4DCODEBENCH: BENCHMARKING AGENTS ON INVERSE GRAPHICS OF DYNAMIC SCENES

Anonymous authors
Paper under double-blind review

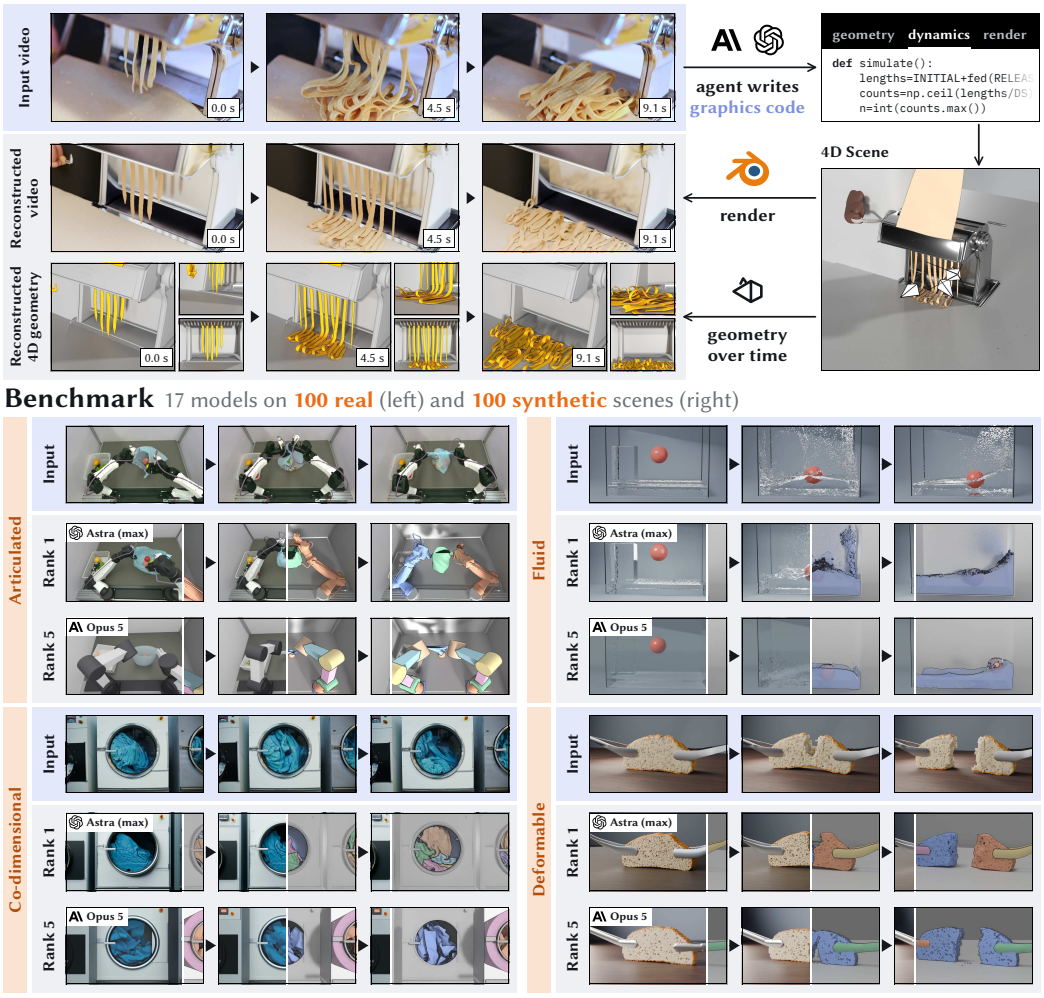

Figure 1: **4DCODEBENCH. Top:** the task. Given an input video, an agent must reconstruct it as an explicit 4D scene by writing graphics code; the scene is rendered back to video and its geometry is inspected over time. **Bottom:** we benchmark 17 models on 100 real and 100 synthetic scenes. Each example shows the input and two model reconstructions, split into render (left) and geometry (right).

ABSTRACT

We introduce 4DCODEBENCH, a benchmark for 4D inverse graphics through code generation, in which agents reconstruct dynamic scenes from video as executable graphics programs. To accomplish this, agents must translate video observations into compact representations of scene structure and dynamics by implementing abstractions such as physical simulations to reproduce complex behavior. To evaluate this capability, we curate a set of real-world videos and construct synthetic scenes

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.

spanning diverse physical phenomena, including deformation, fluid flow, and fracture. We perform extensive benchmarking of frontier models, finding that strong static reconstruction capabilities do not yet translate into reliable reconstruction of complex dynamics. 4DCODEBENCH provides a testbed for tracking progress toward agents that can interpret the dynamics of the world through code. Anonymous project page: https://anon-4d-code-bench-iclr27.site/.

1 INTRODUCTION

Inverse graphics is often approached by fitting a prescribed model to visual observations. Recovering an executable graphics program moves more of the modeling into the inference problem. The program specifies what geometry to construct and, in dynamic scenes, how to represent its motion.

Dynamic scenes render these modeling choices more consequential. Flow, fracture, and deformation involve changes in shape and often topology, while material interactions affect each others' motion. Representing such extends beyond the placement of objects or recovery from a moving camera in a static scene. We ask how well multimodal coding agents can reconstruct the dynamics of interacting objects and materials from video, choosing and implementing their own representations.

We introduce 4DCODEBENCH, to evaluate this capability through 4D inverse graphics. Agents are given a reference video and tasked with writing code that generates the 3D geometry of the scene over time, and renders the result. We leave implementation of these dynamics open rather than requiring the use of a particular method. This freedom renders the choices made, and analysis of the generated code, part of the investigation.

Our benchmark includes 100 real-world videos and 100 synthetic scenes that span diverse materials and interactions. We separately evaluate appearance, geometry, and dynamics of the scene, comparing against reference videos and 3D ground truth on synthetic scenes.

Though analytic motion is the most-common strategy, analysis of the generated programs highlights substantial differences in implementation across models. Through evaluation of 17 frontier models, we observe that models' reconstruction of dynamics consistently lags behind that of appearance and static geometry.

In total, we contribute the following:

1. **A benchmark and dataset for 4D inverse graphics through code.** We introduce 4DCODEBENCH, comprising 200 dynamic scenes—100 real videos and 100 synthetic scenes with 4D ground truth—spanning diverse materials, interactions, and physical phenomena. Agents must reconstruct each video as an executable graphics program that exposes explicit geometry and dynamics over time.
2. **A comprehensive evaluation of frontier and open-weight models.** We benchmark 17 multimodal coding models under a common execution environment and evaluation protocol, measuring visual fidelity, 2D and 3D geometry, and dynamics. Our analysis identifies substantial differences across models and shows that reconstructing dynamics remains a consistent weakness even when appearance and static geometry are recovered well.
3. **Human-aligned evaluation and analysis of reconstruction failures.** We conduct a large-scale human preference study and develop automated VLM- and reconstruction-based metrics that closely track human judgments. Using these measurements, we analyze which scene properties and physical regimes expose different reconstruction failures, providing a diagnostic view of current model capabilities.

2 4DCODEBENCH

4DCODEBENCH comprises of 200 scenes: 100 real-world videos curated from the internet and 100 synthetic scenes generated using physics simulations. The two sources serve complementary roles. Real-world videos capture realistic appearance and complex dynamics without the simplifying physical assumptions of a simulator, but lack 4D ground truth. Synthetic scenes depend on these assumptions but provide ground-truth geometry and motion at every frame, enabling direct evaluation

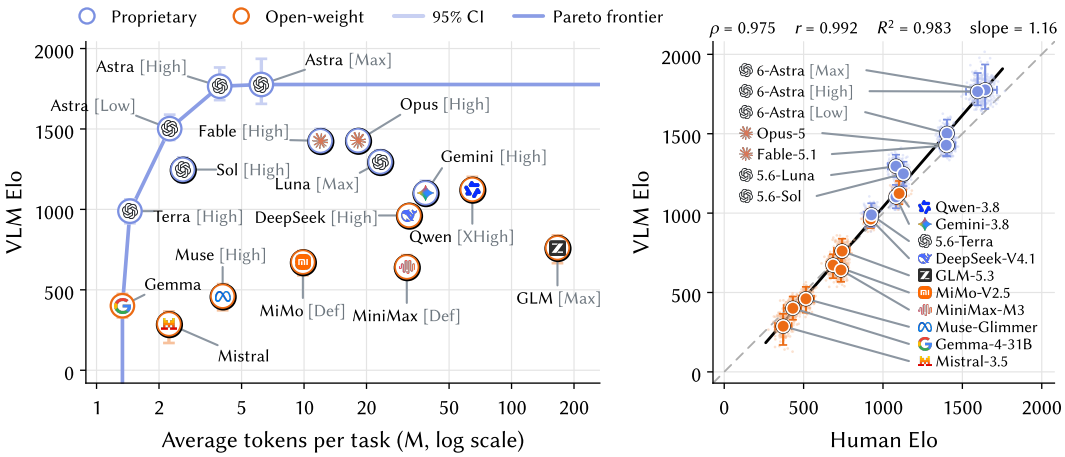

Figure 2: **Cost–quality trade-off and human–VLM agreement.** The left panel plots each model's VLM Elo against its average token use per task, with the Pareto frontier; the right panel plots VLM Elo against human Elo for all 17 models. Error bars show 95% intervals.

in 3D and 4D space, rather than solely comparing rendered frames in image space. Below we describe the task definition (§2.1), benchmark construction(§2.2), data statistics (§2.3), and evaluation metrics (§2.4).

2.1 Task Definition: 4D Inverse Graphics through Code Generation

Inverse graphics recovers the underlying representation of a scene from observations. We pose *4D inverse graphics* as code generation: given a video, an agent writes an executable program, which we call *4D code*, that reconstructs the scene's 3D geometry and its evolution over time.

Formally, given a reference video $x_{\text{ref}} = (x_0, \ldots, x_{T-1})$ of a physical scene, an agent must produce executable graphics code $\Gamma = (\gamma_1, \ldots, \gamma_m)$, for 4D inverse graphics, i.e. reconstructing the scene's 3D geometry and describing its evolution over time. We impose no restrictions on the number, organization, or roles of the individual programs γ_i. Executing Γ must expose a geometric state $s_\Gamma(t)$ at each evaluated timestep and render that state in Blender (Blender Foundation, 2026) through a rendering procedure $\mathcal{R}_\Gamma$:

$$\Gamma = \mathcal{A}(x_{\text{ref}}), \qquad \hat{x}_k = \mathcal{R}_\Gamma\left(s_\Gamma(k/r)\right), \quad k = 0, \ldots, T-1, \tag{1}$$

where $\mathcal{A}(\cdot)$ denotes the agent and r is the reference frame rate. $\mathcal{R}_\Gamma$ renders the geometric state using the camera, lighting, and materials specified by Γ. The rendered output must match the reference's resolution, frame count, and frame rate, and Γ must execute without access to x_{ref}. We render in Blender (Blender Foundation, 2026), but the code implementation is simulator-agnostic: dynamics may be keyframed or simulated with any tool, *e.g.,* Blender, Taichi (Hu et al., 2021), or Warp (Macklin, 2022). Figure A1 gives an intuitive overview of our task pipeline, which supports diverse agents and videos.

2.2 Benchmark Overview and Construction

To construct the benchmark, we collect real-world videos and generate synthetic scenes covering a wide range of physical dynamics. These include scenes of robotic manipulation, controlled physics demonstrations, and everyday activities. We organize scenes into four primary categories: rigid and articulated bodies, deformable solids, co-dimensional structures such as cloth and rope, and flowing materials such as grains and fluids. Across these categories, scenes exhibit phenomena including collisions, deformation, flow, and fracture. We further characterize scenes by object count, material composition, and dynamics regime, distinguishing passive dynamics from externally driven motion, such as robotic manipulation or prescribed motion of rigid objects.

Real-world scenes. We curate videos from existing physical-understanding, video-generation, and robotics datasets, supplemented with web-sourced footage. We select clips with diverse physical

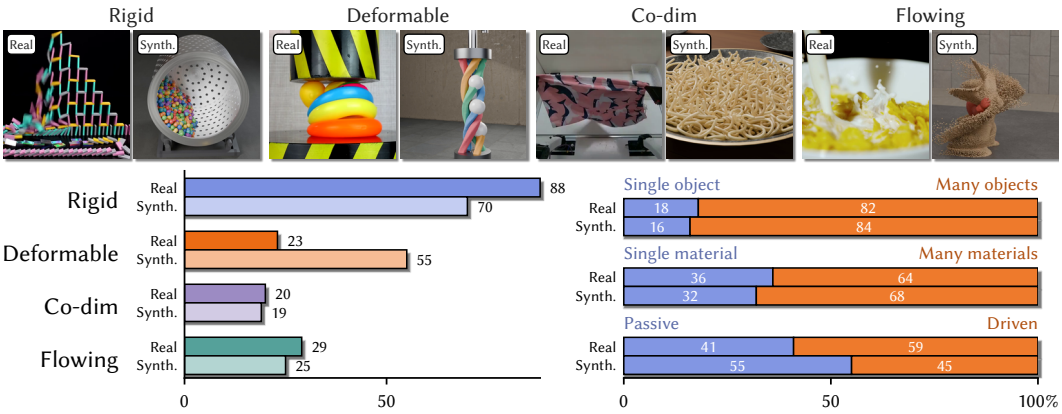

Figure 3: **Dataset examples and statistics.** The top row illustrates each matter family with an example scene, shown by a reference frame beside its ground-truth normals and depth. The bottom panels count the real and synthetic scenes that contain each family (left) and break both splits down by the number of dynamic objects, the number of materials, and whether the motion is passive or driven (right).

behaviors while minimizing repeated scenarios, prioritizing stationary cameras, limited occlusion, and continuous footage without cuts or editing-induced discontinuities. We also exclude humans and animals and minimize visible hands, since reconstructing their detailed geometry and articulation introduces challenges beyond our focus on physical dynamics. Each clip is manually reviewed for visual quality and temporal continuity. Appendix C.2 lists the video sources, licenses, and number of scenes from each source.

Synthetic scenes. We construct synthetic scenes using existing physical simulators, extending their implementations where needed to support additional material models. Scenes are designed by the authors, combining procedurally generated geometry with publicly available meshes such as the Stanford bunny and armadillo. We select simulators suited to each scene's desired materials and interactions and use them to compute the objects' geometry and motion over time. We design diverse scenarios that vary geometry, material types and parameters, initial conditions, and rendering settings to create varied reconstruction tasks. Authors with experience in physical simulation developed and reviewed each scene for numerical instability, unintended interpenetration, and other visible simulation artifacts. Appendix C.3, Table A3 lists the simulators used and provides a breakdown of scenes by simulator.

2.3 Dataset Statistics

Figure 3 summarizes the distribution of scene properties in 4DCodeBench. We characterize each scene by the physical behavior of its primary dynamic objects, together with properties such as the number of dynamic objects, the presence of multiple material types, and whether the motion is passive or externally driven.

Scene ontology. We organize scenes hierarchically by the matter of their main dynamic objects. We distinguish four broad classes: *rigid and articulated*, including rigid bodies and articulated systems; *deformable solids*, including hyperelastic, viscoelastic, and elastoplastic materials; *co-dimensional* structures, including shells and rods; and *flowing* materials, including granular media, Newtonian fluids, and viscoplastic fluids. Scenes may contain multiple matter types interacting with each other. The full breakdown is visualized in Appendix C

Beyond matter type, we characterize scenes by the number of dynamic objects and by their interactions. **83%** of scenes contain multiple dynamic objects, and **66%** contain multiple matter types. These include interactions such as rigid−deformable, rigid−fluid, fluid−deformable, deformable−co−dimensional, and rigid−co−dimensional coupling. We also distinguish passive dynamics, where motion follows from the initial state and subsequent physical interactions, from externally driven dynamics, such as robotic manipulation or prescribed motion of an object. The distribution of these properties across our scenes, together with the breakdown across the real and synthetic splits, is shown in Figure 3.

All videos are capped at 300 frames and a maximum frame rate of 60 `fps` during dataset construction. Videos with a higher source frame rate are temporally subsampled to at most 60 `fps` while videos recorded at lower frame rates retain their original frame rate. Figure A2 reports the final distributions of video length and frame rate.

2.4 Metrics and Evaluation: What is a good reconstruction?

A faithful 4D reconstruction must reproduce both the visual appearance and the physical dynamics of the observed scene. Because ground-truth availability is asymmetric—synthetic scenes provide full 4D world states, whereas real videos provide only monocular views—our evaluation spans five metric families: **Perceptual**, **2D Dynamics**, and **2.5D Geometry** measured against the reference video, alongside **3D Geometry** and **3D Dynamics** evaluated against the reference world. The **Overall** score averages these five families; failed runs receive the worst score on the affected metrics (Appendix D.1).

Video-reference evaluations. On all scenes, Perceptual measures visual and semantic fidelity to the reference video using DINOv3 (Siméoni et al., 2025) similarity, and 2D Dynamics measures the coverage of moving objects (*Dynamic IoU*). On real videos, we additionally estimate optical flow, point trajectories, and depth from the reference video with off-the-shelf vision models; 2D Dynamics compares the flow (*Flow*) and trajectories (*Track2D*), and 2.5D Geometry compares the depth (*Depth error*). On the reconstruction side, these quantities are computed analytically from the 4D world rather than estimated from rendered pixels (Appendices D.2 and D.3).

VLM-based VQA and pairwise evaluation. We further evaluate reconstructions with a vision-language model (VLM)-as-judge. VQA asks scene-specific questions in four categories: initial state, final state, key events, and contact, gated by the presence of the scene's main object (Figure 4; Appendix D.5). Pairwise evaluation presents the reference video together with two reconstructions and asks which more closely matches the reference in geometry and dynamics. Pairwise preferences are aggregated across scenes into a model-level Elo rating using a Bradley–Terry model (Appendix D.6).

4D-reference evaluations. On synthetic scenes, after aligning the reconstruction to the reference (Appendix D.4), 3D Geometry measures static surface accuracy at frame 0 via Chamfer distance (*Scene 3D*). 3D Dynamics compares motion from two complementary views: a Lagrangian perspective tracking matter along persistent 3D paths (*Trajectory DTW*), and an Eulerian perspective comparing frame-to-frame displacement distributions without particle correspondence (*EMD step*).

Geometry diagnostics. We also check each delivered 4D world for structural defects: open boundaries, non-manifold edges, degenerate faces, self-intersections, and object interpenetrations (Figure A11). Because these checks assess mesh validity rather than fidelity to the reference, they are reported as diagnostics outside the Overall score (Appendix D.8).

3 Experiments

3.1 Experimental Setup

We evaluate **17** multimodal coding models, covering both proprietary models (GPT-6 Astra [Max, High, Low], Claude Fable 5.1 [High], Claude Opus 5 [High], GPT-5.6 Sol [High], GPT-5.6 Luna [Max], GPT-5.6 Terra [High], Gemini 3.8 Flash [High]) and open-weight models (Qwen3.8 Flash [XHigh], DeepSeek v4.1 Flash [High], GLM 5.3 Flash [Max], MiniMax M3 [Def], MiMo v2.5 [Def], Muse Glimmer [High], Gemma-4 31B, Mistral Medium 3.5).[1] We evaluate multiple reasoning-effort settings of Astra to study the effect of inference-time reasoning on reconstruction performance. We run each model once on each of the 200 benchmark scenes.

Each benchmarked agent takes as input only the reference RGB video and a fixed task prompt describing the reconstruction task and required output format. We do not provide scene names, semantic descriptions, object lists, or any other information about the scene; agents must infer these from the input video. The complete prompt is provided in Appendix G.2.

[1] We also evaluated Kimi K3 but omit it from the reported results: its unusually long agent trajectories cost about 50% more than Fable, without a commensurate gain in reconstruction quality.

Proprietary models run in their provider's native agent CLI, and open-weight models in the open Stirrup harness (Artificial Analysis, 2025). Each run executes in an isolated container with one GPU, an empty workspace, and read-only access to the reference video and task files. The environment provides Blender, common numerical and simulation libraries, and an offline copy of the Blender API documentation, so agents may implement dynamics with Blender, the provided libraries, or their own code. Appendix G.1 details the environment.

Each agent must return an executable program together with a reconstructed world containing the rendered video, camera parameters, per-frame geometry, and temporal correspondence for dynamic matter. The program must regenerate the reconstruction from scratch without accessing the reference video. We describe the exact file structure, coordinate conventions, and validation requirements in Appendix G.3. Note, we do not restrict how dynamics are implemented, for example, animation and physical simulation are two possible approaches. Dynamics may be implemented in Blender, using simulation libraries such as Taichi (Hu et al., 2019; 2020; 2021) or Warp Macklin (2022), or with custom code. The program must also expose its geometric state in each evaluated frame, allowing evaluation of both rendered appearance and, for synthetic scenes, geometry and motion against 4D ground truth.

Each model is evaluated on all 200 benchmark scenes. If a run fails to produce a valid rendered video or world representation, metrics requiring the missing output are assigned their worst value (Section 2.4). Submission success rates are reported in Appendix D.1.

3.2 Human Evaluation

We run a pairwise human preference study on all 200 scenes to measure perceived reconstruction quality and to validate the automated metrics. Each trial shows the reference video, then the renders of two models side by side in random order; participants pick the one closer to the reference in geometry, layout, and dynamics, or mark them *about the same* (10.5% of responses). Model pairs are sampled adaptively to shrink the widest confidence intervals (Chiang et al., 2024), and judgments are aggregated into Elo ratings with a Bradley–Terry model (Bradley & Terry, 1952), as for the VLM judge. We collect 3,587 judgments from 76 participants; 209 comparisons are judged by several participants and 916 also by the VLM, which lets us measure inter-rater and human–VLM agreement (Section 4.1; Appendix E).

4 Results and Ablations

4.1 Overall Performance and Human Alignment

We evaluate 17 models on 200 scenes (Figure 5). Astra [Max] leads the Overall ranking, followed by Astra [High], Fable [High], Astra [Low], and Opus [High]. Among the models evaluated, open-weight models generally trail behind proprietary models. The leaderboard distinguishes substantial differences in reconstruction quality, including a wide range among open-weight models. Failed submissions receive the worst applicable metric scores; 89.6% of submissions are fully executable (Appendix Figure A7). Figure 6, 1 shows reconstruction examples across several models. More sample results are included in Appendix F and on the website.

Human judgments support this ranking. Across 3,587 pairwise judgments from 76 participants, human and VLM Elo correlate very strongly across models (Spearman $\rho = 0.975$; Figure 2). We additionally measure agreement on the shared set of individual comparisons. On 916 shared comparisons, human and VLM judgments agree in 89.3% of cases ($\kappa = 0.761$), close to the 92.1% agreement between human annotators on repeated comparisons ($\kappa = 0.816$). Human Elo also correlates with the Overall score ($\rho = 0.96$). Individual VLM judgments are less reliable for closely matched models: agreement is near chance within 50 Elo points, reaches 75% at a 132-point separation, and reaches 90% at 343 points.

VQA provides a complementary assessment of individual reconstructions (Figure 4). Astra [Max] answers 87.6% of questions correctly, compared with 78.8% for Fable [High] and 29.6% for GLM [Max]. These averages also conceal large differences across scenes: Astra [Max] answers every question on 59% of scenes, whereas Mistral answers none on 94% (Appendix Figure A16).

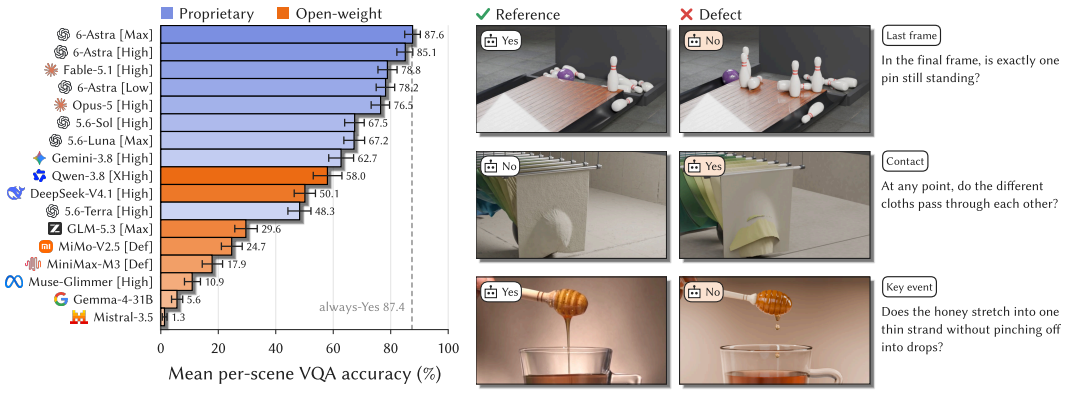

Figure 4: **Visual question answering (VQA).** The left panel reports the proportion of questions each model answers correctly; the dashed line is the accuracy of always answering Yes. The right panel illustrates the protocol with example questions, each asked on the reference video and on an edited render that breaks the queried property.

Figure 2 (left) shows that greater token use across different models does not consistently yield higher Elo. The same pattern appears in the running statistics: token volume and agent steps have little correlation with Overall score across models (Appendix Figure A21).

Together, the agreement with human preferences supports using the automated evaluation suite to compare new models on 4DCODEBENCH, while the separate scores identify where their reconstructions succeed or fail.

4.2 PERFORMANCE ACROSS METRIC FAMILIES

The leaderboard reveals a consistent weakness in reconstructing motion (Figure 5). Across all 17 models, Perceptual and 3D Geometry scores are higher on average than 2D and 3D Dynamics scores (Figure 8). The gap remains when appearance is compared with 2D motion and 3D geometry with 3D motion separately (Appendix Figure A20). Even the strongest model, Astra [Max], scores 0.91 on the static families versus 0.67 on the dynamic families. Reconstructing a scene's evolution remains harder than recovering its appearance and geometry.

The metrics give broadly similar rankings across models (Figure 5), but agree less when ranking reconstructions of the same scene (Appendix Figure A9). They therefore expose differences in how well each model reconstructs appearance, geometry, and motion on individual scenes. VQA shows a related temporal distinction: first-frame questions are generally answered more reliably than questions about the final frame and key events (Appendix Figure A18). model reconstructs appearance, geometry, and motion on individual scenes.

Five additional reconstruction metrics broadly agree with the leaderboard, whereas mesh-quality checks do not: simple but inaccurate geometry can score well on watertightness and related measures (Appendix Figure A10). We therefore report mesh-quality checks separately as diagnostics of structural defects.

4.3 PERFORMANCE ACROSS SCENE ONTOLOGIES

We next examine how reconstruction quality varies with the scene ontology of Section 2. Figure 7 reports standardized differences across scene categories. Relative to synthetic scenes, real scenes have lower VQA and Perceptual scores (-0.73 and -0.85). Scenes containing multiple matter types show a similar pattern relative to single-matter scenes (-0.55 and -0.38). Thus, both the source of a scene and its material composition particularly affect its recognizable appearance and events.

Other categories expose different weaknesses. Co-dimensional scenes have lower 2D Dynamics and 2.5D Geometry scores (-0.35 and -0.76), while driven scenes have lower VQA but higher 3D Geometry scores (-0.30 and $+0.47$). The highlighted effects recur across most individual models, rather than arising only from pooled averages. Flowing scenes also show lower 3D Dynamics (-0.22),

	Human Elo	VLM Elo	VQA	Perceptual	2D Dynamics	2.5D Geometry	3D Geometry	3D Dynamics	Overall
6-Astra [Max]	1634	1766	0.88	0.91	0.60	0.79	0.92	0.73	**0.79**
6-Astra [High]	1586	1756	0.85	0.90	0.55	0.77	0.92	0.69	**0.77**
Fable-5.1 [High]	1394	1416	0.79	0.86	0.52	0.74	0.90	0.68	**0.74**
6-Astra [Low]	1393	1492	0.78	0.88	0.48	0.75	0.90	0.63	**0.73**
Opus-5 [High]	1389	1418	0.76	0.85	0.51	0.73	0.86	0.64	**0.72**
5.6-Luna [Max]	1072	1285	0.67	0.83	0.42	0.70	0.86	0.53	**0.67**
5.6-Sol [High]	1119	1238	0.67	0.83	0.36	0.63	0.85	0.56	**0.65**
Gemini-3.8 [High]	1072	1092	0.63	0.81	0.36	0.63	0.84	0.56	**0.64**
DeepSeek-V4.1 [High]	912	952	0.50	0.78	0.34	0.62	0.77	0.48	**0.60**
Qwen-3.8 [XHigh]	1092	1114	0.58	0.71	0.34	0.55	0.71	0.50	**0.56**
5.6-Terra [High]	917	979	0.48	0.79	0.23	0.52	0.77	0.43	**0.55**
GLM-5.3 [Max]	733	751	0.30	0.69	0.20	0.51	0.60	0.36	**0.47**
Muse-Glimmer [High]	505	450	0.11	0.61	0.11	0.47	0.68	0.33	**0.44**
MiMo-V2.5 [Def]	676	662	0.25	0.69	0.12	0.48	0.57	0.31	**0.43**
MiniMax-M3 [Def]	725	632	0.18	0.52	0.11	0.35	0.44	0.26	**0.34**
Mistral-3.5	361	278	0.01	0.46	0.02	0.41	0.42	0.26	**0.32**
Gemma-4-31B	422	391	0.06	0.51	0.00	0.13	0.17	0.13	**0.19**

Figure 5: **Leaderboard of 17 models on 200 scenes.** Each row lists the human and VLM Elo, the six family scores in $[0, 1]$ (higher is better), and Overall, the mean of Perceptual, 2D Dynamics, 2.5D Geometry, 3D Geometry, and 3D Dynamics; rows are sorted by Overall.

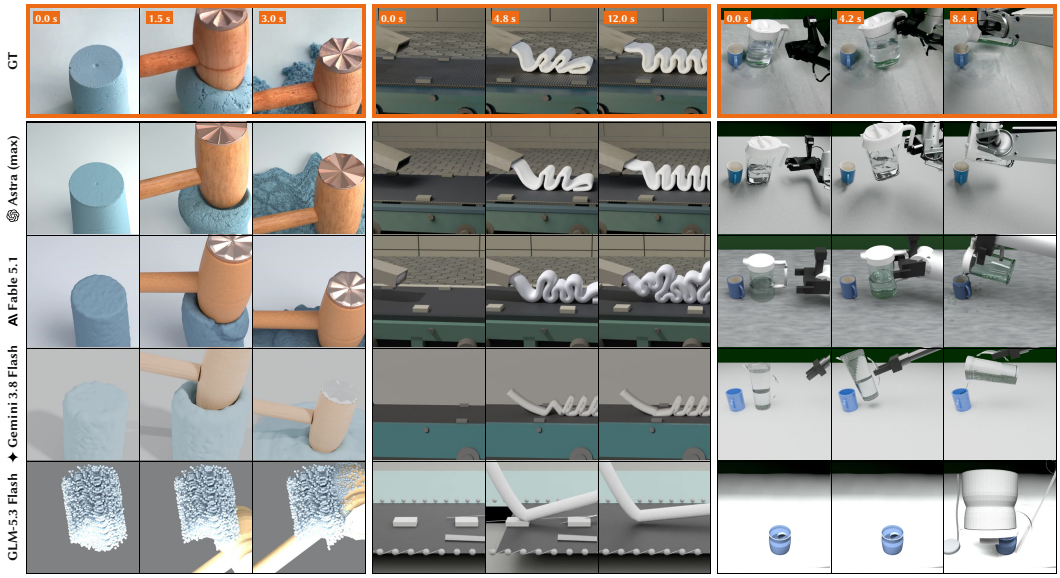

Figure 6: **Qualitative comparison across models.** Each row shows one model's reconstruction (top: ground truth; then GPT-6 Astra [Max], Claude Fable 5.1, Gemini 3.8 Flash, and GLM 5.3 Flash). Stronger models recover both better geometry and dynamics.

although this effect does not meet the figure's consistency criterion. Together, these patterns show why scene categories must be considered alongside aggregate model scores

4.4 How Do Agents Reconstruct Dynamics?

The executable submissions let us examine how agents implement motion. Table A4 classifies their solutions as analytic motion, custom simulation, Blender physics, or keyframing. Overall, 71% of solutions use analytic motion, 16% custom simulation, 9% Blender physics, and 3% keyframing. The distribution varies substantially across models: Opus [High] uses custom simulation in 51% of its solutions, whereas Astra [Low] uses analytic motion in 85% and Terra [High] in 100%. Models therefore use different representations of dynamics even when solving the same reconstruction task. Appendix F.3 describes what each category contains in more detail.

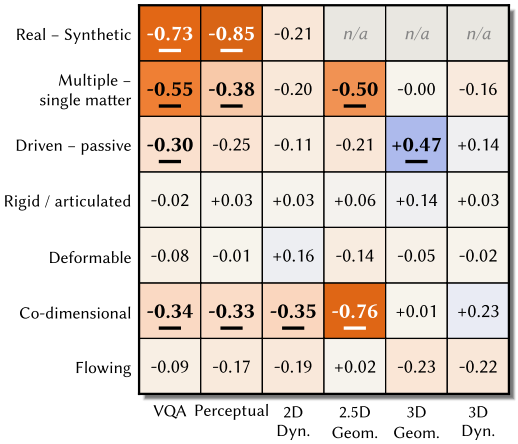

Figure 7: **Category effects.** Each cell is the standardised change in a family score on one category of scenes, measured against its counterpart or, for a matter family, against all scenes; underlined effects have a 95% interval that excludes zero and the same sign for at least 14 of 17 models.

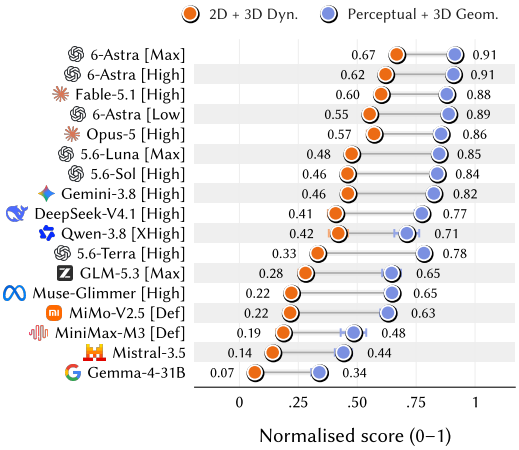

Figure 8: **Static–dynamic gap.** For each model, the blue dot averages the static families (Perceptual, 3D Geometry) and the orange dot the dynamic ones (2D and 3D Dynamics), with 95% intervals over scenes.

4.5 Effect of Inference-Time Reasoning

Increasing Astra's reasoning effort from Low to High to Max raises its Overall score from 0.73 to 0.77 to 0.79 (Figure 5). Its 2D Dynamics score increases from 0.48 to 0.55 to 0.60, and its 3D Dynamics score from 0.63 to 0.69 to 0.73. VQA similarly increases from 78.2% to 85.1% to 87.6%. These gains accompany greater output-token use and cost (Appendix Figure A21). Unlike the weak relationship between token use and quality across different models, additional reasoning within Astra consistently improves its reconstruction scores.

5 Related Work

Recent benchmarks have begun evaluating coding agents on spatial and physical tasks, from constructing 3D objects to implementing simple simulations. 3DCodeBench tests whether agents can model 3D objects in code from text or images (Gao et al., 2026). PhysCodeBench tests whether they can write executable physics simulations from text descriptions (Xie et al., 2026).

VisPhyWorld instead gives agents synthetic physics videos to reconstruct as simulator code (Liang et al., 2026). MPMWorlds also reconstructs synthetic videos, covering deformation, material changes, and topology-changing events in 2D (Kovačič & Ellis, 2026). BVB reconstructs primarily static indoor scenes observed by a moving camera (Tang et al., 2026). 4DCodeBench instead curates videos with minimal camera motion to test whether agents reconstruct changes in scene geometry and physical motion, which we also evaluate directly against 3D ground truth on synthetic scenes. More related work in Appendix B.

6 Conclusion

We introduced 4DCodeBench, which evaluates agents that reconstruct videos as executable 4D scenes. Across 17 models, automated rankings closely agree with human preferences, yet even the strongest models reconstruct dynamics less accurately than appearance and geometry. By measuring these capabilities across diverse scene ontologies, 4DCodeBench provides a way to track progress in 4D inverse graphics and identify where models still fail.

AI USE STATEMENT

We used generative AI tools to assist with several parts of benchmark construction and software development. For the VQA annotations, the authors wrote the underlying scene descriptions and seed questions, and a VLM was used to rephrase and format them for consistency and clarity. All resulting questions were reviewed and edited by the authors.

Generative AI tools also assisted with curating the real-world dataset: given author-specified criteria, they were used to search and filter large collections for candidate videos exhibiting particular physical behaviors. The authors manually inspected and confirmed all selected examples. Generative AI tools also assisted with software development for constructing the synthetic dataset, including implementation of scene-generation and rendering code; all resulting scenes were reviewed and validated by the authors. We additionally used generative AI tools for implementing plotting code to reproduce a consistent author-designed figure style across analyses, and for literature search.

AI-assisted code and references were reviewed and verified by the authors, and all benchmark data, annotations, figures, and scientific claims were manually checked. The authors made the final decisions regarding dataset composition, methodology, experiments, analysis, and presentation, and take responsibility for the final content of this work.

ETHICS STATEMENT

User study. All participants in our user study were undergraduate or graduate students from multiple universities, spanning a range of academic disciplines, and they participated on a voluntary basis. We do not collect personal or otherwise identifiable information about participants, and all participants consented to the use of their responses for the purposes of this study.

Data privacy and licensing. The real-world cases in 4DCODEBENCH are drawn from previously published datasets, Youtube Creative Commons videos and from publicly available videos on Pexels (Pexels, 2026), Pixabay (Pixabay, 2026), and Mixkit (Mixkit, 2026). We use these materials in accordance with their respective licenses and terms of use, which permit their use and modification for research purposes. We strictly adhere to the licensing requirements of all data sources. For videos hosted on these platforms, our dataset release will provide links to the original content together with our preprocessing scripts.

REPRODUCIBILITY STATEMENT

We will publicly release 4DCODEBENCH, including the benchmark data and annotations, evaluation code, metric implementations, model outputs, and scripts used to produce the reported results and figures. We will also release the execution environment as a container image, allowing the benchmark software stack and dependencies to be reproduced consistently across machines. New models can therefore be evaluated in the same environment and with the same scoring pipeline used in this paper.

REFERENCES

Arthur Allshire, Himanshu Gaurav Singh, Ritvik Singh, Adam Rashid, Hongsuk Choi, David McAllister, Justin Yu, Yiyuan Chen, Huang Huang, Pieter Abbeel, et al. Scalable behavior cloning with open data, training, and evaluation. *arXiv preprint arXiv:2606.27375*, 2026.

Ryoichi Ando. A cubic barrier with elasticity-inclusive dynamic stiffness. *ACM Transactions on Graphics (TOG)*, 2024.

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

Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. Videophy: Evaluating physical commonsense for video generation. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025.

Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 2013.

Jan Bender et al. SPlisHSPlasH Library. URL https://github.com/InteractiveComputerGraphics/SPlisHSPlasH.

Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*. Spie, 1992.

Blender Foundation. Blender. https://www.blender.org, 2026.

Florian Bordes, Quentin Garrido, Justine T Kao, Adina Williams, Michael Rabbat, and Emmanuel Dupoux. Intphys 2: Benchmarking intuitive physics understanding in complex synthetic environments. *arXiv preprint arXiv:2506.09849*, 2025.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

Bingyi Cao, Koert Chen, Kevis-Kokitsi Maninis, Kaifeng Chen, Arjun Karpur, Ye Xia, Sahil Dua, Tanmaya Dabral, Guangxing Han, Bohyung Han, et al. Tipsv2: Advancing vision-language pretraining with enhanced patch-text alignment. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *Proceedings of International Conference on Machine Learning (ICML)*, 2024.

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

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 2010.

Yu Fang, Minchen Li, Ming Gao, and Chenfanfu Jiang. Silly rubber: An implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids. *ACM Transactions on Graphics (TOG)*, 2019.

Yu Fang, Ziyin Qu, Minchen Li, Xinxin Zhang, Yixin Zhu, Mridul Aanjaneya, and Chenfanfu Jiang. IQ-MPM: An interface quadrature material point method for non-sticky strongly two-way coupled nonlinear solids and fluids. *ACM Transactions on Graphics (TOG)*, 2020.

Xiang Feng, Yunuo Chen, Chang Yu, Hao Su, Demetri Terzopoulos, Yin Yang, Joseph Masterjohn, Alejandro Castro, and Chenfanfu Jiang. MPM Lite: Linear kernels and integration without particles. *ACM Transactions on Graphics (TOG)*, 2026.

Yipeng Gao, Lei Shu, Genzhi Ye, Xi Xiong, Ameesh Makadia, Meiqi Guo, Laurent Itti, and Jindong Chen. 3dcodebench: Benchmarking agentic procedural 3d modeling via code. *arXiv preprint arXiv:2606.01057*, 2026.

Genesis AI Team. The role of simulation in scalable robotics, genesis world 1.0, and the path forward. *Genesis AI Blog*, May 2026. URL https://www.genesis.ai/blog/the-role-of-simulation-in-scalable-robotics-genesis-world-10-and-the-path-forward.

Guangzhao He, Rundong Luo, Wei-Chiu Ma, and Hadar Averbuch-Elor. Thinking in blender: Staged executable inverse graphics with vision-language models. *arXiv preprint arXiv:2606.02580*, 2026.

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 2018.

Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. Taichi: A language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6), 2019. doi: 10.1145/3355089.3356506.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020.

Yuanming Hu, Jiafeng Liu, Xuanda Yang, Mingkuan Xu, Ye Kuang, Weiwei Xu, Qiang Dai, William T. Freeman, and Frédo Durand. QuanTaichi: A compiler for quantized simulations. *ACM Transactions on Graphics (TOG)*, 40(4), 2021. doi: 10.1145/3450626.3459671.

Kemeng Huang, Floyd M. Chitalu, Huancheng Lin, and Taku Komura. Gipc: Fast and stable gauss-newton optimization of ipc barrier energy. *ACM Trans. Graph.*, 2024.

Kemeng Huang, Xinyu Lu, Huancheng Lin, Taku Komura, and Minchen Li. Stiffgipc: Advancing gpu ipc for stiff affine-deformable simulation. *ACM Transactions on Graphics (TOG)*, 2025.

Christian Internò, Alexander Pondaven, Habon Issa, Fabio Pizzati, Francesco Pinto, Markus Olhofer, Ivan Laptev, Philip Torr, Eero P Simoncelli, Barbara Hammer, et al. Geophys: The geometry of physical plausibility. *arXiv preprint arXiv:2606.20707*, 2026.

Chia-Hsiang Kao, Cong Phuoc Huynh, Chien-Yi Wang, Noranart Vesdapunt, Stefan Stojanov, Bharath Hariharan, Oleksandr Obiednikov, and Ning Zhou. δynamics: Language-based representation for inferring rigid-body dynamics from videos. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labeling real videos. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2025.

Daniel Kersten, Pascal Mamassian, and Alan Yuille. Object perception as bayesian inference. *Annual Review of Psychology*, 2004.

Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. Drucker–Prager elastoplasticity for sand animation. *ACM Transactions on Graphics (TOG)*, 2016.

Lingyu Kong, Ruicheng Li, Ruicheng Wang, Sicheng Xu, Chengtang Yao, Jianfeng Xiang, and Jiaolong Yang. Moge-3: Fine-detail monocular geometry estimation with self-guided sparse volumetric refinement. *arXiv preprint arXiv:2607.17967*, 2026.

Žiga Kovačič and Kevin Ellis. Mpmworlds: Material-point-method simulations for inferring and extrapolating physical dynamics. *arXiv preprint arXiv:2606.01538*, 2026.

Peter Kulits, Haiwen Feng, Weiyang Liu, Victoria Fernandez Abrevaya, and Michael J. Black. Re-thinking inverse graphics with large language models. *Transactions on Machine Learning Research*, 2024.

Tejas D Kulkarni, Pushmeet Kohli, Joshua B Tenenbaum, and Vikash Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Lei Lan, Danny M. Kaufman, Minchen Li, Chenfanfu Jiang, and Yin Yang. Affine body dynamics: Fast, stable, and intersection-free simulation of stiff materials. *ACM Transactions on Graphics (TOG)*, 2022.

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. Incremental potential contact: Intersection- and inversion-free, large-deformation dynamics. *ACM Transactions on Graphics (TOG)*, 2020.

Puyin Li, Tiange Xiang, Ella Mao, Shirley Wei, Xinye Chen, Adnan Masood, Fei-Fei Li, and Ehsan Adeli. Quantiphy: A quantitative benchmark evaluating physical reasoning abilities of vision-language models. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Wenqiao Li, Yao Gu, Xintao Chen, Xiaohao Xu, Ming Hu, Xiaonan Huang, and Yingna Wu. Towards visual discrimination and reasoning of real-world physical dynamics: Physics-grounded anomaly detection. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Jiarong Liang, Max Ku, Ka-Hei Hui, Ping Nie, and Wenhu Chen. Visphyworld: Probing physical reasoning via code-driven video reconstruction. *arXiv preprint arXiv:2602.13294*, 2026.

Litian Liang, Liuyu Bian, Caiwei Xiao, Jialin Zhang, Linghao Chen, Isabella Liu, Fanbo Xiang, Zhiao Huang, and Hao Su. Robo360: a 3d omnispective multi-material robotic manipulation dataset. *arXiv preprint arXiv:2312.06686*, 2023.

Chunjiang Liu, Xiaoyuan Wang, Haoyu Chen, Yizhou Zhao, Ming-Hsuan Yang, and László A Jeni. Simworlds: A multi-agent system for dynamic 3d scene creation. *arXiv preprint arXiv:2607.01766*, 2026.

Michael Liu, Xinlei Wang, and Minchen Li. CK-MPM: A compact-kernel material point method. *ACM Transactions on Graphics (TOG)*, 44(4), 2025. doi: 10.1145/3731155.

Miles Macklin. Warp: A High-performance Python Framework for GPU Simulation and Graphics, March 2022. URL https://github.com/NVIDIA/warp. NVIDIA GPU Technology Conference (GTC).

Mixkit. Mixkit stock video free license, 2026. URL https://mixkit.co/license/.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? In *Proceedings of Winter Conference on Applications of Computer Vision (WACV)*, 2026.

Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, Anirudh Pai, Yuvan Sharma, Stefano Saravalle, Ruijie Zheng, Jing Wang, et al. T-rex: Tactile-reactive dexterous manipulation. *arXiv preprint arXiv:2606.17055*, 2026.

Pexels. Pexels license - what is allowed?, 2026. URL https://www.pexels.com/license/.

Pixabay. Pixabay content license summary, 2026. URL https://pixabay.com/service/license-summary/.

René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020.

Daniel Ritchie, Paul Guerrero, R Kenny Jones, Niloy J Mitra, Adriana Schulz, Karl DD Willis, and Jiajun Wu. Neurosymbolic models for computer graphics. In *Computer graphics forum*, 2023.

Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. In *Conference on Robot Learning (CoRL)*, 2023.

Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. *The International Journal of Robotics Research*, 2024.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 2007.

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 2013.

Yu Sun, Meng Cao, Yang Ping, Kaidong Zhang, Qingxuan Chen, Rongtao Xu, Liangwang Ruan, Xuecheng Chen, Dongxiu Liu, Yunxiao Yan, et al. Maniparena: Comprehensive real-world evaluation of reasoning-oriented generalist robot manipulation. *arXiv preprint arXiv:2603.28545*, 2026.

Krishna Suresh and Chris Atkeson. Learning dynamic rope manipulation using task-level iterative learning control. In *Robotics: Science and Systems (RSS)*, 2026.

Yolo Y Tang, Daiki Shimada, Jiayue Meng, Jing Bi, Pinxin Liu, Yicheng Wang, Yunzhong Xiao, Zhangyun Tan, Zeliang Zhang, Chao Huang, et al. Bvb: Benchmarking agentic video understanding via programmatic reconstruction in blender. *arXiv preprint arXiv:2609.15478*, 2026.

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2020.

Hsiao-Yu Tung, Mingyu Ding, Zhenfang Chen, Daniel Bear, Chuang Gan, Josh Tenenbaum, Dan Yamins, Judith Fan, and Kevin Smith. Physion++: Evaluating physical scene understanding that requires online inference of different physical properties. In *Proceedings of the Neural Information Processing Systems (NeurIPS) Track on Datasets and Benchmarks*, 2023.

Tomer D Ullman, Elizabeth Spelke, Peter Battaglia, and Joshua B Tenenbaum. Mind games: Game engines as an architecture for intuitive physics. *Trends in cognitive sciences*, 2017.

Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. 1991.

Hanyang Wang, Yimo Cai, Weiliang Chen, Jiawei Chi, Haowen Sun, Qiyu Dai, Yi-Hsin Hung, Xingzhuo Guo, Jinshan Ren, Runmao Yao, et al. Code as worlds: Agentic discovery of executable world representations for physical reasoning. *arXiv preprint arXiv:2608.27549*, 2026.

Jing Wang, Ao Ma, Ke Cao, Jun Zheng, Jiasong Feng, Zhanjie Zhang, Wanyuan Pang, and Xiaodan Liang. Wisa: World simulator assistant for physics-aware text-to-video generation. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Xinlei Wang, Minchen Li, Yu Fang, Xinxin Zhang, Ming Gao, Min Tang, Danny M. Kaufman, and Chenfanfu Jiang. Hierarchical optimization time integration for CFL-rate MPM stepping. *ACM Transactions on Graphics (TOG)*, 39(3), 2020. doi: 10.1145/3386760.

Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. CD-MPM: Continuum damage material point methods for dynamic fracture animation. *ACM Transactions on Graphics (TOG)*, 2019.

Joshuah Wolper, Yunuo Chen, Minchen Li, Yu Fang, Ziyin Qu, Jiecong Lu, Meggie Cheng, and Chenfanfu Jiang. AnisoMPM: Animating anisotropic damage mechanics. *ACM Transactions on Graphics (TOG)*, 2020.

Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *British Machine Vision Conference (BMVC)*, 2016.

Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2017a.

Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. Neural scene de-rendering. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017b.

Tianyidan Xie, Peiyu Wang, Hu Jiaxin, Yuyi Qian, Yuxuan Wang, Shenyi Wang, Rui Ma, Yanlun Peng, Lanjun Wang, Ying Tai, et al. Physcodebench: Benchmarking physics-aware symbolic simulation of 3d scenes via self-corrective multi-agent refinement. *arXiv preprint arXiv:2604.23580*, 2026.

Shaofeng Yin, Jiaxin Ge, Zora Zhiruo Wang, Chenyang Wang, Xiuyu Li, Michael J Black, Trevor Darrell, Angjoo Kanazawa, and Haiwen Feng. Vision-as-inverse-graphics agent via interleaved multimodal reasoning. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2026.

Alan Yuille and Daniel Kersten. Vision as bayesian inference: analysis by synthesis? *Trends in cognitive sciences*, 2006.

Yi Zhang, Yunshuang Wang, Zeyu Zhang, and Hao Tang. Code2worlds: Empowering coding llms for 4d world generation. In *Proceedings of International Conference on Machine Learning (ICML)*, 2026.

Tony Z Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Kamyar Ghasemipour, Chelsea Finn, and Ayzaan Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *Conference on Robot Learning (CoRL)*, 2024.

Yifei Zhao, Xiangxin Zhou, Wenhao Yang, Jiaqi Tang, Pu Jian, Huanjin Yao, Jiarui Yao, Haowei Lin, Chunchao Guo, Zhuo Chen, et al. Sceneactbench: Can agents act on the 3d scenes they see? *arXiv preprint arXiv:2607.22393*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2024.

APPENDIX

A Task Illustration **16**

B More Related Work **16**

C Dataset **17**

C.1 Dataset Samples 18

C.2 Real-World Dataset Composition and Sources 18

C.3 Synthetic Dataset Simulators 18

D Metrics and Evaluation **23**

D.1 Executability and Scoring 23

D.2 Metrics on All Scenes 23

D.3 Metrics on Real Videos 25

D.4 Metrics on Synthetic Scenes 25

D.5 VQA 26

D.6 VLM-based pairwise rankings. 28

D.7 Metric Concordance 28

D.8 Additional Metrics and Geometry Quality 28

E Human Study **30**

F Additional Results **34**

F.1 Sample Agent Results 34

F.2 VQA Results 34

F.3 How do models describe motion? 34

G Experimental Setup **37**

G.1 Execution Environment 37

G.2 Agent Prompt 37

G.3 Submission Format 38

G.4 Running Statistics 39

A TASK ILLUSTRATION

B MORE RELATED WORK

Physical world understanding. Humans reason about physical scenes by mentally simulating them with an approximate internal physics engine, grounded in core knowledge of objects and mechanics (Spelke & Kinzler, 2007; Battaglia et al., 2013; Ullman et al., 2017). Early computational frameworks such as Galileo and visual de-animation operationalize this principle by inferring structured, object-centric states from video and simulating them to predict and explain observed dynamics (Wu et al., 2015; 2017a). Most contemporary benchmarks, however, probe physical

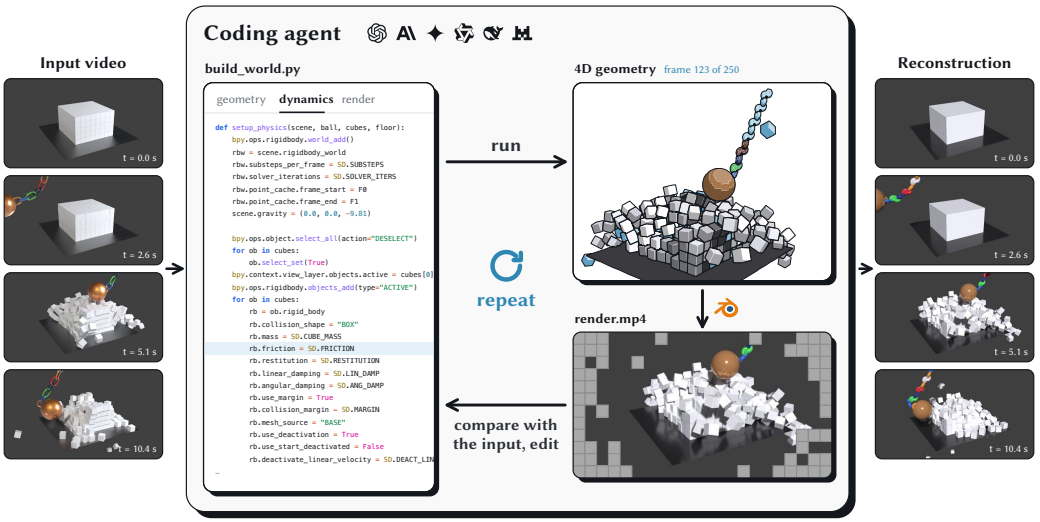

Figure A1: **Task.** Given the input video, the coding agent writes a program that builds a 4D world, runs and renders it, compares the render with the input, and edits the program in a loop.

reasoning through indirect surrogate tasks rather than explicit state representations—relying on outcome prediction and plausibility judgments (Tung et al., 2023; Bordes et al., 2025), question answering (Chow et al., 2025; Li et al., 2026), or the physical plausibility of generated videos (Bansal et al., 2025; Motamed et al., 2026). Formulating scene understanding as world-program generation revives this explicit state representation, requiring agents to commit their physical understanding to an executable 4D world that can be simulated, inspected, and verified.

Inverse graphics and executable world modeling. Inverse graphics frames visual perception as inverting the forward rendering process through analysis-by-synthesis (Kersten et al., 2004; Yuille & Kersten, 2006). Classical visual program induction pioneered this philosophy by recovering scenes as programs (Kulkarni et al., 2015; Wu et al., 2017b; Ritchie et al., 2023), but was largely constrained to hand-crafted domain-specific languages. Recent LLM-powered coding agents lift this limitation, parsing static objects and indoor layouts from imagery into general-purpose graphics code (Kulits et al., 2024; Yin et al., 2026; He et al., 2026; Gao et al., 2026). Extending such programs to dynamic scenes introduces a fundamentally more demanding inverse problem: the program must capture not only spatial geometry, but also how the physical scene evolves over time. Forward simulation from text (Zhang et al., 2026; Liu et al., 2026; Xie et al., 2026) avoids this inverse challenge, as generated code needs only to satisfy an open-ended description rather than match a specific physical observation. Video-conditioned benchmarks directly tackle this inverse problem, yet each focuses on a restricted regime (Table A1): VisPhyWorld and MPMWorlds operate on controlled synthetic simulations, with the latter confined to 2D (Liang et al., 2026; Kovačič & Ellis, 2026); Δynamics and SceneActBench target primarily rigid and articulated systems (Kao et al., 2026; Zhao et al., 2026); and BVB reconstructs static indoor scenes under moving camera trajectories (Tang et al., 2026). Concurrently, Code as Worlds builds executable scenes from real footage, but its video pipeline fits object trajectories kinematically rather than inferring the underlying physics (Wang et al., 2026). By contrast, 4DCODEBENCH establishes a unified inverse-graphics benchmark across diverse real-world footage—spanning rigid, deformable, co-dimensional, and flowing matter—while evaluating the reconstructed 3D geometry and motion directly against ground truth on synthetic scenes.

C DATASET

Figure A2 shows the number of scenes in 4DCODEBENCH that include an object of each material category (left) and the distributions of clip length and frame rate (middle, right).

Table A1: Comparison with benchmarks for code-based world generation and reconstruction. Type: generation from text (Gen.) or reconstruction from images or video (Recon.). Real: real visual inputs. 3D geom.: the output contains 3D scene geometry. Dynamics columns mark which of our matter classes move in the tasks. 3D eval.: scores 3D geometry or simulator state rather than only rendered frames.

				Dynamics by matter class				
Benchmark	**Type**	**Real**	**3D geom.**	**Rigid**	**Deform.**	**Co-dim.**	**Flowing**	**3D eval.**
3DCodeBench (2026)	Gen./Recon.	✗	✓	✗	✗	✗	✗	✓
PhysCodeBench (2026)	Gen.	–	✓	✓	✓	✓	✓	✓
Code4D (2026)	Gen.	–	✓	✓	✓	✓	✓	✗
4DBuildBench (2026)	Gen.	–	✓	✓	✓	✓	✓	✓
VisPhyBench (2026)	Recon.	✗	✓	✓	✗	✗	✗	✓
MPMWorlds (2026)	Recon.	✗	✗	✓	✓	✗	✓	✗
Δynamics (2026)	Recon.	✓	✓	✓	✗	✗	✗	✓
BVB (2026)	Recon.	✓	✓	✗	✗	✗	✗	✗
SceneActBench (2026)	Recon.	✗	✓	✓	✗	✗	✗	✓
Ours	**Recon.**	✓	✓	✓	✓	✓	✓	✓

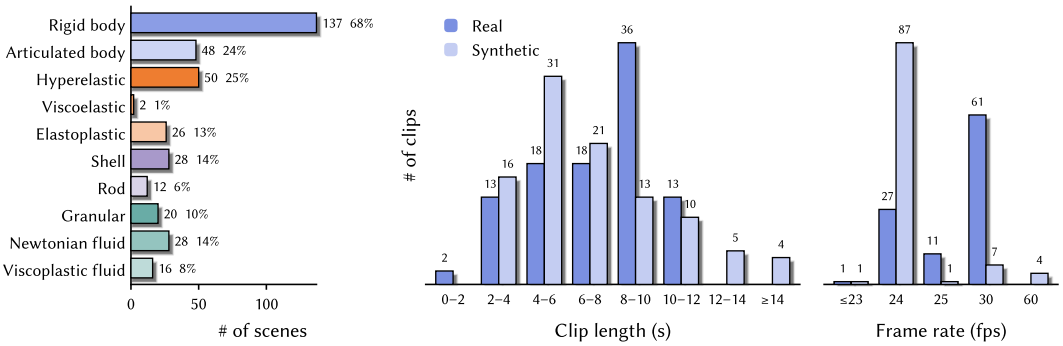

Figure A2: **Dataset distributions.** The number of scenes that contain each material and their proportion of the 200 scenes (left); since a scene can contain several materials, the shares sum to more than 100%. The clip length (middle) and frame rate (right) distribution of the reference videos, shown separately for real and synthetic scenes.

C.1 Dataset Samples

Figures A3–A6 show 40 of the 200 scenes: the synthetic pages cover every simulation method, and the real pages cover every physics category.

C.2 Real-World Dataset Composition and Sources

The real-world subset contains 100 video clips drawn from physics video benchmarks (51 clips), robot manipulation datasets (21 clips), and independently collected web footage (28 clips). These sources provide controlled physical demonstrations, robotic interactions with objects and materials, and physical events in everyday settings. Table A2 lists the sources and their contributions.

C.3 Synthetic Dataset Simulators

We use several simulation methods to cover the range of materials and interactions represented in the synthetic subset. At a high level, Incremental Potential Contact (IPC) formulates contact using barrier potentials to robustly prevent intersections during deformable simulation Li et al. (2020), while Affine Body Dynamics (ABD) extends related ideas to stiff and near-rigid bodies Lan et al. (2022). Smoothed Particle Hydrodynamics (SPH) represents fluids using Lagrangian particles and is used for liquid dynamics, including fluid–rigid interactions and surface-tension effects. Material Point Method (MPM) combines Lagrangian material points with an Eulerian background grid and



Figure A3: **Synthetic scenes (1/2).** Each scene is shown as four frames spaced evenly over its reference video.

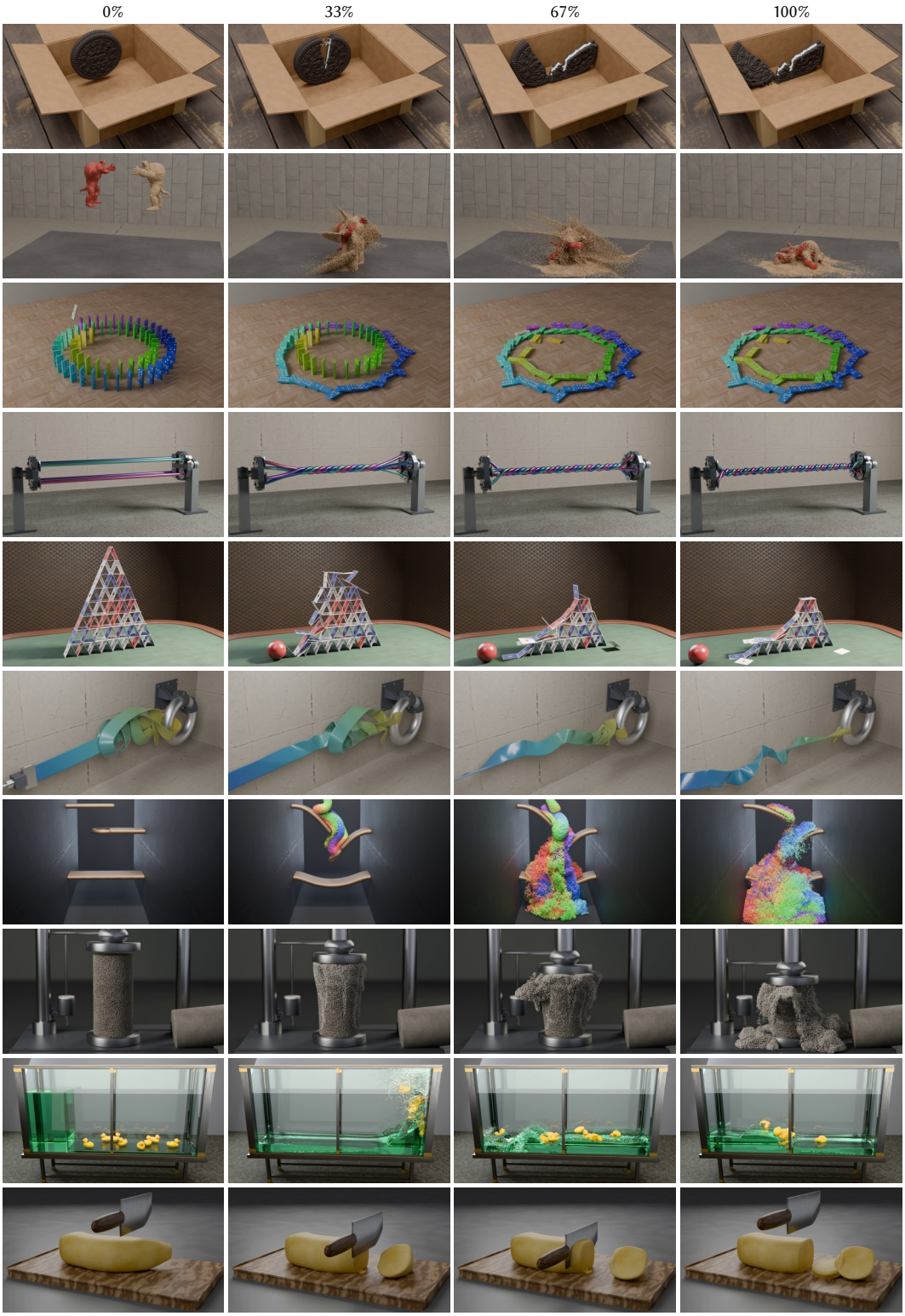

Figure A4: **Synthetic scenes (2/2).** Each scene is shown as four frames spaced evenly over its reference video.

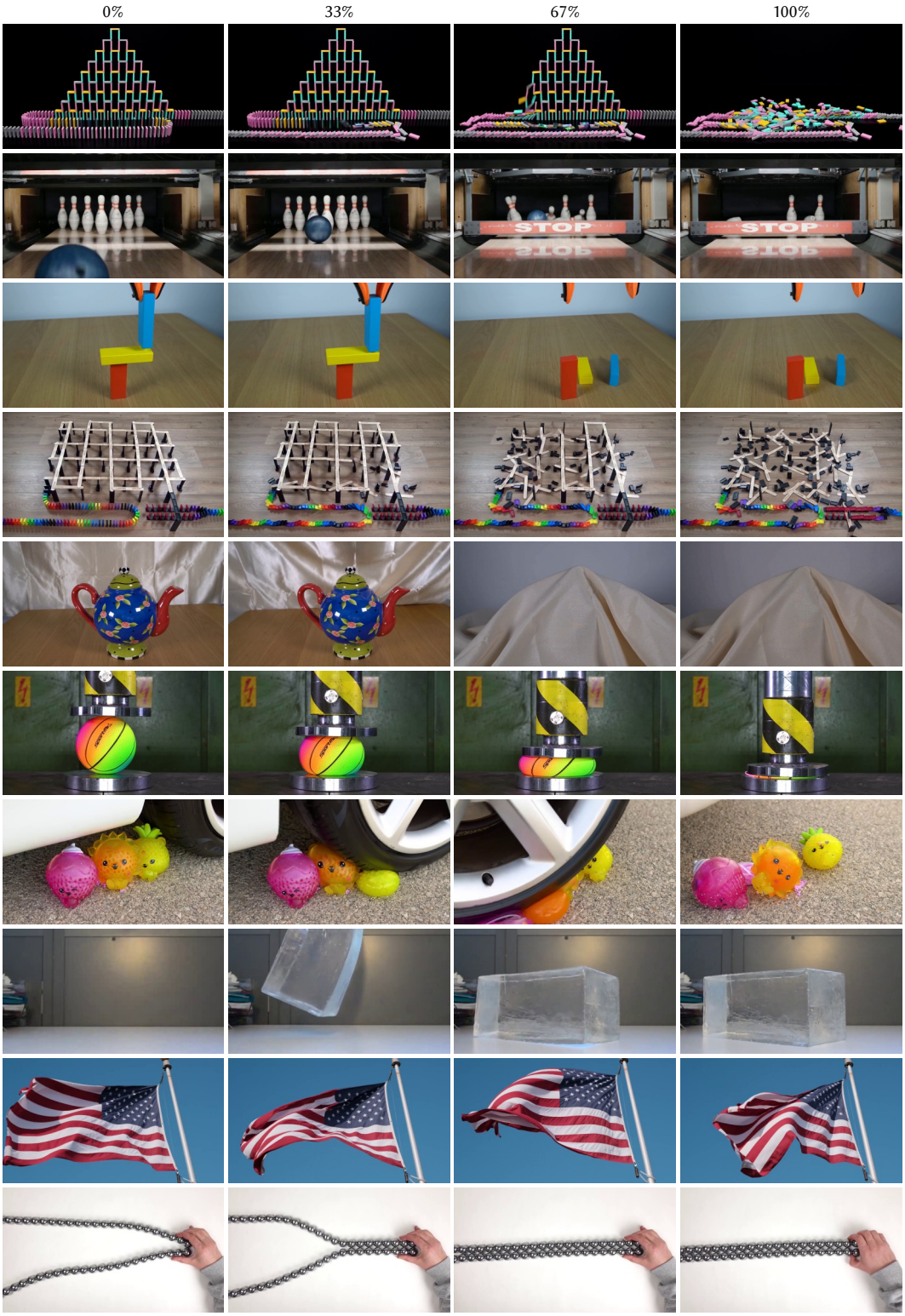

Figure A5: **Real-world scenes (1/2).** Each scene is shown as four frames spaced evenly over its reference video.

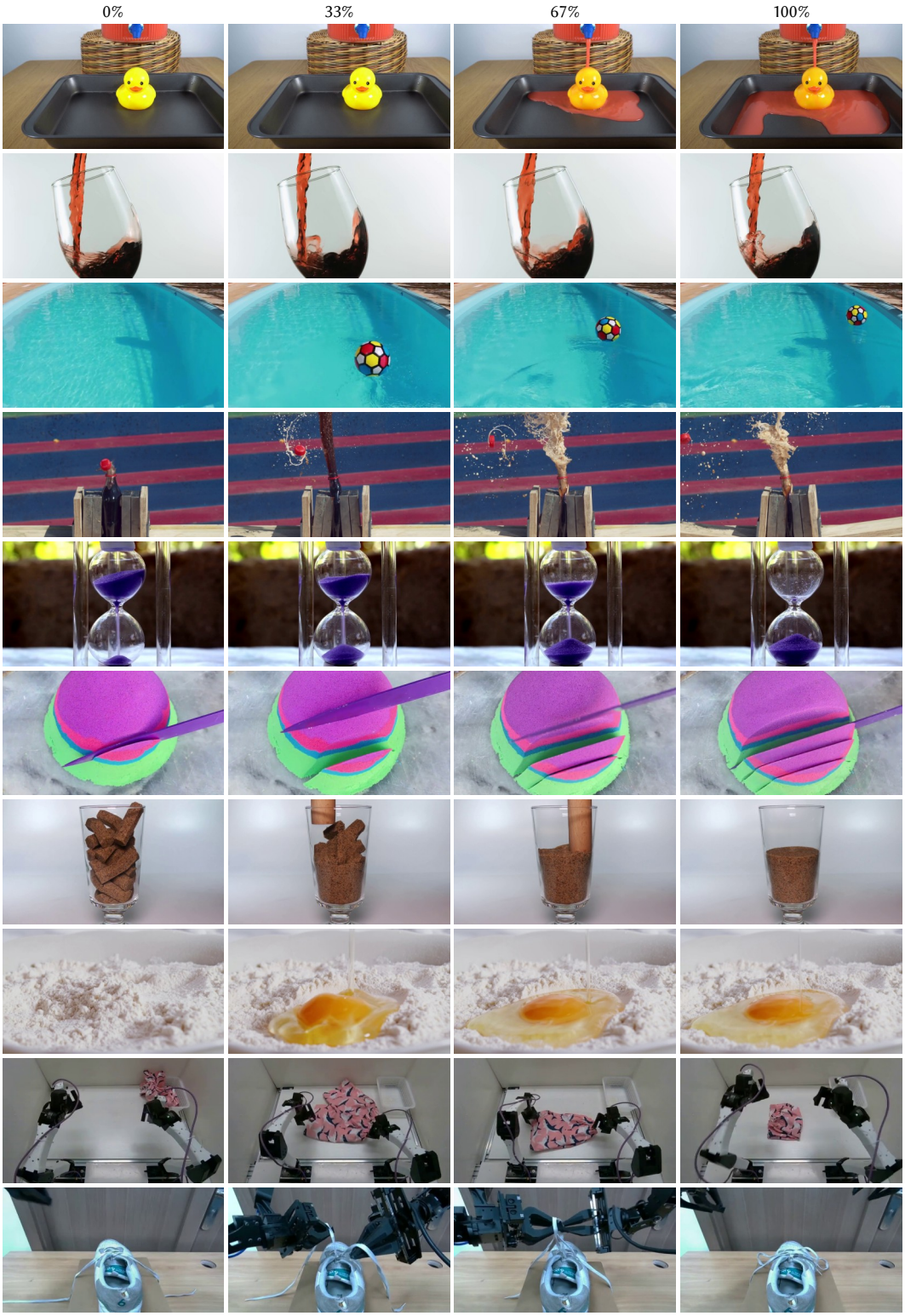

Figure A6: **Real-world scenes (2/2).** Each scene is shown as four frames spaced evenly over its reference video.

Table A2: **Real-world video sources** and the number of clips drawn from each.

Source type	Source	Clips
Physics video benchmark	WISA-80K (Wang et al., 2025)	33
	Physics-IQ (Motamed et al., 2026)	12
	Phys-AD (Li et al., 2025)	3
	Phys101 (Wu et al., 2016)	3
Robot manipulation	ABC-130K (Allshire et al., 2026)	5
	T-Rex (Niu et al., 2026)	4
	Robo360 (Liang et al., 2023)	3
	AgiBot World (Bu et al., 2025)	2
	RoboCook (Shi et al., 2023)	2
	RoboCraft (Shi et al., 2024)	2
	ALOHA Unleashed (Zhao et al., 2024)	1
	ManipArena (Sun et al., 2026)	1
	Task-Level ILC (Suresh & Atkeson, 2026)	1
Web-collected	Pexels	21
	Youtube Creative Commons	5
	Mixkit	1
	Pixabay	1
Total		**100**

supports a broad range of constitutive behaviors, including elasticity, plasticity, fracture, and coupled solid–fluid dynamics. We use several MPM variants specialized for different numerical and material regimes. Finally, the PPF Contact Solver uses finite-element models together with barrier-based contact handling for contact-rich simulations of shells, rods, and solids Ando (2024). More than 2/3 of the synthetic examples are generated using a simulator that is more advanced and rich than Blender.

We extend the Genesis simulator Genesis AI Team (2026) with three new MPM constitutive models: a viscoplastic model to simulate materials such as shaving cream and toothpaste; the snow model of Stomakhin et al. Stomakhin et al. (2013); and a Drucker–Prager model Klár et al. (2016) for sand. Sand and snow materials do exist in Genesis, but are implemented using different approaches than the ones we add.

D Metrics and Evaluation

D.1 Executability and Scoring

To ensure benchmark scores reflect both reconstruction fidelity and practical execution reliability, we first evaluate the validity of each submission. A submission is designated as *executable* if (i) its rendered video successfully decodes and matches the reference in resolution, frame count, and frame rate, and (ii) its 4D world passes structural format validation. Submissions producing an invalid video automatically fail all metrics; those producing a valid video but an invalid 4D world fail only geometry-dependent metrics. When a metric fails due to non-executability, it receives its worst possible value—0 for bounded similarities and 2 for normalized depth error (matching their bounds in Sec. D.2 and D.3)—so that aggregate scores penalize unexecutable outputs. When the reference scene itself lacks the prerequisite signal for a specific metric (e.g., it contains no trackable dynamic motion), that scene is excluded from the corresponding metric across all models. Figure **??** shows the executability of each model: 89.6% of all submissions are executable, with at least 96.5% for every proprietary model and 29.0–95.5% for open-weight models.

D.2 Metrics on All Scenes

Let x_k and $\hat{x}_k$ denote frame k of the reference and rendered videos, respectively ($k = 0, \ldots, T-1$).

Table A3: **Simulators of the synthetic split**, the phenomena each one covers, and the number of clips it produced. All scenes are rendered in Blender.

Simulator	Simulation role	Clips
Blender	Simpler physical scenes	31
libuipc Huang et al. (2025; 2024)	Rigid, deformable, shell, and rod simulation with contact; affine-body assemblies	10
SPlisHSPlasH Bender et al.	Liquids, fluid–rigid coupling, and surface tension	3
MLS-MPM Hu et al. (2018)	Cutting, discontinuities, and rigid-body coupling	3
CK-MPM Liu et al. (2025)	Deformable and elastoplastic materials	9
MPM-lite Feng et al. (2026)	Elastic and elastoplastic materials	4
HOT Wang et al. (2020)	Stiff deformable and elastoplastic materials	7
AnisoMPM Wolper et al. (2020)	Anisotropic damage and fracture; Drucker–Prager sand	4
CD-MPM Wolper et al. (2019)	Continuum-damage fracture	4
Silly Rubber Fang et al. (2019)	Non-equilibrated viscoelastic and elastoplastic solids	3
IQ-MPM Fang et al. (2020)	Coupled deformable solids and fluids	1
PPF Ando (2024)	Co-dimensional structures, deformables, and contact	9
Genesis Genesis AI Team (2026)	Robotic manipulation, viscoplastic fluids, and MPM–SPH coupling	12
Total		**100**

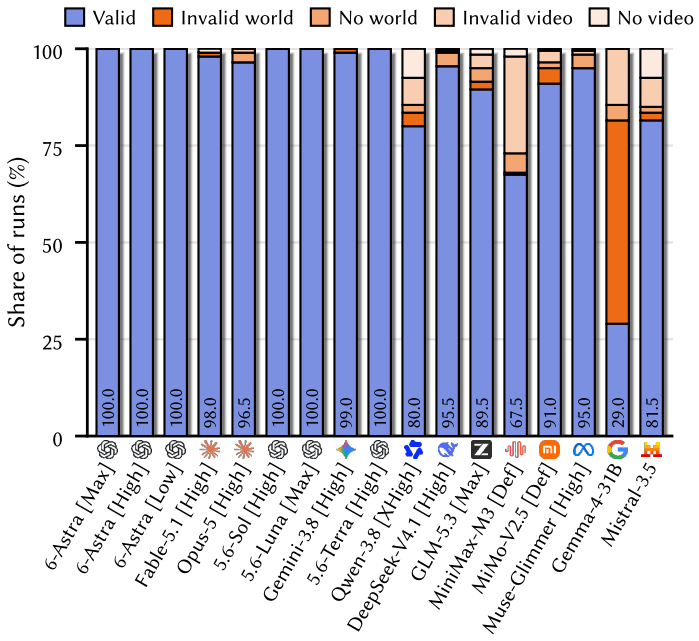

Figure A7: **Executability breakdown across models.**

DINOv3 similarity. To evaluate high-level visual appearance and semantic consistency across frames without being overly sensitive to minor pixel-level illumination or color shifts, we compute deep feature similarity using DINOv3 (Siméoni et al., 2025). Using the class-token embeddings e_k and $\hat{e}_k$ of the reference and rendered frames, we map the mean frame-wise cosine similarity to $[0, 1]$:

$$\text{DINOv3} = \frac{1}{2}\Big(1 + \frac{1}{T}\sum_{k} \frac{e_k^\top \hat{e}_k}{\|e_k\|\,\|\hat{e}_k\|}\Big). \tag{2}$$

Dynamic IoU. To capture the coarse spatial extent and general motion trends of dynamic elements across time, we measure Dynamic IoU. Let $\hat{M}_k$ denote the rasterized mask of dynamic objects defined by the submission's world program, and M_k the reference dynamic mask, rasterized from the

reference world or manually annotated for real videos. Following PASCAL VOC (Everingham et al., 2010), we ignore a thin boundary band B_k to avoid edge rasterization artifacts:

$$\mathrm{DynIoU} = \frac{1}{T}\sum_{k}\frac{|(M_k \cap \hat{M}_k) \setminus B_k|}{|(M_k \cup \hat{M}_k) \setminus B_k|}. \tag{3}$$

Scenes whose main object is transparent are excluded, as surface rasterization cannot capture background visibility.

D.3 Metrics on Real Videos

For real videos, reference motion and depth are estimated from the reference video, whereas reconstruction metrics are analytically rasterized directly from the predicted 4D world to assess true underlying geometry rather than rendered pictures.

Depth error. To evaluate 3D spatial layout and relative distance ordering from a monocular viewpoint, we compute normalized disparity error against estimated reference depth. Let $\tilde{d}_k$ and $\tilde{\hat{d}}_k$ be the disparities from Video Depth Anything (Chen et al., 2025) and the camera-rasterized reconstruction mesh, respectively, each median-and-MAD normalized as in MiDaS (Ranftl et al., 2020). Discarding the largest 1% of residuals per frame to suppress outlier sensitivity, the pixel-averaged error over N pixels is:

$$\mathrm{DepthErr} = \frac{1}{T}\sum_{k}\frac{1}{N}\sum_{p}\left|\tilde{d}_k(p) - \tilde{\hat{d}}_k(p)\right|, \tag{4}$$

which is bounded in $[0, 2]$.

Flow. To assess 2D apparent motion dynamics and velocity distributions across consecutive frames, we compare optical flow fields. Let U_k be the reference RAFT (Teed & Deng, 2020) flow, and $\hat{U}_k$ the reconstruction flow derived analytically from surface geometry, between frames k and $k+1$ at the pixels that move in either, and $\mathcal{K}$ the steps at which any pixel moves. Rather than penalizing minor spatial misalignments with rigid pixel-wise end-point error, we compare them as unordered sets at each motion step $k \in \mathcal{K}$ using the sliced Wasserstein distance (SW), a fast approximation of the earth mover's distance (EMD). Normalizing by the root-mean-square magnitude m of each side ensures slow and fast scenes share one common scale:

$$\mathrm{Flow} = 1 - \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\min\left(1, \frac{\mathrm{SW}(U_k, \hat{U}_k)}{m(U_k) + m(\hat{U}_k)}\right). \tag{5}$$

Track2D. To evaluate long-range temporal motion persistence and tracking fidelity over extended sequences, we compute 2D point trajectory concordance. CoTracker3 (Karaev et al., 2025) tracks a grid of query points through the reference video. On the reconstruction, the corresponding 3D surface point under each query is followed through the 4D world and projected into the frame, so path i on one side corresponds to path i on the other. Retaining the n reference paths that travel more than 2% of the image diagonal δ after camera motion compensation, we compute the Dynamic Time Warping (DTW) distance D_i between corresponding paths in units of δ, which accommodates variations in execution speed, capped at $\kappa = 0.03$:

$$\mathrm{Track2D} = 1 - \frac{1}{n}\sum_{i}\frac{\min(D_i, \kappa)}{\kappa}. \tag{6}$$

D.4 Metrics on Synthetic Scenes

Because synthetic scenes provide complete ground-truth geometry, the metrics below are evaluated directly in 3D after registering the reconstruction to the reference world.

Registration. Let $\mathcal{P}$ be the point cloud of surfaces visible at frame 0 of the reference world, with RMS radius σ; all 3D distances are measured in units of σ. Because lower-performing models frequently produce erroneous camera extrinsics due to subtle Blender camera export API conventions, relying solely on camera-view surfaces can lead to catastrophic misalignment despite otherwise plausible reconstruction. To prevent these models from failing 3D metrics purely due to misplaced cameras, we fit two similarity transforms by trimmed ICP (Besl & McKay, 1992; Umeyama, 1991)—one on the visible surfaces at frame 0 and one on the dynamic objects' trajectories across the video—and keep the one that brings the reconstruction closer to $\mathcal{P}$ through the reference camera.

Scene 3D. To measure static 3D geometric fidelity and surface reconstruction accuracy under full 3D supervision, we evaluate the aligned frame-0 reconstruction against the reference ground truth. Let $\hat{\mathcal{P}}$ be the registered reconstruction's frame-0 surfaces seen through the reference camera, and D_κ the symmetric Chamfer distance capped at $\kappa = 0.5$:

$$\text{Scene3D} = 1 - D_\kappa(\mathcal{P}, \hat{\mathcal{P}})/\kappa. \tag{7}$$

Trajectory DTW. To evaluate Lagrangian motion consistency—measuring whether distinct physical matter follows the correct 3D trajectory over time—we compute path-level dynamic time warping. We sample dynamic matter across both worlds uniformly and follow each sample over time, giving two sets of 3D paths $\{\tau_i\}$ and $\{\hat{\tau}_j\}$ of equal size n. Solving the optimal one-to-one assignment π via the Hungarian algorithm with saturation threshold $\kappa = 1$:

$$\text{TrajDTW} = 1 - \frac{1}{n}\sum_i \frac{\min\left(\text{DTW}(\tau_i, \hat{\tau}_{\pi(i)}), \kappa\right)}{\kappa}. \tag{8}$$

EMD step. To evaluate Eulerian instantaneous velocity fields—capturing overall 3D physical speed and directional plausibility at each frame without requiring persistent particle tracking—we measure one-step displacement distributions. Let U_k and $\hat{U}_k$ be the one-step 3D displacements of the same samples at frame k. To evaluate instantaneous velocity distributions without enforcing trajectory-level temporal persistence, we measure their sliced Wasserstein distance (SW):

$$\text{EMD step} = \max\left(0,\ 1 - \frac{\sum_k \text{SW}(U_k, \hat{U}_k)}{\sum_k \left(m(U_k) + m(\hat{U}_k)\right)}\right). \tag{9}$$

Normalizing by the total motion over the video prevents nearly static frames from dominating the score.

D.5 VQA

Each scene carries hand-written binary questions in four categories: *initial state* and *final state* (what is present and where, in the first and last frames), *key events* (what happens in between, and in what order), and *contact* (what the materials do where they touch). Over the 200 scenes there are 1241 questions: 369 initial state, 353 final state, 303 key events, and 216 contact. A VLM judge (`Gemini 3.8 Flash`) sees 16 frames of the reconstruction–including the first and last frame–and answers all of a scene's questions in one call, without the reference video.

Gating. Two checks run before the VLM is asked to answer VQAs. Layer 1 catches renders with nothing in them: a render fails if it has almost no edges, or if almost every pixel is the same colour. These are blank, black, or flat-lit outputs. The second check is per scene. Every scene carries a presence question about its main object, and if the reconstruction fails it, the object is not in the video and the remaining answers say nothing about physics. The question essentially asks if the reconstruction contains anything that could even remotely try to represent the main dynamic object of the scene.

We score the questions of a gated scene wrong rather than skipping them, and we do the same for scenes that produced no video. Every model therefore answers the same 1241 questions, so no model can raise its average by rendering fewer scenes. Across all models, 77.4% of scenes reach the judge cleanly, 7.3% produce no video, 8.5% fail Layer 1, and 7.0% fail the presence question. Figure A8 shows the questions of two scenes with the answers of two models on each.

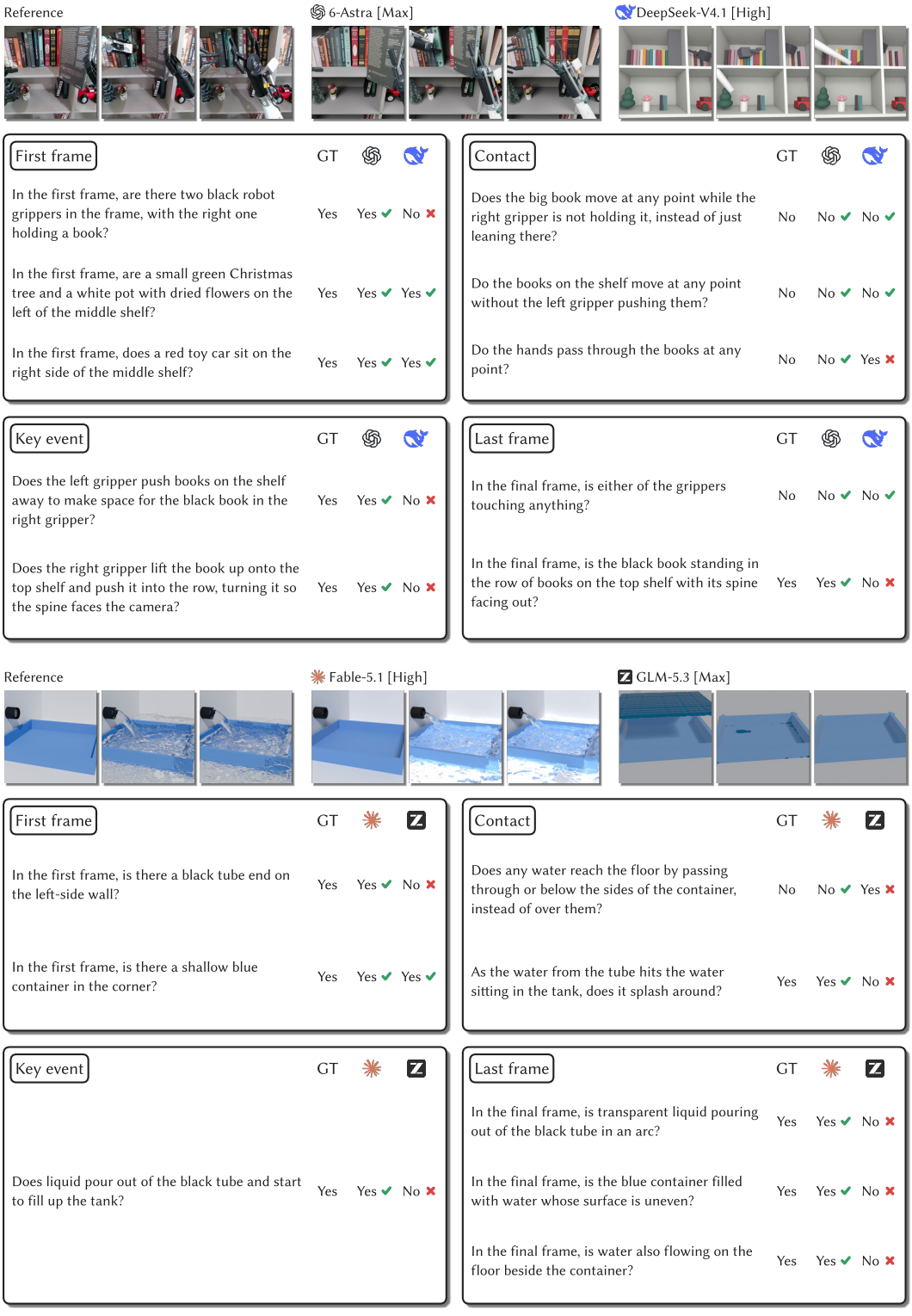

Figure A8: **VQA examples on a real rigid (top) and a synthetic flowing (bottom) scene.** Each shows frames at the start, middle, and end of the reference video and of two reconstructions, and up to three questions per category with the ground-truth (GT) answer and each model's answer, marked right or wrong.

D.6 VLM-BASED PAIRWISE RANKINGS.

For each comparison, the judge receives the reference video and two model reconstructions, sampled at 16 frames per video. The model used is (`Gemini 3.8 Flash`). The VLM chooses which reconstruction more closely matches the reference in object geometry, motion, collisions, deformation, and final state. Model identities are hidden and A/B assignment is balanced across comparisons.

We sample model pairs across scenes rather than exhaustively evaluating every pair. Pairwise outcomes are aggregated with a Bradley–Terry model and reported on the Elo scale; draws count as half a win for each side. Confidence intervals are obtained by bootstrapping scenes.

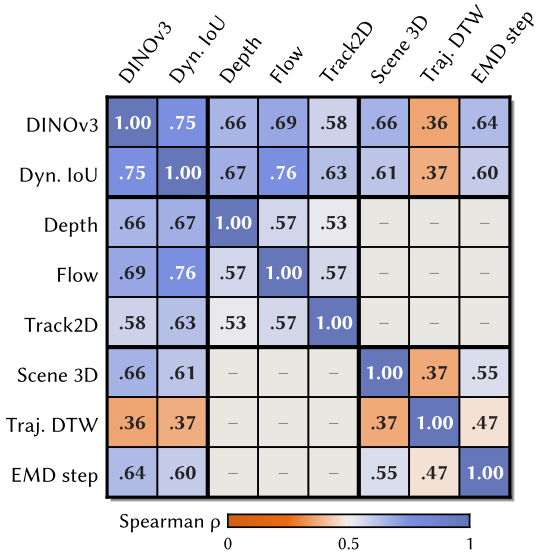

Figure A9: **Metric concordance.** Each cell is Spearman's ρ between the rankings that two metrics assign to the runs on a scene, averaged over scenes; metrics are oriented so that higher is better, and grey pairs share no scene.

Pairwise judge prompt

```
You are shown a reference recording of a physical event, then two
attempts by coding models to reproduce it: video A and video B.

Which attempt is closer to the reference? Consider the geometry of
the objects in the scene, how they move, the physics, the
collisions, the deformations, where things end up.

Answer with one word: A or B.
```

D.7 METRIC CONCORDANCE

Figure A9 evaluates whether two metrics rank submissions to the same scene consistently. Every pair agrees positively, with ρ between 0.36 and 0.76, indicating that the metrics point in a common direction without being redundant. Appearance and layout agree closely: DINOv3 similarity and Dynamic IoU correlate at 0.75, and both at 0.66–0.67 with depth error. The motion metrics agree only moderately with each other and with appearance (0.53–0.69), indicating that a rendered video can look plausible while its underlying motion is inaccurate. Trajectory DTW is the most independent metric (0.36–0.47), since it uniquely requires the correct matter to follow the correct 3D path, which neither appearance nor per-frame motion statistics reveal.

D.8 ADDITIONAL METRICS AND GEOMETRY QUALITY

Figure A10 (left) reports five complementary metrics that broadly corroborate the primary benchmark rankings: (i) *TIPS similarity* is the same semantic similarity, computed with TIPSv2 (Cao et al., 2026)

embeddings in place of DINOv3; (ii) *GeoPhys DINOv3* evaluates temporal smoothness using five trajectory regularity descriptors from GeoPhys (Internò et al., 2026), scoring $1-\frac{1}{5}\sum_{\phi}|\phi-\hat{\phi}|/(\phi+\hat{\phi})$; (iii) *Uni3D point* and *Uni3D MoGe* compute cosine similarities of Uni3D (Zhou et al., 2024) embeddings between reconstruction frame-0 surfaces and, respectively, the registered reference world and a single-frame MoGe (Kong et al., 2026) point map; and (iv) *Occupancy DTW* evaluates dynamic matter sequence matching under sliced Wasserstein distance without requiring persistent particle identity.

	Real & Synthetic		Real only	Synthetic only		Geometry quality				
	TIPS	GeoPhys DINOv3	Uni3D MoGe	Uni3D point	Occupancy DTW	Water-tight	Mani-fold	Clean faces	No self-intersect.	No inter-penetr.
Proprietary models										
6-Astra [Max]	**0.972**	**0.888**	**0.845**	**0.890**	**0.763**	0.830	0.997	0.881	0.877	**0.956**
6-Astra [High]	**0.969**	0.865	**0.823**	**0.895**	**0.725**	0.832	**0.999**	0.874	0.901	0.939
Fable-5.1 [High]	0.948	**0.871**	0.795	0.880	0.715	0.944	0.996	0.965	0.956	0.953
6-Astra [Low]	0.963	0.838	0.811	0.878	0.680	0.888	**0.999**	0.900	0.943	0.928
Opus-5 [High]	0.946	0.857	0.809	0.836	0.689	0.923	**0.999**	0.925	0.938	0.950
5.6-Luna [Max]	0.948	0.827	0.768	0.830	0.592	0.911	0.995	0.905	0.931	0.934
5.6-Sol [High]	0.947	0.818	0.760	0.839	0.609	0.889	**1.000**	0.895	0.914	**0.959**
Gemini-3.8 [High]	0.936	0.819	0.750	0.822	0.583	0.914	0.994	0.914	0.932	0.915
5.6-Terra [High]	0.932	0.782	0.640	0.765	0.485	0.892	0.997	0.832	0.937	0.925
Open-weight models										
DeepSeek-V4.1 [High]	0.921	0.793	0.727	0.735	0.518	0.911	0.987	0.894	0.915	0.929
Qwen-3.8 [XHigh]	0.808	0.712	0.594	0.693	0.541	0.838	0.971	0.808	0.814	0.932
GLM-5.3 [Max]	0.863	0.666	0.559	0.608	0.468	0.834	0.945	0.788	0.805	0.935
Muse-Glimmer [High]	0.852	0.628	0.553	0.677	0.380	**0.971**	**1.000**	**0.986**	**0.991**	0.904
MiMo-V2.5 [Def]	0.870	0.691	0.512	0.576	0.393	0.947	0.992	0.936	0.933	0.851
MiniMax-M3 [Def]	0.661	0.548	0.409	0.439	0.293	0.936	0.990	0.949	0.967	0.912
Mistral-3.5	0.706	0.345	0.189	0.459	0.366	**0.964**	0.987	**0.970**	**0.969**	0.902
Gemma-4-31B	0.730	0.532	0.012	0.181	0.176	0.954	0.889	0.851	0.938	0.942

Figure A10: **Additional metrics and geometry quality.** The left table scores five further reconstruction metrics, counting failed runs at the penalty value, and the right table scores the geometry checks over the runs that deliver a world; higher is better, and the best and second best in each column are shaded. The geometry checks do not follow reconstruction quality: weaker models, which tend to build simple geometry, often score higher on them.

Geometry quality. Beyond visual and dynamic tracking, we evaluate the physical validity of the reconstructed meshes. Figure A11 illustrates common structural defects (e.g., self-intersections and non-manifold edges). In Figure A10 (right), we report the fraction of mesh components across all frames that are *watertight*, *manifold*, *non-degenerate*, and *free of self-intersections*, alongside *no interpenetration* (1− vertex penetration ratio). These structural checks evaluate only submissions that successfully deliver a valid 4D world (executability rates in Figure **??**).

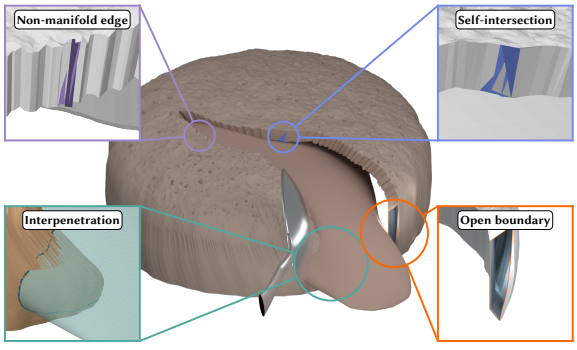

Figure A11: **Examples of geometry defects** on a reconstruction of a spoon splitting a lava cake at frame 150; each circle is magnified in the inset of the same colour.

Notably, mesh quality does not correlate positively with reconstruction fidelity. For instance, Muse Glimmer [High] leads or ties on all four mesh integrity checks, whereas Astra [Max]—the top-performing reconstruction model—achieves only 0.830 watertightness. This divergence arises because stronger models attempt to reconstruct highly intricate geometry (such as fractured fragments, thin shells, and fluid surfaces)

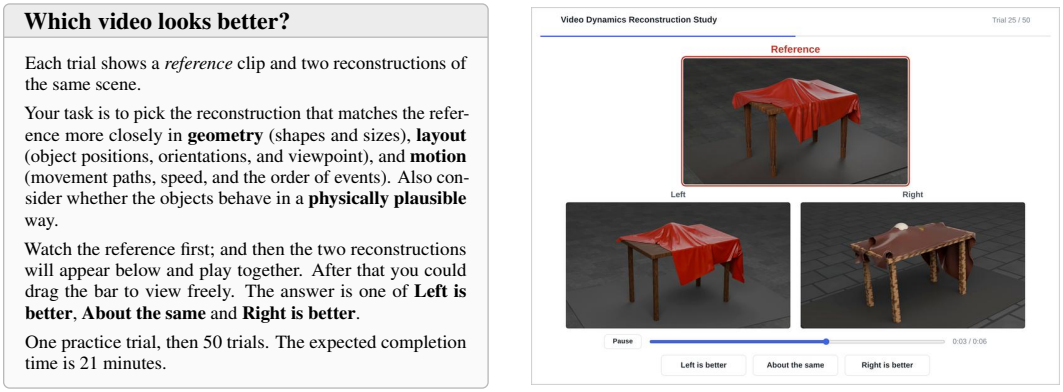

Figure A12: **Human evaluation interface.** *Left:* Instructions provided to participants outlining criteria for geometry, layout, motion, and physical plausibility. *Right:* The side-by-side trial interface showing the reference video above two candidate reconstructions, equipped with synchronized scrub bars and ternary voting buttons (*left*, *right*, *about the same*).

that inherently poses severe meshing challenges, whereas weaker models generate simple geometric primitives that pass structural tests effortlessly. Interpenetration remains uniformly low across all models (0.851–0.959).

E Human Study

Protocol. We conduct a side-by-side user study to measure which reconstruction people judge closer to the reference. In each trial, participants view the reference video alongside rendered outputs from two randomly ordered models and, after replaying and scrubbing, select the one that matches the reference more closely in geometry, layout, and motion (*left*, *right*, or *about the same*). A session contains 49 comparisons on distinct scenes, balanced between real and synthetic splits, plus a practice trial and one comparison repeated with the sides swapped to evaluate self-consistency; participants give the same answer on 95.5% of the 67 repeated comparisons. Figure A12 shows the instructions and the interface.

Sampling. With 17 models, uniform sampling would waste judgments on pairs whose relative ranking is already certain. To concentrate the budget where rankings remain ambiguous, each trial selects the least-evaluated scene and samples model pair (a, b) from its candidate pool by expected confidence interval shrinkage (Chiang et al., 2024):

$$p_{ab} \propto \sqrt{\hat{v}_{ab}/n_{ab}} - \sqrt{\hat{v}_{ab}/(n_{ab}+1)}, \tag{10}$$

where n_{ab} and $\hat{v}_{ab}$ are the number and variance of the pair's past outcomes.

Ratings. We fit a Bradley–Terry model (Bradley & Terry, 1952) on the Elo scale:

$$P(a \succ b) = \frac{1}{1 + 10^{-(R_a - R_b)/400}}, \tag{11}$$

via weighted maximum likelihood, applying importance weights $1/(K\, p_{ab})$ to correct for non-uniform sampling (K denoting candidate pairs per scene) and counting ties as half-wins. Confidence intervals are obtained via 100 bootstrap resamples. Human Elo ratings correlate strongly with our aggregate automated ranking, the Overall score of Figure 5 (Spearman's $\rho = 0.96$).

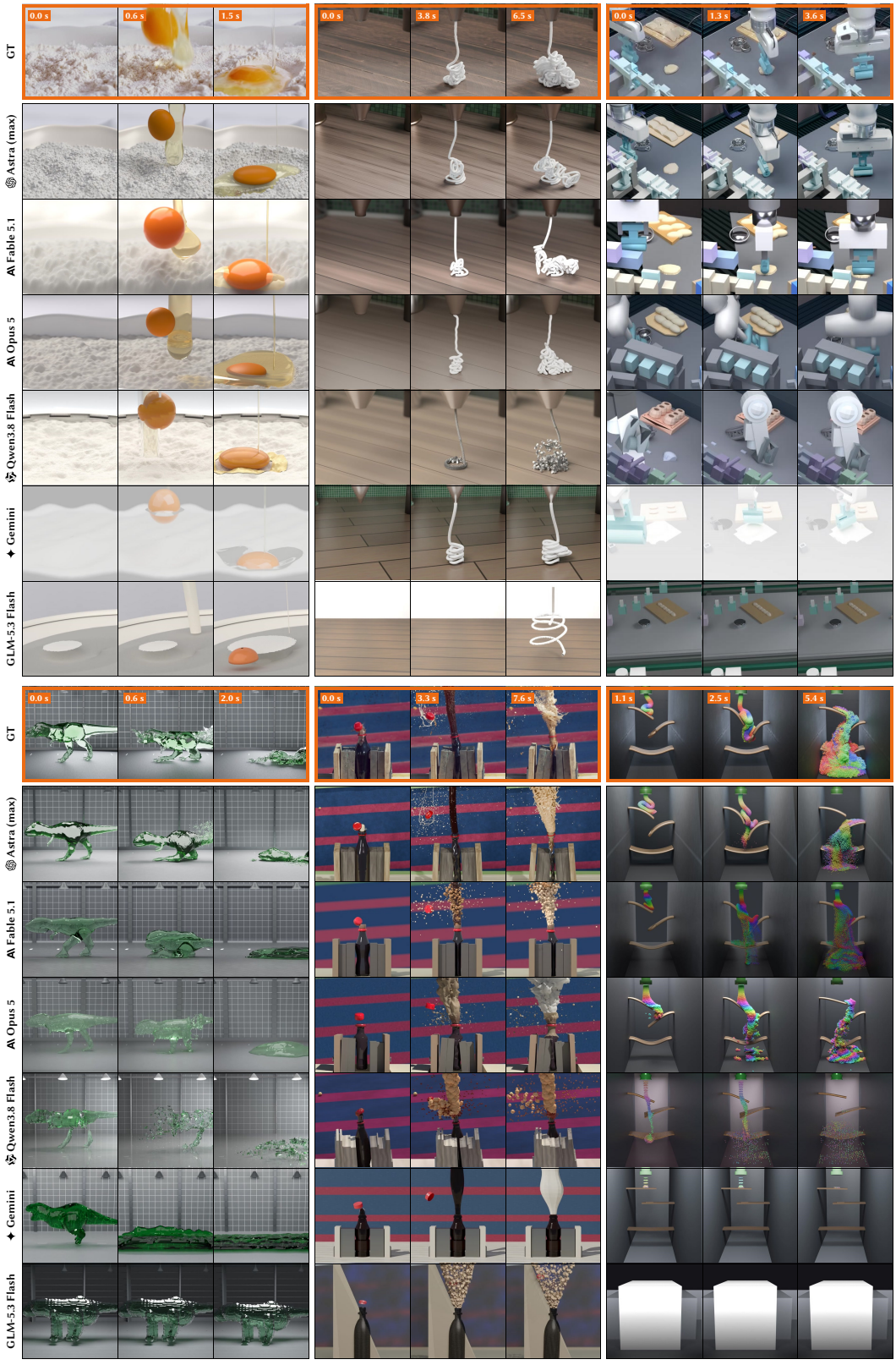

Figure A13: **Model Results.** This figure shows additional model results of 6 models on 6 different 4DCODEBENCH tasks.

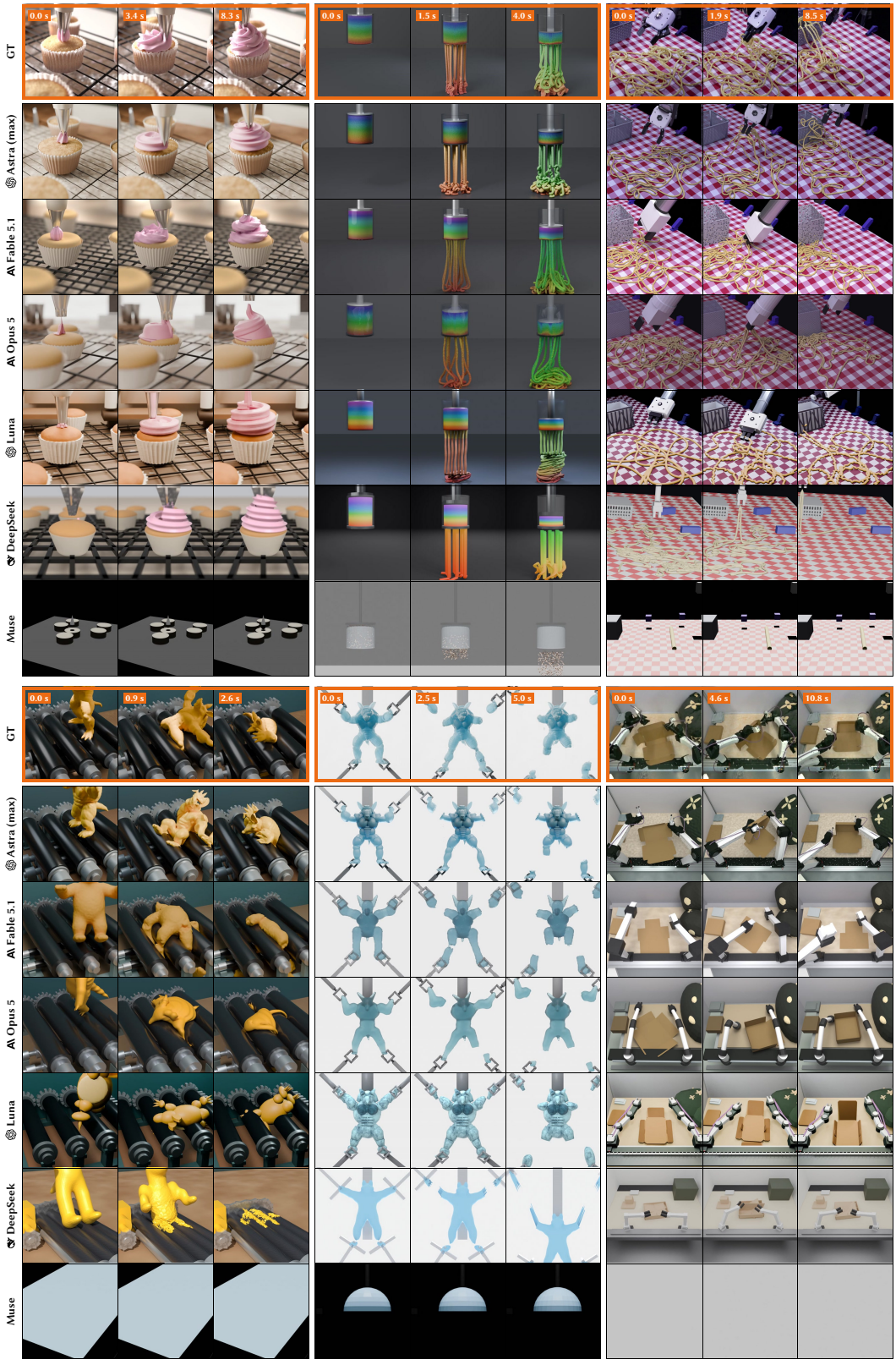

Figure A14: **Model Results.** This figure shows additional model results of 6 models on 6 different 4DCODEBENCH tasks.

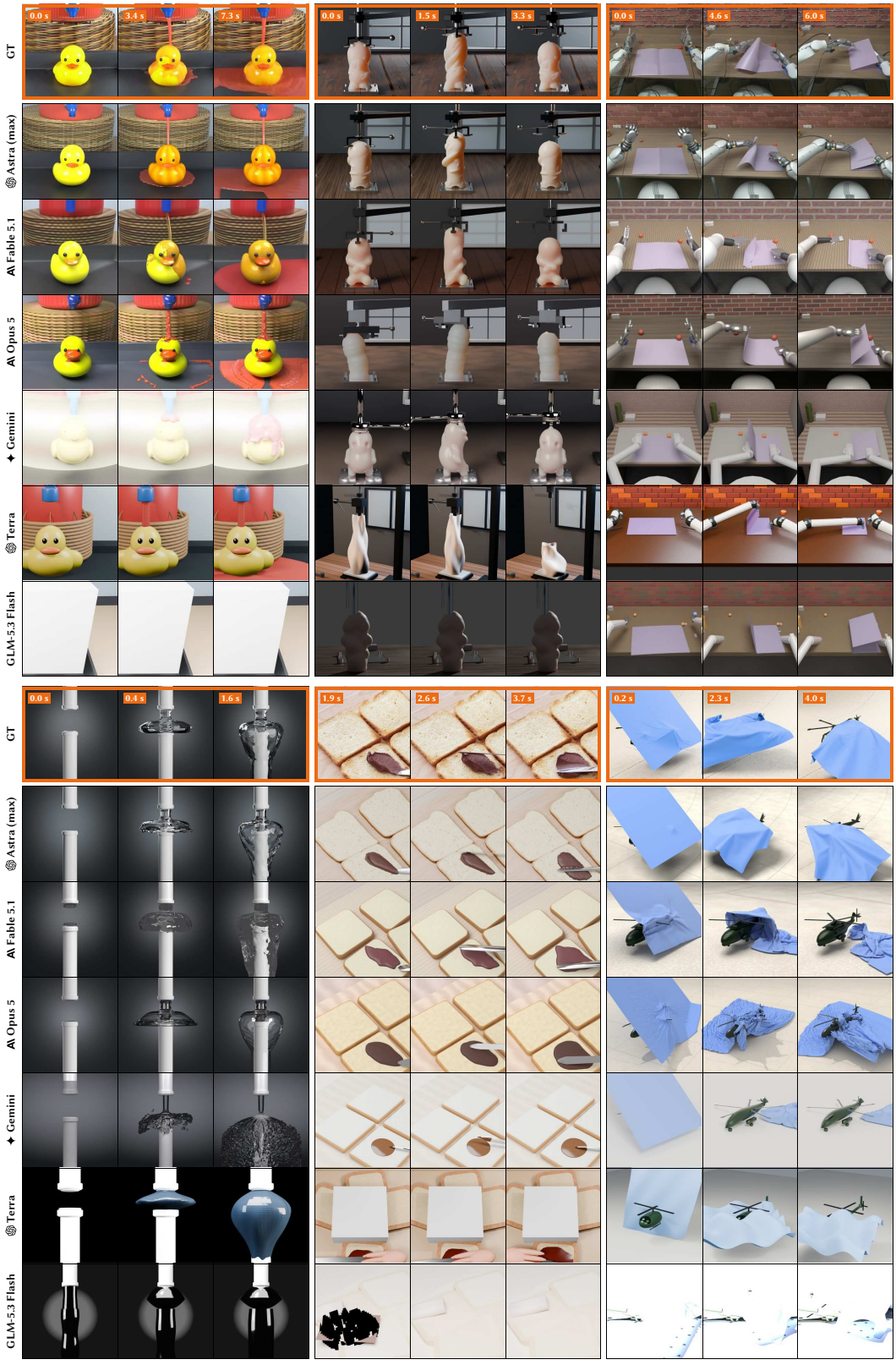

Figure A15: **Model Results.** This figure shows additional model results of 6 models on 6 different 4DCODEBENCH tasks.

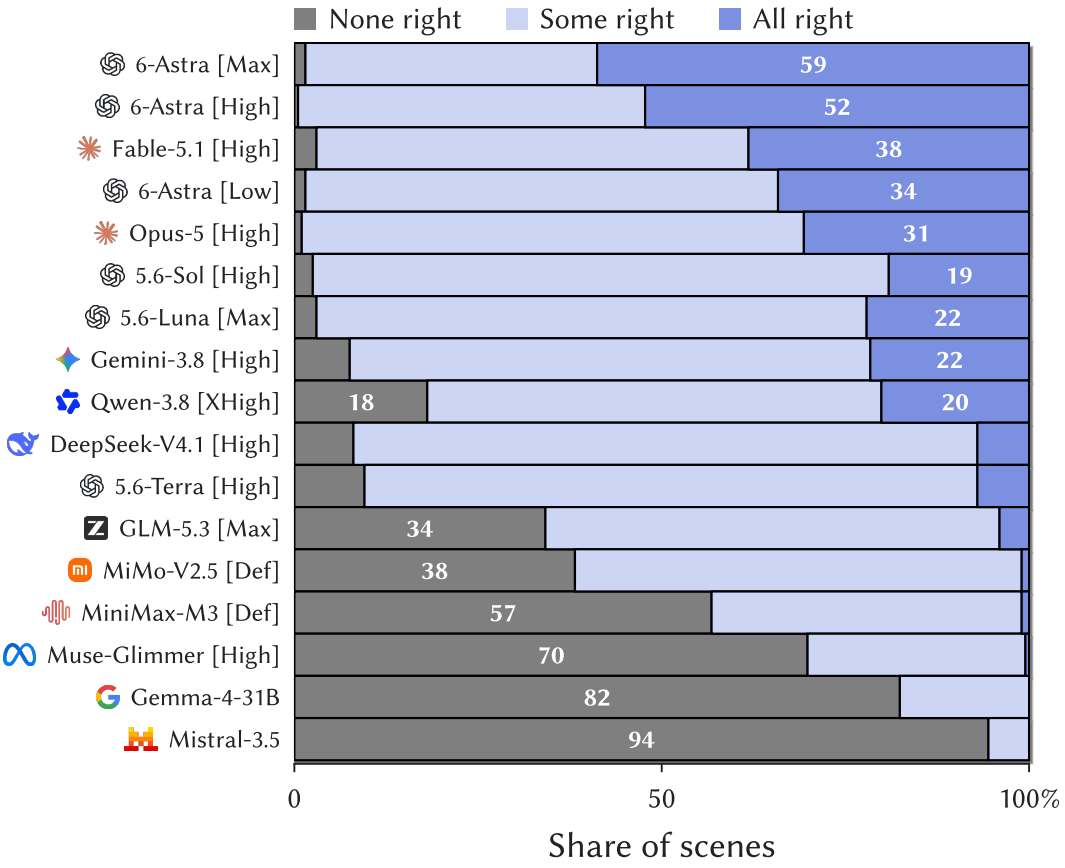

Figure A16: **Scene-level VQA.** Each bar divides a model's scenes by whether it answers none, some, or all of their questions correctly.

F Additional Results

F.1 Sample Agent Results

F.2 VQA Results

Figure A16 splits each model's scenes by how many of their VQA questions it answers correctly. The strongest models answer every question on a majority of their scenes (Astra [Max]: 59%), whereas the weakest open-weight models answer none on most (Mistral: 94%). Figure A17 shows the same split as a distribution over scenes for proprietary and open-weight models, and Figure A18 breaks VQA accuracy down by question category. Figure A19 lists every metric behind the main leaderboard, and Figure A20 splits the static–dynamic gap by space.

F.3 How do models describe motion?

Table A4 shows breakdown of how the models attempted to solve the 4DCodeBench tasks. Blender physics means the scene is carried by Blender's own solvers, mostly point-cache bakes (176 solutions), the rigid-body world (83) and the cloth modifier (34), with Mantaflow used three times; custom simulations are written in the solution itself, 209 of them identifying an established method (MPM or MLS-MPM 119, PBD or XPBD 96, DEM 21, PBF 15, mass-spring 12, FLIP 8, APIC 7, FEM 6, SPH 6, IPC 1) and the other 296 integrating an unnamed scheme, which we detect from explicit time stepping (209), GPU kernels (38), neighbour queries for contact (30) or impulse exchange between bodies (11); 91 run on taichi and 69 on Warp. Analytic solutions instead evaluate a closed-form function of time. Examples include interpolating splines through chosen poses (484), a prescribed map that displaces every vertex (293), ballistic formulas with gravity written in (237),

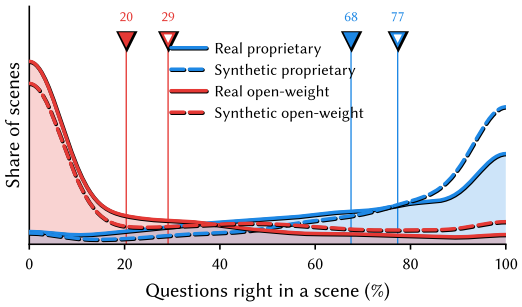

Figure A17: **VQA score distribution.** Density of the proportion of a scene's questions answered correctly over (model, scene) pairs, for proprietary and open-weight models on real and synthetic scenes; triangles mark the means.

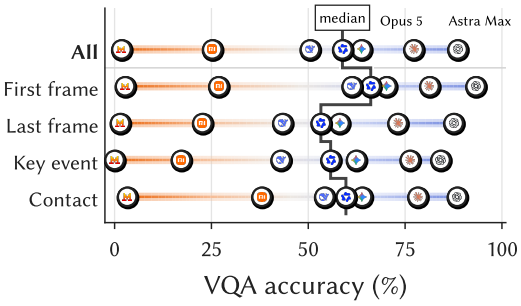

Figure A18: **VQA by question category.** Each row spans the accuracy of all models on one category, from the worst to the best; badges mark selected models, and the stepped line connects the medians.

	VQA	Perceptual	2D Dynamics			2.5D Geometry	3D Geometry	3D Dynamics		Overall
		DINOv3	Flow	Track2D	Dyn. IoU	Depth	Scene 3D	Traj. DTW	EMD step	
Proprietary models										
6-Astra [Max]	**0.876**	**0.906**	**0.486**	**0.597**	**0.716**	**0.791**	**0.922**	**0.810**	**0.659**	**0.791**
6-Astra [High]	**0.851**	**0.897**	0.443	**0.536**	**0.676**	**0.770**	**0.919**	**0.773**	0.605	**0.765**
Fable-5.1 [High]	0.788	0.858	**0.461**	0.492	0.602	0.742	0.902	0.758	**0.612**	**0.741**
6-Astra [Low]	0.782	0.878	0.407	0.466	0.563	0.751	0.898	0.723	0.530	**0.726**
Opus-5 [High]	0.765	0.851	0.435	0.477	0.606	0.732	0.861	0.725	0.547	**0.717**
5.6-Luna [Max]	0.672	0.834	0.365	0.426	0.468	0.700	0.861	0.655	0.410	**0.669**
5.6-Sol [High]	0.675	0.833	0.326	0.367	0.376	0.631	0.845	0.668	0.453	**0.645**
Gemini-3.8 [High]	0.627	0.809	0.332	0.359	0.398	0.627	0.840	0.653	0.462	**0.639**
5.6-Terra [High]	0.483	0.794	0.202	0.281	0.204	0.523	0.772	0.553	0.317	**0.550**
Open-weight models										
DeepSeek-V4.1 [High]	0.501	0.779	0.287	0.355	0.371	0.617	0.769	0.572	0.385	**0.596**
Qwen-3.8 [XHigh]	0.580	0.707	0.284	0.322	0.425	0.554	0.714	0.576	0.415	**0.563**
GLM-5.3 [Max]	0.296	0.691	0.169	0.211	0.209	0.514	0.601	0.505	0.224	**0.473**
Muse-Glimmer [High]	0.109	0.611	0.085	0.196	0.056	0.473	0.684	0.438	0.218	**0.442**
MiMo-V2.5 [Def]	0.247	0.690	0.091	0.164	0.091	0.483	0.571	0.455	0.174	**0.435**
MiniMax-M3 [Def]	0.179	0.525	0.121	0.127	0.093	0.355	0.445	0.339	0.178	**0.339**
Mistral-3.5	0.013	0.461	0.011	0.026	0.034	0.411	0.422	0.433	0.084	**0.315**
Gemma-4-31B	0.056	0.512	0.002	0.000	0.011	0.132	0.166	0.191	0.061	**0.188**

Figure A19: **Expanded leaderboard.** The individual metrics behind each family score of Figure 5, grouped by family; higher is better, and the best and second best in each column are shaded.

placement along a curve by arc length (218), hand-written pose tables (126) and and piles rebuilt each frame to keep their volume, as a height field or a cone at a fixed slope (91).

On the three cases in Figure 1, the choice follows the phenomenon. Every model bakes the dam break with Blender's Mantaflow FLIP solver, and almost all write position-based rod simulations for the pasta strands, with Fable 5.1 and Opus 5 using PBD and XPBD with kd-tree contacts and DeepSeek v4.1 Flash writing taichi kernels. The bread tear separates them, since Fable 5.1 runs two-field MLS-MPM with a per-particle fracture threshold, DeepSeek v4.1 Flash snaps XPBD ligaments across the seam, and Astra [Max] writes no solver and instead advances a cohesive fracture front down the loaf analytically.

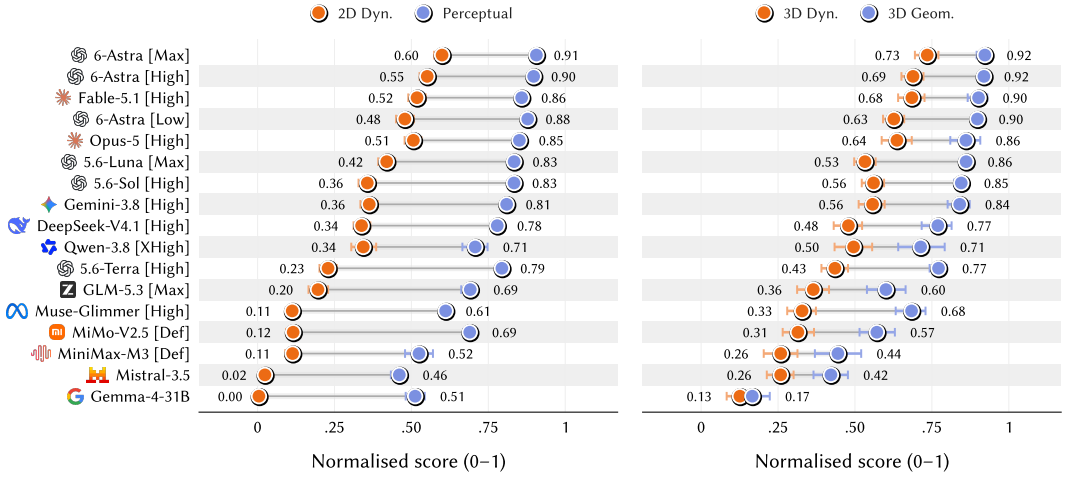

Figure A20: **Static–dynamic gap by space.** The pooled gap of Figure 8 separated into Perceptual against 2D Dynamics (left) and 3D Geometry against 3D Dynamics (right), with 95% intervals over scenes.

Table A4: How models construct motion. Each solution either uses Blender physics, writes a custom simulation, computes motion analytically, or keyframes poses. Results cover 3,172 solutions from 16 models; 19 results use other approaches, so some rows do not sum to 100%. The final row pools all solutions.

Model [Reasoning effort]	Blender	Custom sim.	Analytic	Keyframed
Claude Opus 5 [High]	16%	**51%**	25%	7%
Qwen3.8 Flash [XHigh]	6%	45%	45%	4%
Claude Fable 5.1 [High]	22%	44%	27%	7%
DeepSeek v4.1 Flash [High]	10%	25%	56%	7%
GPT-6 Astra [Max]	8%	**24%**	**63%**	5%
GPT-6 Astra [High]	8%	14%	73%	5%
MiMo v2.5 [Def]	13%	14%	70%	3%
Gemini 3.8 Flash [High]	28%	12%	49%	11%
Mistral Medium 3.5	8%	9%	83%	0%
MiniMax M3 [Def]	15%	8%	74%	1%
GPT-6 Astra [Low]	7%	6%	85%	3%
GPT-5.6 Sol [High]	5%	5%	88%	1%
GPT-5.6 Luna [Max]	0%	5%	93%	2%
Gemma-4 31B	4%	2%	94%	0%
Muse Glimmer [High]	5%	1%	92%	1%
GPT-5.6 Terra [High]	0%	**0%**	**100%**	0%
Average	9%	16%	71%	3%

G EXPERIMENTAL SETUP

This section details the agent execution environment (Appendix G.1), the task prompt (Appendix G.2), the unified 4D world submission specification (Appendix G.3), and aggregate compute statistics (Appendix G.4).

G.1 EXECUTION ENVIRONMENT

To evaluate each model with its intended harness, proprietary models run in their provider's native command-line agent: Codex CLI for GPT models, Claude Code for Claude models, and the Antigravity CLI for Gemini. All other models run in Stirrup (Artificial Analysis, 2025), an open agent loop driving an OpenAI-compatible endpoint with shell, image-viewing, and finish tools; these open-weight models are served via SGLang (Zheng et al., 2024) using recommended model configurations. Each agent maintains its native system prompt and receives the task prompt (Appendix G.2) as its initial instruction.

Each run executes in a fresh, isolated container mounting the reference video (`/input/reference.mp4`) and task context (`/task`) read-only, alongside an empty writable workspace (`/workspace`). The task directory provides the task prompt, world format specification (`world-format.md`), and render settings (`render-settings.md`). To prevent benchmark hacking and information leakage, all scene-specific metadata (e.g., scene names, descriptions, and reference 3D worlds) are stripped; operational parameters such as duration, resolution, and frame rate must be extracted directly from the video. Upon completion, only `world/` and `solution/` are collected for evaluation.

The container environment (CUDA 13.0, Ubuntu 24.04) provides Blender 5.2.0 (as both CLI and `bpy` module) with FFmpeg and ImageMagick for rendering and visual processing. For external geometry and dynamic computation, it pre-installs NumPy 2.3.1, SciPy 1.18.0, PyTorch 2.13.0, Taichi 1.7.4, Warp 1.17.0, Trimesh 5.0.0, and Pillow 11.3.0. It also includes an offline copy of the Blender 5.2 API reference and the public format checker (`python -m checker`), but not the evaluation scorer. We impose no artificial step limits; only runs that stall or exceed a generous 6 h timeout are terminated.

G.2 AGENT PROMPT

Task prompt (`TASK.md`)

```
`input.mp4` is a video of a dynamic scene.
Your task is to write graphics code that, when rendered, reproduces the observed 4D scene.

Your reconstruction should model the scene and its dynamics explicitly.
It should be both visually and physically consistent with the target.
Render the result in Blender, matching the input's dimensions, framerate, and camera.

## Scoring
Your solution will be scored frame-by-frame.
It will also be evaluated against the ground-truth 3D geometry.
Accordingly, solutions that represent motion or appearance in 2D or copy from the video
↪  instead of explicitly modeling the scene will be strongly penalized.
The autograder will fail submissions that copy pixels from the input video into the
↪  Blender scene for any purpose, such as to texture objects; all must be
↪  programmatically defined.

## Output
The output should contain two directories:
1. `/workspace/solution` is the program, an executable `build.sh`, and the Python it runs.
2. `/workspace/world` is its output, holding the camera, the render, the geometry at every
↪  frame, and the 3D state of the moving matter over time.

`build.sh` takes no arguments, does **not** read the video file, and writes all of
↪  `/workspace/world` from nothing.
Both are required to verify that the reconstruction is reproducible.

Run `python -m checker` to confirm that what you have written is in the correct format.
The packaged checker does not contain the privileged autograder.

## How to build the scene and its motion

One option is to do all of it in Blender: describe the 3D scene and its motion
```

````
through time there, and render that 4D sequence. You can also write external
graphics code or programs using the libraries provided to describe the motion of the scene
through time, and use Blender only to render the final 4D sequence. NumPy,
SciPy, PyTorch, Taichi, Warp and Trimesh are installed, and anything else on the
machine is fair game.

However the motion is computed, the evolution of the scene has to be rendered
with Blender, through the camera you export.

The rendered geometry in `meshes/` may change freely from frame to frame, vertex count and
topology included, as happens with liquids or fractures. `dynamics/` is where the
frame-to-frame correspondence lives: column `j` of `pos` is the same piece of
material at every frame, so `pos[t, j]` and `pos[t+1, j]` are that matter one
frame apart. Geometry rebuilt independently at each frame cannot give you that.

## Output format

The full format of `/workspace/world` is in `/task/world-format.md`.
It lists every file with its dtype and shape, the camera conventions, and what to return
↪  for a liquid.
Read it before you write your export.

### Program: `/workspace/solution`

`/workspace/solution` should contain an executable `build.sh` and the `.py` and `.sh` files
↪  it runs, in any layout.
Running `bash build.sh` should write the complete `/workspace/world`, replacing anything
↪  already there.
It cannot read the video, access the network, install packages, or load assets shipped
↪  alongside it.
Anyone with this environment should be able to rebuild your world by running it.

### World: `/workspace/world`

```
world/
  camera.json          {"intrinsics": K, "extrinsic": E}, the fixed camera
  render.mp4           the rendered video
  meshes/<id>.npz      each rendered object's triangles at every frame
  dynamics/<name>.npz  each piece of moving material's positions over time
  solver/              a liquid solver's own bake, if it keeps no particle identity
```

`meshes/` is the geometry that gets rendered, and its vertex count and topology may change
↪  from frame to frame.
`dynamics/` describes how the 3D geometry of the moving objects moves through time: `pos`
↪  has shape `(F, N, 3)`, and column `j` is the same piece of material at every frame.
Static objects need only a `meshes/` archive.
The points in `dynamics/` are the vertices of your simulation mesh for a rigid body, cloth
↪  or soft body, and the particles for MPM, SPH or sand.

All positions are in world-space metres, with Z up.
`render.mp4` must match `/input/reference.mp4` in frame count, frame rate and resolution,
↪  or it will not be scored.
Every file in `meshes/` and `dynamics/` uses the same `F` frames.
Each frame of `render.mp4` should show exactly the triangles in `meshes/` and nothing else.

### Environment

The environment has FFmpeg, FFprobe, ImageMagick, Python and Blender.
You can run Blender as the `blender` executable or import it in Python as `bpy`.
NumPy, SciPy, PyTorch, Taichi, Warp, Trimesh, Pillow and the checker are available in
↪  Python.
The Blender Python API reference is in `/opt/blender-docs`.
`/task/render-settings.md` has the render settings we expect.

### Checking your work

`python -m checker` reports every format error, and a world that fails it scores zero.
`python -m checker.preview` saves `/workspace/tmp/preview_0000.png`, with frame 0 of
↪  `render.mp4` next to your `meshes/` projected through `camera.json`.
If the two don't line up, your camera or meshes don't match what you rendered.
Neither tool checks whether your world matches the video.
Put intermediate files in `/workspace/tmp`, not in `world/` or `solution/`.
````

G.3 SUBMISSION FORMAT

To enable automated, unified evaluation directly from the true 4D state rather than relying solely on 2D video appearance, we require models to submit explicit 4D assets beyond the rendered

video. Specifically, each submission must provide an executable `build.sh` inside `solution/` that deterministically regenerates the explicit `world/` representation, ensuring full reproducibility. Crucially, this export specification is designed to be lightweight—typically requiring only about 100 lines of standard Python export code. In practice, format compliance poses little barrier for most models: among runs that pass the initial video gate, only 5.8% fail subsequent 4D world validation ($\leq 2.5\%$ across proprietary models and 3.5–7.5% across open-weight models). The one exception is Gemma 4 31B (Figure A7). For all other models, benchmark failures thus stem from 4D reasoning and simulation limits rather than formatting friction.

A submission consists of two directories: `solution/` contains the standalone program (`build.sh` and accompanying scripts) that deterministically regenerates the reconstruction without external network or local assets, while `world/` stores the output artifacts:

```
world/
  camera.json          {"intrinsics": K, "extrinsic": E}
  render.mp4           the rendered video
  meshes/<id>.npz      vertices_TTTT (V_t,3) float32, faces_TTTT (T_t,3) int32
  dynamics/<name>.npz  pos (F,N,3) float32, ids (M,) uint16; faces (T,3) or tets (K,4) int32
  solver/              the solver's bake, for an identity-less liquid only
```

All positions are in world-space metres ($+z$ up), and all frame-indexed data strictly align with the F frames of `render.mp4`.

Camera. `camera.json` defines a fixed pinhole camera shared across frames: a 3×3 intrinsic matrix K (principal point at image centre) and a 4×4 camera-to-world extrinsic transform E ($+x$ right, $+y$ down, $+z$ forward).

Meshes. `meshes/` contains one archive per rendered object, storing vertices and triangle indices per frame. Vertex count and topology may vary arbitrarily across time; objects absent from `dynamics/` are treated as static.

Dynamics. `dynamics/` tracks moving matter across the F frames. Column j of `pos` follows a persistent physical point, taking finite values over its lifespan and NaN otherwise. Meshes provide surface (`faces`) or tetrahedral (`tets`) connectivity for rigid/deformable bodies, or omit connectivity for particle systems (MPM, SPH, granular).

Liquids. Particle-based fluids export directly via `dynamics/`. For Eulerian solvers without persistent particle identity (e.g., Blender Mantaflow), raw simulation bakes are retained under `solver/` alongside a `manifest.json` (specifying domain coordinates, resolution, and time scale), from which particle trajectories are extracted at scoring time.

Render. `render.mp4` is an H.264 video rendered in Blender through `camera.json`. Each frame must display exclusively the geometry in `meshes/` (lighting and materials are left unconstrained), without copying pixels from the reference video.

Checks. To help agents catch low-level formatting or syntax errors during development and avoid accidental execution failures, we provide a public offline checker (`python -m checker`). At scoring time, the video gate enforces valid decoding and resolution/framerate parity, while the world gate verifies the presence and structural integrity of `camera.json`, `meshes/`, `dynamics/`, and `build.sh`.

G.4 Running Statistics

To quantify the computational and economic footprint of agentic 4D reconstruction, Figure A21 reports per-task execution statistics across all 3,400 runs, totaling 5,661 hours and $15,241 in compute (open-weight models priced at OpenRouter rates). Per task, open-weight models expend a median of 151 agent steps and 14.1M tokens compared to 64 steps and 5.2M tokens for proprietary models, yet cost less (median $0.78 vs. $3.68) due to lower token pricing.

Crucially, sheer agent loop activity does not inherently yield better 4D reconstructions across models: while per-task cost correlates moderately with the Overall score (Spearman $\rho = 0.66$), raw step

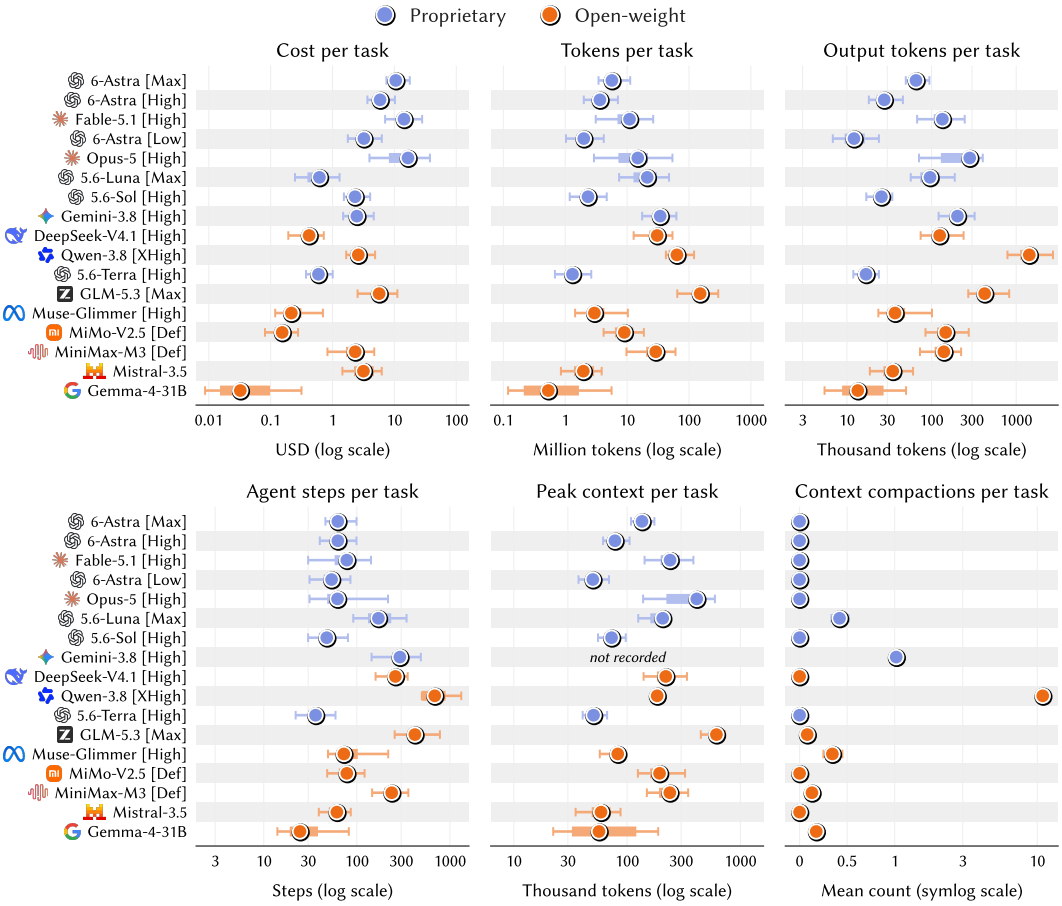

Figure A21: **Running statistics.** The first five panels show, over each model's runs, the median (dot), interquartile range (thick bar), and 5th–95th percentile range (whisker) of the cost, tokens, output tokens, agent steps, and peak context per task; the last panel shows the mean number of context compactions per task with its bootstrap 95% interval. The Antigravity CLI does not record peak context.

counts and token volume exhibit negligible correlation ($|\rho| \leq 0.13$). For example, Qwen [XHigh] expends an extreme median of 688 steps and 1.4M output tokens, yet ranks tenth among 17 models. Conversely, holding model architecture fixed, increasing test-time reasoning scales performance monotonically: raising GPT-6 Astra from [Low] to [High] to [Max] increases output tokens (12K $\rightarrow$ 28K $\rightarrow$ 66K) and cost (\$3.2 $\rightarrow$ \$10.5), steadily improving its Overall score from 0.726 to 0.791.