no ground-truth content was returned; both runs are kept and pooled, and without them the arm's column in Table 3 would read 0.71 / 0.64 / 27 (17) / 0.46 / 0.33 / 0.61 for Cat., Mod., $\mathcal{L}_2$, Ori., Bbox IoU and Cen. IoU, with 107 of 131 objects placed over the 14 runs. Two runs (room F, r8; room J, r18) met the lock while listing files with `rg`: the permission-denial lines named the locked directories, 24 in the first run and one in the second, and returned no entry or content from inside any of them; for this arm the lock also covered the main checkout's run records and the pre-migration copies of earlier run records, and no command named them. No run was rerun, none resumed after a rate limit, and every run spawned exactly one reviewer. Both runs of room D are marked by the harness for stream reconnections during the session; each turn completed with exit code 0 and a scored scene file, as on five records of the other Codex arms. Cost is the same list-price estimate as for the other Codex arms, at $10, $1 and $50 per million input, cached-input and output tokens, since the runs were billed to a subscription.

Limits and the one capped run. No turn, wall-clock, or cost cap was imposed on any reported run except one: the earlier Opus 5 run in room C was started under an earlier 400-turn / 90-minute / $30 cap and finished within all three, so it is kept. Two per-tool limits applied throughout: a single shell command times out after 10 minutes, and one SDK message may carry at most 64 MB. On an API rate-limit rejection the harness sleeps until the limit window resets and resumes the same session with "Continue where you left off."; two reported runs (Fable 5, room G; Fable 5.1, room I) took that path, and the wait is excluded from wall time.

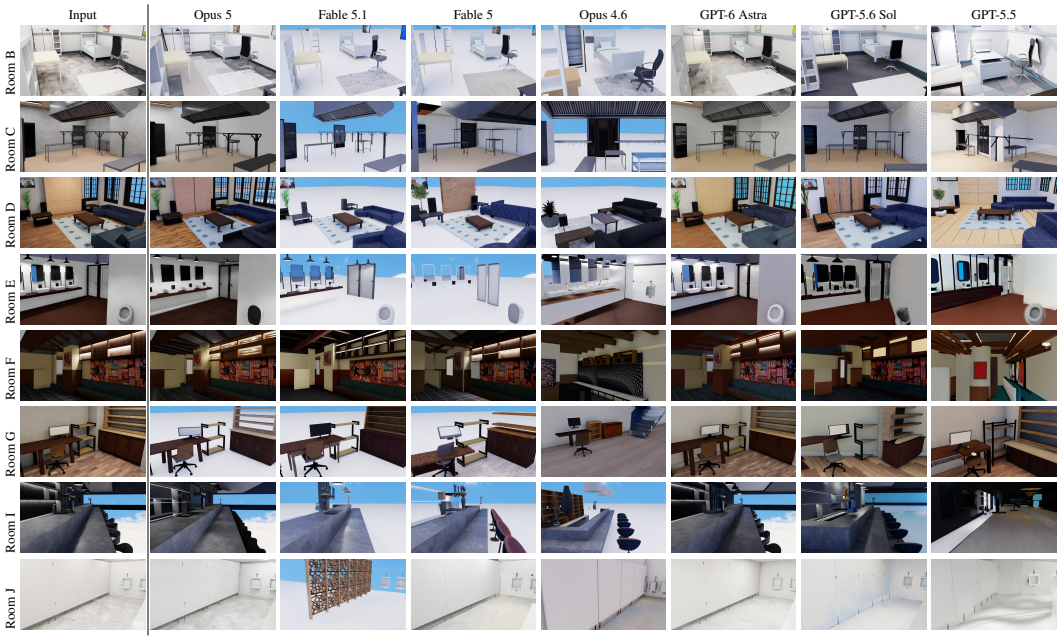

Figure 6: All eight rooms: input photograph and each agent's final render from its first run, from the same fixed camera.

D FARM RECONSTRUCTION TRACES

This appendix summarises working traces from the first eight Opus 5 runs, showing how each of those eight twins' camera, plants and man-made structures were derived from the photograph. Trace excerpts are quoted from the run logs with symbols typeset and wording unchanged.

Campaign prompt. The first eight Opus 5 twins and the 46 GPT-6 Astra twins were set by the following prompt to the top-level agent, with local paths replaced by placeholders (the Astra campaign used the same prompt with the line "Use GPT-6 Astra as subagents with high effort."):

```
Goal: For each image under input scene folder, iterate on scripts to
recreate the digital twin in IsaacSim and render.
How: Spawn a workflow. Generate agent re-generate and re-render until
Reviewer agent approves for general similarity between input scene image and
render image. Use Opus 5 as subagents with medium effort.
Instructions:
- Input scene folder: <input folder>
- Output reconstructed scene folders:
   - Scripts: <scripts folder>
   - Render: <renders folder>
- Store temporary artifacts under: <artifacts folder>
- Reference example scene twin generation:
   - Script: <reference script>
   - Image: <reference image>
   - Reconstructed render in IsaacSim: <reference render>
   - Real-time tracing during iterations; Path tracing for the final render
   - Cap at 3 reviewer rounds
   - Stop whenever similarity is reached within existing assets and
capability, or when new round cannot yield meaningful improvement.
```

Camera estimation. Each farm run receives no camera information, so the agent recovers the viewpoint from the photograph before it can place any object. Table 4 lists the cue each run recorded in its scene script together with the camera it solved, taken from the Opus 5 run that produced that campaign's final render. The solved viewpoints span eye heights of 1.55 to 2.30 m, downward pitches of 2.3 to 46.8 degrees, lens fields of view of 40 to 66 degrees, and both landscape and portrait framings, which covers ground-level alley views, oblique field views, a steep look-down over beds, and a greenhouse interior.

Table 4: Camera solutions recovered by the Opus 5 agent in the first eight runs. The cue column quotes the measurement the run recorded in its scene script. Eye is the solved camera height in metres, pitch is the downward tilt in degrees, and hfov is the horizontal field of view in degrees.

Twin	Primary cue read from the photograph	Eye (m)	Pitch (°)	hfov (°)	Aspect
apple 1	horizon at 0.435 H, rows vanish at 0.34 W	1.60	3.4	66	4:3
apple 2	horizon at 0.455 H, single vanishing point at 0.25 W	1.60	2.3	62	4:3
broccoli 1	horizon at 0.098 H, sky only the top tenth	1.60	17.3	60	3:2
broccoli 2	sea waterline as horizon at 0.336 H	1.55	7.5	62	3:2
lettuce 1	woodland base as horizon at 0.121 H, portrait framing	1.70	20.2	40	3:4
lettuce 2	no horizon in frame, row vanishing point fitted above the frame	2.10	46.8	50	2:3
strawberry 1	horizon at 0.07 H, vanishing point far off frame right	1.55	18.5	62	3:2
strawberry 2	no outdoor horizon, structural horizon fitted from bag rows at 0.27 H	2.30	10.0	60	3:2

The initial estimate comes from explicit projective fits rather than guesswork. In the greenhouse strawberry run the agent wrote *fitting three readable bag rows (image $y \approx 1150/590/490$) to $y = y_h + f\,\Delta z/d$ under a constant bed pitch solves to horizon $y_h \approx$ 247–320 px (0.19–0.25 H); with $f = 1663$ px (hfov 60 deg on 1920 px) that gives camera pitch $\approx$ 10.9 deg down*, then adopted an eye height of 1.67 m. In 6 of the 8 runs the camera stayed fixed from the first round onward and the remaining rounds went to the scene itself.

Two runs corrected the viewpoint in later rounds in response to the reviewer critique. The field strawberry run held eye height, pitch and lens fixed while rotating its yaw off the row axis from 17 to 21.5 and then 30 degrees, after the reviewer measured the near bed rising more steeply across the photograph than in the render. The greenhouse strawberry run raised the camera itself, from 1.67 to 1.72 and finally 2.30 m (Figure 5), and its round-three log states the geometric reason, *the near-pair ratio $v_0/v_1 = 773/593$ forces bed pitch $\approx 0.30\,d_0$, and the condition to see any floor over a bag crown ($eye/(eye - bag_top) \cdot d_0 < d_0 + pitch$) then only holds from a viewpoint well above the bags*.

Plant configuration. The runs place plants pre-grown with the botanical model and fit their scale, spacing, density and colour to the photograph. The first apple run back-projected its most prominent tree, *base at 0.665 H → 7.2 m away, top at 0.305 H → ≈2.5 m tall, subtending ≈2.4 m wide → cached* `apple_bush_*.usdc` *(3.7 m native) scaled 0.62–0.74*. The high-angle lettuce run reported *head sizes measured by unprojection: ≈0.30 m near, ≈0.14 m far* and scaled its rows by depth accordingly, so plant scale falls with depth as the photograph shows. Colour is fitted by measurement rather than by eye. The coastal broccoli run sampled matched patches of photograph and render, *crop-near (53,63,46) vs (73,87,48) → render too bright and blue-starved (B/G 0.55 vs 0.73)*, and retuned its leaf tint against a fresh sample in each subsequent round.

Primitive structures. Man-made structures are composed from colored box, cylinder and mesh primitives at dimensions back-projected from the photograph. The greenhouse run framed its roof as *0.11 m posts on a 2.73 × 4.1 m grid [. . .] all dark weathered timber/steel so it reads as the photo's dark tangle against blown-out white film*, and carried a 0.13 m irrigation main across the frame at the image height the photograph shows. The fleece-field broccoli run laid *two crumpled fleece ribbons ≈0.5 m wide* along its tramlines and closed the horizon with a *white farmhouse with red gable roof* at the far edge of the field. The coastal broccoli run strung a telegraph line with sagging wires over poles receding toward the horizon, and the first apple run placed a small bin at the image position the photograph shows.

E FARM VALIDATION METRIC

Twin fidelity is quantified with two independent pretrained vision encoders, DINOv2 (Oquab et al., 2024) and CLIP (Radford et al., 2021). Each final render and each reference photograph is embedded once per encoder, using the DINOv2 base model and the CLIP ViT-B/32 image tower at pinned checkpoint revisions, and every embedding is L2-normalised. The similarity of a render to a photograph is the cosine between their embeddings. Each twin's render is scored against every reference photograph in the pool, and the rank of the twin's own photograph is recorded, so rank one means the render is closer to its own photograph than to any other reference.

Two calibrations put the cosine values on a natural scale. First, the mean similarity of matched render–photograph pairs is compared with the mean over all mismatched pairs. Second, the reference photographs of the same crop are scored against each other, and the mean over those pairs is the level that real images of the same crop type reach, so a matched-pair mean at or above that level indicates the renders sit at the similarity level of real same-crop photographs. Both encoders are reported because they respond to different cues, and where they disagree the disagreement is reported rather than resolved. Table 5 lists the values for each agent's 46 twins of Section 5, computed on the final path-traced renders, which the Opus 5 campaign produced at 2600 px wide and 2048 samples per pixel and the Astra campaign at 2000 px and 1024 samples per pixel. Under DINOv2 24 of the 46 Astra twins and 22 of the 46 Opus 5 twins rank first, with median ranks of one and two, matched pairs average 0.52 for Astra and 0.57 for Opus 5 against 0.20 for mismatched pairs, and the 87 same-crop photograph pairs average 0.49 with a cross-crop photograph mean of 0.20. Under CLIP 22 of the 46 Astra twins and 25 of the 46 Opus 5 twins rank first, with median ranks of two and one, matched pairs average 0.76 for Astra and 0.80 for Opus 5, against 0.67 and 0.69 in the same order, and same-crop photograph pairs average 0.79 with a cross-crop photograph mean of 0.71, so CLIP separates matched from mismatched pairs by far less than DINOv2 does.

F PHOTOGRAPH CREDITS

The 46 reference photographs are reproduced under the Creative Commons or public-domain terms stated on their source pages, and Table 6 lists the license and source page of each and the author where the source page names one.

G TOOL USAGE OF THE RECONSTRUCTION AGENT

Table 7 counts every tool call in the transcripts of the 16 Opus 5 room runs of Section 3, the two runs per room that Table 3 pools, main agent and sub-agents pooled and split by round: round 1 is the earlier of the two runs of each room and round 2 the later, started after every round-1 run had

Table 5: Render-to-photograph similarity for the 46 farm twins of each agent, Codex with GPT-6 Astra and Claude Code with Opus 5. Cosine is between the twin's final render and its own photograph; rank is the position of the own photograph among the 46 reference photographs (1 = closest).

	DINOv2				CLIP			
	Astra		Opus 5		Astra		Opus 5	
Twin	cosine	rank	cosine	rank	cosine	rank	cosine	rank
apple 1	0.63	1	0.72	1	0.70	4	0.74	3
apple 2	0.68	1	0.60	1	0.82	1	0.79	1
broccoli 1	0.53	3	0.65	1	0.78	7	0.76	4
broccoli 2	0.34	10	0.52	1	0.71	3	0.84	4
lettuce 1	0.76	1	0.86	1	0.82	1	0.88	4
lettuce 2	0.58	3	0.68	2	0.74	1	0.80	1
strawberry 1	0.64	1	0.47	9	0.88	1	0.86	1
strawberry 2	0.47	2	0.49	3	0.85	2	0.79	5
apple 3	0.46	1	0.52	1	0.67	27	0.72	13
apple 4	0.40	1	0.53	1	0.65	4	0.73	5
apple 5	0.40	1	0.49	1	0.77	1	0.73	1
strawberry 3	0.71	1	0.73	1	0.81	2	0.84	2
strawberry 4	0.76	1	0.87	1	0.89	1	0.85	1
strawberry 5	0.49	2	0.69	1	0.73	1	0.86	1
lettuce 3	0.75	1	0.64	1	0.83	2	0.77	2
lettuce 4	0.63	2	0.63	2	0.86	1	0.77	1
lettuce 5	0.62	3	0.66	3	0.91	1	0.92	1
blueberry 1	0.38	9	0.56	3	0.75	12	0.79	2
blueberry 2	0.49	1	0.65	2	0.74	1	0.87	1
blueberry 3	0.55	1	0.56	2	0.72	10	0.75	5
blueberry 4	0.54	1	0.42	2	0.74	9	0.80	2
table grape 1	0.68	1	0.66	1	0.73	4	0.90	1
table grape 2	0.42	5	0.45	3	0.58	26	0.72	6
table grape 3	0.82	1	0.76	1	0.76	3	0.79	2
table grape 4	0.72	1	0.68	1	0.69	13	0.72	1
table grape 5	0.53	4	0.46	5	0.72	1	0.79	1
peach 1	0.35	12	0.35	9	0.72	18	0.64	20
peach 2	0.38	8	0.54	2	0.79	1	0.82	1
peach 3	0.28	10	0.51	2	0.76	1	0.84	2
peach 4	0.38	5	0.55	1	0.73	1	0.87	1
peach 5	0.14	18	0.36	4	0.67	7	0.70	11
sweet cherry 1	0.56	1	0.65	2	0.69	4	0.70	5
sweet cherry 2	0.46	5	0.56	3	0.88	1	0.86	1
sweet cherry 3	0.72	1	0.72	2	0.86	1	0.88	1
sweet cherry 4	0.65	3	0.74	1	0.85	1	0.88	1
sweet cherry 5	0.71	1	0.72	3	0.83	1	0.82	1
tomato 1	0.49	1	0.52	2	0.67	12	0.73	34
tomato 2	0.55	1	0.53	1	0.88	1	0.73	1
tomato 3	0.64	1	0.71	1	0.76	2	0.80	1
tomato 4	0.36	5	0.35	2	0.81	1	0.83	1
tomato 5	0.57	1	0.66	1	0.91	1	0.89	1
sweet potato 1	0.59	2	0.50	1	0.76	16	0.85	5
sweet potato 2	0.16	24	0.26	4	0.68	1	0.84	1
sweet potato 3	0.16	21	0.14	6	0.67	5	0.79	1
sweet potato 4	0.37	1	0.54	2	0.76	9	0.77	1
sweet potato 5	0.55	2	0.59	1	0.67	25	0.72	3
matched mean	0.52		0.57		0.76		0.80	
mismatched mean	0.20		0.20		0.67		0.69	
same-crop photograph pair mean	0.49				0.79			

finished, under the same prompt and model. Definitions are under the table; Section 4 interprets the distribution.

Table 6: Credits for the 46 reference photographs. Titles are the Wikimedia Commons file names for the photographs taken from Commons, and the source column links to the page that states the license.

Twin	Title	Author	License	Source
apple 1	Apple Orchard England.jpg	Shameela Yoosuf Ali	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Apple_Orchard_England.jpg
apple 2	Apfelplantage.JPG	R2Dine (Diskussion)	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Apfelplantage.JPG
strawberry 1	Field of Strawberry Plants.jpg	Alabama Extension	CC0	https://commons.wikimedia.org/wiki/File:Field_of_Strawberry_Plants.jpg
strawberry 2	Fungiculture cameron erdbeeren 1.jpg	Arne Müseler	CC BY-SA 3.0 de	https://commons.wikimedia.org/wiki/File:Fungiculture_cameron_erdbeeren_1.jpg
lettuce 1	Weeding Lettuce (4858251381).jpg	Dwight Sipler from Stow, MA, USA	CC BY 2.0	https://commons.wikimedia.org/wiki/File:Weeding_Lettuce_(4858251381).jpg
lettuce 2	Cultivating the lettuce with the new tractor (8069104279).jpg	Dwight Sipler from Stow, MA, USA	CC BY 2.0	https://commons.wikimedia.org/wiki/File:Cultivating_the_lettuce_with_the_new_tractor_(8069104279).jpg
broccoli 1	Brokkoli Ernteverfrühung-1-Josef Schlaghecken.jpg	Schlaghecken Josef	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Brokkoli_Ernteverfr%C3%BChung-1-Josef_Schlaghecken.jpg
broccoli 2	Broccoli beside Black Road - geograph.org.uk - 7890281.jpg	Richard Sutcliffe	CC BY-SA 2.0	https://commons.wikimedia.org/wiki/File:Broccoli_beside_Black_Road_-_geograph.org.uk_-_7890281.jpg
apple 3		mikecogh	CC BY-SA 2.0	https://www.flickr.com/photos/89165847@N00/52227889707
apple 4		artq55	CC BY-SA 2.0	https://www.flickr.com/photos/34388067@N00/51515724940
apple 5		muffinn	CC BY 2.0	https://www.flickr.com/photos/26445715@N00/51489397640
strawberry 3		Michele Dorsey Walfred	CC BY 2.0	https://www.flickr.com/photos/97485958@N02/52381884860
strawberry 4		kewl	CC BY 2.0	https://www.flickr.com/photos/58411470@N00/53426154277
strawberry 5		Aleksandr Zykov	CC BY-SA 2.0	https://www.flickr.com/photos/84292292@N00/52184909454
lettuce 3		USDAgov	Public Domain Mark	https://www.flickr.com/photos/41284017@N08/52490807311
lettuce 4		nrcs-pia-enews	Public Domain Mark	https://www.flickr.com/photos/194676499@N06/51830110202

Table 6: Credits for the 46 reference photographs, continued.

Twin	Title	Author	License	Source
lettuce 5			CC0	https://www.rawpixel.com/image/5912933/image-public-domain-leaf-plants
blueberry 1		NRCS Oregon	CC BY 2.0	https://www.flickr.com/photos/55204730@N08/52748945037
blueberry 2		jdnx	CC BY 2.0	https://www.flickr.com/photos/21442511@N08/54277333342
blueberry 3		trekkyandy	CC BY-SA 2.0	https://www.flickr.com/photos/87054972@N00/51826854168
blueberry 4	Mount Ephraim Kedumim Neighborhood 16.jpg	Daniel Ventura	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Mount_Ephraim_Kedumim_Neighborhood_16.jpg
table grape 1		Hervé S, France	CC BY-SA 2.0	https://www.flickr.com/photos/125601701@N03/52610481863
table grape 2		kuhnmi	CC BY 2.0	https://www.flickr.com/photos/31176607@N05/51836678920
table grape 3		Ivan Radic	CC BY 2.0	https://www.flickr.com/photos/26344495@N05/51380492763
table grape 4		Ruben Holthuijsen	CC BY 2.0	https://www.flickr.com/photos/60057960@N06/52252622254
table grape 5		photobeppus	CC BY-SA 2.0	https://www.flickr.com/photos/65517846@N03/23342400454
peach 1		National Interagency Fire Center	Public Domain Mark	https://www.flickr.com/photos/165415395@N07/53452275862
peach 2		Alabama Extension	CC0	https://www.flickr.com/photos/184594136@N08/53831593920
peach 3	04.12.2022 Prefeita Paula Mascarenhas e Vice-Prefeito Idemar Barz participam da Abertura da Festa Oficial do Pêssego Foto Rodrigo Chagas (52543506100).jpg	Prefeitura de Pelotas	Public domain	https://commons.wikimedia.org/wiki/File:04.12.2022_Prefeita_Paula_Mascarenhas_e_Vice-Prefeito_Idemar_Barz_participam_da_Abertura_da_Festa_Oficial_do_P%C3%AAssego_Foto_Rodrigo_Chagas_(52543506100).jpg
peach 4	Autumn (4424038505) (2).jpg	Milan Nykodym from Kutna Hora, Czech Republic	CC BY-SA 2.0	https://commons.wikimedia.org/wiki/File:Autumn_(4424038505)_(2).jpg
peach 5	Starr-140614-4505-Prunus persica var persica-Tropic Beauty habit-Hawea Pl Olinda-Maui (24612409084).jpg	Forest and Kim Starr	CC BY 3.0 us	https://commons.wikimedia.org/wiki/File:Starr-140614-4505-Prunus_persica_var_persica-Tropic_Beauty_habit-Hawea_Pl_Olinda-Maui_(24612409084).jpg

Table 6: Credits for the 46 reference photographs, continued.

Twin	Title	Author	License	Source
sweet cherry 1		Ion Chibzii	CC BY-SA 2.0	https://www.flickr.com/photos/55635450@N04/51387344912
sweet cherry 2	Cherry orchard in Bemmel.jpg	Lendskaip	CC0	https://commons.wikimedia.org/wiki/File:Cherry_orchard_in_Bemmel.jpg
sweet cherry 3	Covered with Blossoms (33839893091).jpg	Johan Neven from Borgloon-Gotem, Belgium	CC BY 2.0	https://commons.wikimedia.org/wiki/File:Covered_with_Blossoms_(33839893091).jpg
sweet cherry 4	Old tree 2, cherry orchards, Frauenstein.jpg	Gerda Arendt	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Old_tree_2,_cherry_orchards,_Frauenstein.jpg
sweet cherry 5	Cireşi înfloriţi lângă Chişinău.jpg	Tolea93	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Cire%C8%99i_%C3%AEnflori%C8%9Bi_l%C3%A2ng%C4%83_Chi%C8%99in%C4%83u.jpg
tomato 1		USAID Mozambique	Public Domain Mark	https://www.flickr.com/photos/62256332@N08/54339875114
tomato 2	NC Greenhouse Nursery Tomatoes in Aquaponics.jpg	Gogerr	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:NC_Greenhouse_Nursery_Tomatoes_in_Aquaponics.jpg
tomato 3	Test Label rouge tomates1.jpg	Arn	CC BY 4.0	https://commons.wikimedia.org/wiki/File:Test_Label_rouge_tomates1.jpg
tomato 4	Tomaten - Biogärtnerei Auenhof (36587140795).jpg	Maja Dumat from Deutschland (Germany)	CC BY 2.0	https://commons.wikimedia.org/wiki/File:Tomaten_-_Biog%C3%A4rtnerei_Auenhof_(36587140795).jpg
tomato 5	Tomato - . . . 02.JPG	Shijan Kaakkara	CC BY-SA 3.0	https://commons.wikimedia.org/wiki/File:Tomato_-_%E0%B4%A4%E0%B4%95%E0%B5%8D%E0%B4%95%E0%B4%BE%E0%B4%B3%E0%B4%BF_02.JPG
sweet potato 1	0581jfLandscapes Roads Vegetables Fields Binagbag Angat Bulacanfvf 22.JPG	Judgefloro	Public domain	https://commons.wikimedia.org/wiki/File:0581jfLandscapes_Roads_Vegetables_Fields_Binagbag_Angat_Bulacanfvf_22.JPG
sweet potato 2	Matembele.jpg	Max.kit	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Matembele.jpg
sweet potato 3	Plante de patates douces.jpg	Frankyta	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Plante_de_patates_douces.jpg
sweet potato 4	Potato plant in Jalingo garden.jpg	Abumuslimjalingo	CC0	https://commons.wikimedia.org/wiki/File:Potato_plant_in_Jalingo_garden.jpg
sweet potato 5	Sweet potato garden.jpg	DidoLumanyika	CC BY-SA 4.0	https://commons.wikimedia.org/wiki/File:Sweet_potato_garden.jpg

Table 7: Tool usage of the Opus 5 reconstruction agent: 7,690 tool calls over 16 runs (8 rooms × 2 rounds)

	Runs	Calls	Bash	Read	ListAgents	Agent	Other	Turns†	Spawns†	Renders†
Round 1	8	4,055	66.8%	30.4%	1.7%	0.7%	0.4%	362.8	3.4	61.6
Round 2	8	3,635	63.1%	35.0%	0.3%	1.0%	0.6%	303.5	4.4	51.2
All	16	7,690	65.0%	32.6%	1.1%	0.8%	0.5%	333.1	3.9	56.4

A call is one tool-use block in a run's transcript, main agent and sub-agents pooled. Agent = sub-agent spawns; all 62 carry review-shaped descriptions (e.g. "Review twin similarity"). Other = SendMessage (14), ToolSearch (11), Monitor (9), Write (3), Edit (1), TaskStop (1): the agent writes files through shell heredocs, so Write and Edit total 4 calls in 16 runs, and Grep, Glob and web tools are never called. Of round 1's 69 ListAgents calls, 47 come from one run polling its reviewer. †Per-run means: turns is the harness's own count from the run's result record; spawns are Agent calls; renders are simulator launches inside Bash commands.