MirroS

# Code as Worlds:
## Agentic Discovery of Executable World Representations for Physical Reasoning

MirroS Technical Report


## Abstract

Physical understanding and reasoning depend on forming compact and generalizable representations of the world. While modern vision-language models can recognize and explain diverse physical events, they often lack explicit representations of the underlying mechanisms—such as object states, physical parameters, and governing dynamics—needed for reliably reasoning how the world evolves and responds to interventions. In this work, we introduce **Code-as-World**, a paradigm that represents physical worlds through executable world representations. By expressing physical composition, dynamic evolution, and visual appearance as executable code, Code-as-World provides a compact, quantitatively grounded, and controllable abstraction of the physical world. To construct such representations from multimodal observations, such as natural-language descriptions or real-world videos, we develop an agentic discovery loop inspired by abductive reasoning, where an agent proposes, executes, renders, verifies, and iteratively refines executable world hypotheses. As a concrete application, we use verified executable worlds to provide scalable physical supervision for training vision-language models on quantitative physical reasoning. Experiments show that Code-as-World-VL achieves state-of-the-art performance on QuantiPhy and surpasses leading proprietary models, highlighting the potential of executable world representations as a scalable foundation for physical intelligence.



**Date:** August 27, 2026

**Blog:** https://mirros.ai/blog/representing-physical-world

**Code:** https://github.com/mirros-lab/code-as-world

**Project Page:** https://mirros-lab.github.io/code-as-world

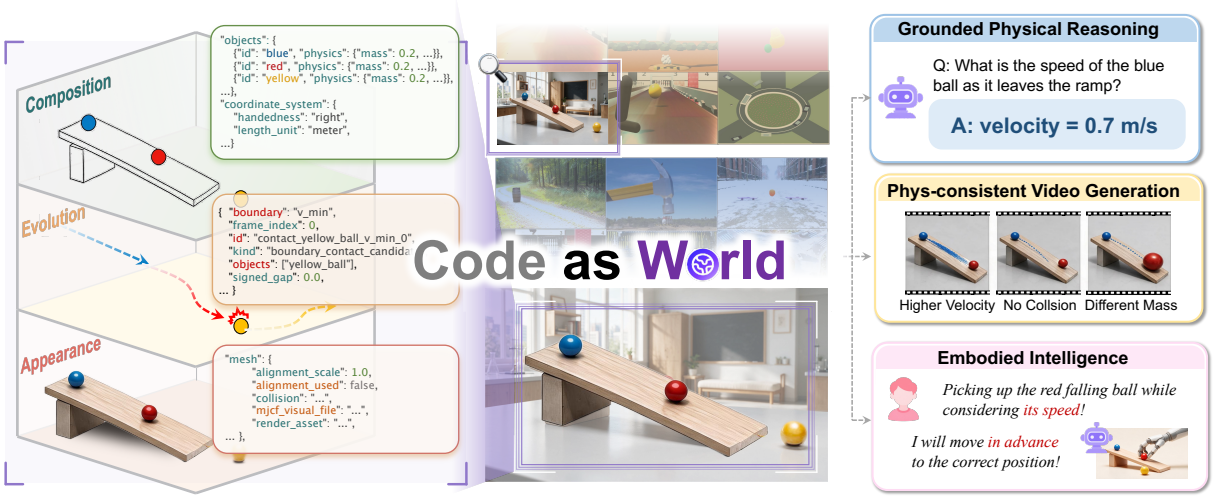




**Figure 1** **Code-as-World** represents physical worlds as executable code for physical intelligence.

# Contents

# 1 Introduction

*"All visible objects, man, are but as pasteboard masks."* — Herman Melville, *Moby-Dick*, Chapter 36

Understanding and reasoning about the physical world is a hallmark of intelligence [1]. Humans can make sense of unfamiliar physical situations and transfer knowledge across them, reasoning in terms of concepts such as objects, mass, motion, gravity, and friction rather than memorizing individual observations. *Generalization* is therefore a defining property of physical intelligence: knowledge acquired from one situation should remain useful when objects, configurations, or observations change. Such generalization calls for *representations* [2] that compress the complexity of sensory experience into a compact abstraction of what exists in the world, how it evolves, and which factors determine its behavior.

This raises a fundamental question: what forms of representation are needed to capture, understand, and reason about the physical world? Modern vision-language models [3–6] provide a powerful interface for describing the physical world. Trained on large-scale image–text and video–text data, they can recognize objects, narrate motion, and verbally explain a wide range of physical phenomena. Yet describing a phenomenon is not the same as recovering the mechanism that produces it [7]. Physical intelligence requires reasoning beyond what is visibly observed toward the structure of a system—its object states [8–10], physical parameters [11], governing dynamics [9, 10, 12], and responses to interventions [9, 13, 14]. This distinction gives rise to a fundamental *phenomenon–mechanism dichotomy*: phenomena describe what happens, whereas mechanisms explain why it happens and predict what would happen under different conditions.

In this work, we present **Code-as-World**, a novel paradigm for representing the physical world through code. Instead of representing a world solely through pixels [15, 16], latent features [17, 18], or natural-language descriptions [19, 20], Code-as-World expresses its task-relevant structure as executable code specifying physical composition, dynamic evolution, and visual appearance. Code offers a *compact yet quantitatively grounded abstraction* of the physical world: objects and their relations can be organized compositionally, while states, physical parameters, and governing dynamics remain explicitly represented. It therefore preserves the mechanistic structure needed for physical reasoning while abstracting away incidental details of individual observations.

Obtaining such a representation from incomplete observations, however, is fundamentally an inverse problem. From heliocentric theory to Newton's laws, scientific discovery has often proceeded through abductive reasoning [21]: recovering physical mechanisms from noisy and incomplete observations by searching for hypotheses that best explain the evidence while adhering to simplicity principles. Inspired by this process, we formulate world representation as an **agentic discovery loop** rather than a one-shot prediction problem [22]. The *executable and agent-native* nature of code enables an agent to treat each representation as a testable world hypothesis: Given multimodal evidence, such as natural-language descriptions or real videos, an agent proposes an executable world hypothesis, instantiates and executes it in a simulator, renders its predicted observations, and verifies them with the available evidence. Discrepancies are fed back to revise the hypothesis, forming a propose–instantiate–execute–render–verify loop that progressively searches for an executable world consistent with the observations.

As a concrete application, we use the resulting executable worlds as physical supervision for **quantitative physical reasoning** [11], where a VLM must infer measurable quantities such as object size, displacement, velocity, and acceleration from monocular videos. Unlike semantic physical question answering [10], these tasks require the model to connect visual evidence with metric states of the underlying world. We train Code-as-World-VL, a family of vision-language models, using measurement supervision together with physical supervision derived from verified executable worlds. The resulting models substantially improve quantitative physical reasoning: Code-as-World-VL-9B outperforms substantially larger models, including Gemini-3.1-Flash [3], on QuantiPhy [11]. Code-as-World-VL-27B further surpasses the 9B variant and all evaluated baselines. These results demonstrate that executable world representations can provide scalable supervision for grounding visual models in understanding physical mechanisms.

In summary, our contributions are as follows:

- **Executable world representation.** Based on a dedicated discussion among existing physical world representations, we introduce Code-as-World, which represents task-relevant physical worlds as executable code over physical composition, dynamic evolution, and visual appearance.
- **Agentic discovery loop.** We formulate the world representation process as an agentic discovery

problem, and develop a propose–instantiate–execute–render–verify loop that iteratively searches for world representations consistent with language or visual evidence.

- **Physical supervision for VLMs.** We use verified executable worlds to provide scalable supervision for quantitative physical reasoning. Code-as-World-VL achieves great improvements and attains state-of-the-art performance on QuantiPhy, outperforming leading proprietary models.

## 2 In Search of Physical World Representations

Physical understanding [1, 9–11, 23] ultimately depends on the representation through which a model encodes the world. Existing approaches have explored this question from several perspectives, including pixel-level prediction, geometric reconstruction, and language-based abstraction. These paradigms have each captured important aspects of the physical world, yet none alone provides a complete representation that simultaneously supports semantic understanding, structural composition, and temporal prediction.

**Pixels.** A natural approach to modeling the physical world is to learn directly from visual observations. Video generative models learn temporal dynamics by predicting future observations from past frames, achieving increasingly realistic and diverse generations [15, 16, 24–26]. However, observation prediction alone does not require a model to explicitly represent the underlying causes of visual changes. If a model only optimizes future pixel prediction, it does not need to distinguish whether a scene change is caused by camera motion or object motion, nor whether an object is temporarily occluded or has disappeared. Multiple contradictory internal explanations may therefore lead to similar predictive accuracy as long as they produce visually plausible futures.

This ambiguity limits physical reasoning. A physically meaningful representation must disentangle the factors that generate observations, including persistent object identity, physical state transitions, interactions, and viewpoint changes. A visually plausible future is not necessarily a physically correct future [27, 28].

**3D.** Another line of research focuses on reconstruction, including 3D reconstruction [29–32], inverse graphics [33–36], and dynamic scene representations [37–42]. By requiring models to recover geometry, viewpoint, and appearance, reconstruction-based approaches provide strong constraints on information preservation and achieve impressive capabilities in scene modeling.

However, reconstructability does not necessarily imply interpretability. A latent representation may faithfully reproduce an observation while entangling persistent structures, dynamic variables, and appearance factors. Recovering the 3D geometry of an object does not explain its physical behavior, such as why it falls after support is removed or how it interacts with other objects. Reconstruction preserves information, but does not automatically uncover the causal factors that govern the evolution of the world.

**Natural language.** Language provides another powerful abstraction of the physical world [43–45]. By converting visual observations into captions, descriptions, or reasoning traces, language-based representations capture high-level concepts that are compact, compositional, and transferable across contexts to support reasoning. For example, a description such as "a person picks up a cup" preserves entities, actions, and semantic relations while discarding irrelevant visual details.

However, language is inherently limited in expressing the continuous and quantitative aspects of physical states. Precise geometry, trajectories, contact relationships, and physical parameters are difficult to encode precisely through a small number of discrete tokens. Language provides an effective semantic interface, but not a complete physical state representation.

These perspectives reveal complementary strengths [46]: pixels preserve rich details, 3D representations preserve geometric structure, and language captures semantic abstractions. Yet physical intelligence may require a representation that better integrates these capabilities: semantic like language, structured like reconstruction, and capable of modeling temporal evolution like generative models. Such a representation should not merely describe or reproduce the world, but explicitly capture the entities, states, and mechanisms that generate observations.

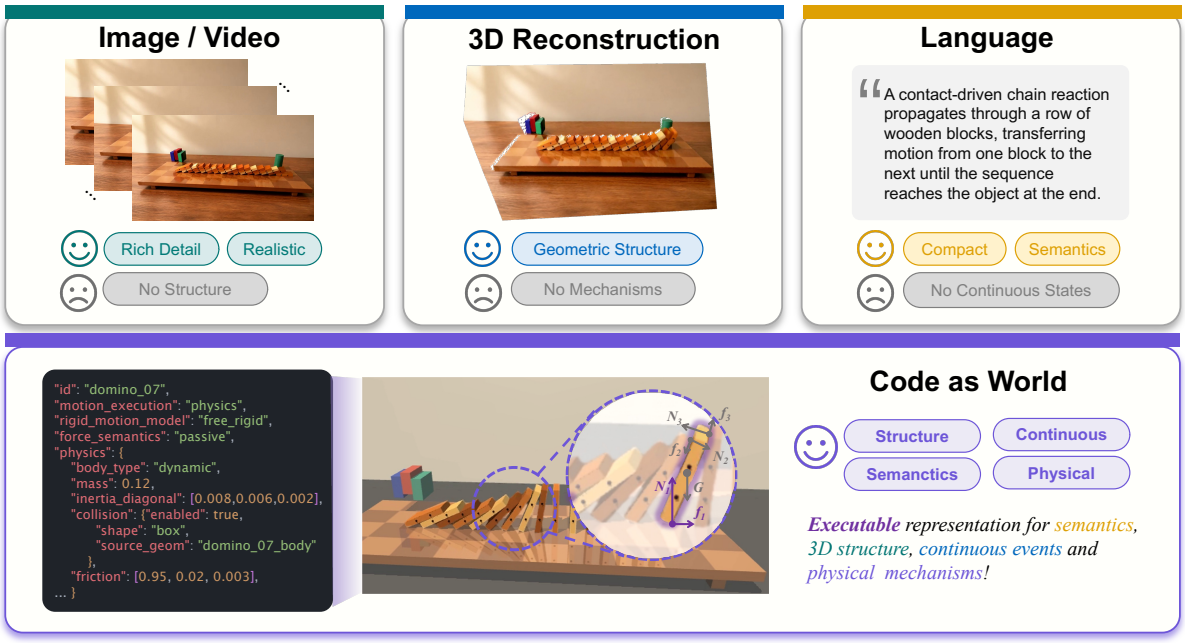



**Figure 2 Comparison among different physical world representations.** Pixels preserve rich visual detail but lack explicit structure, leaving the underlying causes of observations ambiguous; 3D representations capture geometric structure but do not necessarily reveal physical mechanisms; and language provides compact semantic abstractions but lacks precision for continuous physical states. Code complements these representations by expressing the world as an abstract, quantitatively grounded, and executable representation.

## 3 Code as Worlds: Executable World Representations

In this work, we introduce Code-as-World, a paradigm that represents the physical world through code.

**Code** provides a structured and compositional abstraction of the world: it explicitly represents entities, relations, physical parameters, and events as operations over states, while abstracting away incidental details of individual observations. Such a representation separates the structured state underlying physical processes from the continuous appearance through which they are observed. The former provides compositionality, editability, and explicit constraints for reasoning, while the latter preserves rich visual details and uncertainty that are difficult to explicitly encode. Through execution and rendering, code can generate observations consistent with the represented world, where physical equivalence—the consistency of world composition, constraints, and evolution—is prioritized over pixel-level duplication.

### 3.1 Practical Implementation

To instantiate this paradigm, we develop a concrete implementation of executable world representation (EWR) coupled with programmatic calls to a physical simulation engine. Figure 13 presents an example of the representation and its simulation interface.

Specifically, an EWR $p$ conceptually consists of three components:

$$p = (\mathcal{C}, \mathcal{E}, \mathcal{A}) \in \mathcal{P}_{\text{exec}}, \tag{1}$$

where $\mathcal{P}_{\text{exec}}$ denotes the space of valid executable worlds. Its three components jointly specify what the world contains, how it evolves, and how it appears.

**Physical composition.** $\mathcal{C}$ describes what exists in a world and the relatively stable physical properties of its entities. It includes objects in the scene, their geometry, metric dimensions, and physical properties such as mass, friction, and gravity. Environmental structures such as floors, tables, and walls are likewise represented as static physical entities, allowing them to participate in support, contact, and collision. This component defines the persistent entities involved in physical processes and the fundamental conditions under which they unfold.

**Dynamic evolution.** $\mathcal{E}$ describes how the world unfolds over time. It includes initial object states, their temporal changes, key events, and the duration of the simulation. Given the dynamic evolution component, the world composition can be expanded into a complete state trajectory, producing events such as contacts, collisions, velocity changes, and termination conditions during execution.

**Visual appearance.** $\mathcal{A}$ describes how the physical world is observed and presented. It includes camera parameters, backgrounds, materials, lighting, output frame rates, resolutions, and rendering or video-generation configurations. This component does not alter the underlying physical process but determines how the physical trajectory is visually presented. Environmental elements that participate in support or collision belong to physical composition, whereas backgrounds and appearance factors that provide only visual context belong to visual appearance.

This implementation exposes the represented world through an executable and controllable interface. Different components of an EWR can be inspected, modified, and executed independently: an object, physical parameter, dynamic condition, or camera setting can be changed while preserving the remaining structure. Such executable representations provide a foundation for downstream reasoning, simulation, verification, and data generation.

## 4 Agentic Discovery of World Representations

While Code-as-World provides a structured representation space for physical worlds, recovering such representations from partial and heterogeneous observations remains challenging. To address this challenge, Code-as-World formulates world representation as an agentic discovery process rather than a direct prediction problem. Given multimodal evidence, including text descriptions and real videos, the discovery agent constructs semantic or visual constraints and iteratively searches for an EWR consistent with the input evidence through a shared *propose–instantiate–execute–render–verify* loop. The resulting representation externalizes the underlying physical mechanism as an interface that an agent can directly query, intervene on, and verify, supporting physical data generation, quantitative supervision, verifiable reward construction, and broader downstream physical reasoning.

### 4.1 Evidence Construction

Given an input $\xi$ from modality $m$, the agent first constructs modality-specific evidence $\eta$ through a dedicated evidence adapter before entering the discovery loop. Text and video inputs are processed by different adapters, which transform them into semantic or visual evidence that constrains the same EWR space.

**Text-driven world construction.** The agent extracts explicit entities, spatial relations, physical events, and intended outcomes from the input text and organizes them as structured semantic evidence. Because textual descriptions rarely determine geometry, physical parameters, or camera configurations completely, the agent combines physical priors with reasonable default conditions to initialize an executable world hypothesis, which is progressively refined through simulation and semantic verification. For downstream applications as in Section 5, once the executable world is obtained, we apply a video generation model for sim-to-real transfer, enriching objects, materials, backgrounds, and lighting. The resulting videos achieve greater visual realism while remaining aligned with the underlying physical trajectories and world states.

**Video-driven world abstraction.** Given a real video, the agent extracts depth maps, instance masks, and object tracks as visual evidence for world construction and verification. Depth maps constrain the scene's spatial structure and relative distances, masks identify object boundaries and geometric extent, and tracks capture temporal correspondences and image-plane motion. For each segmented scene object, it further employs a 3D object generation model [47] to construct a corresponding mesh. It then combines these meshes with depth and tracking evidence to recover the objects' spatial positions, scales, and dynamic states, thereby constructing an executable 3D scene. The rollout of each candidate EWR is projected back into the input view and iteratively refined by evaluating its consistency with the observed geometry, depth, masks, and trajectories.

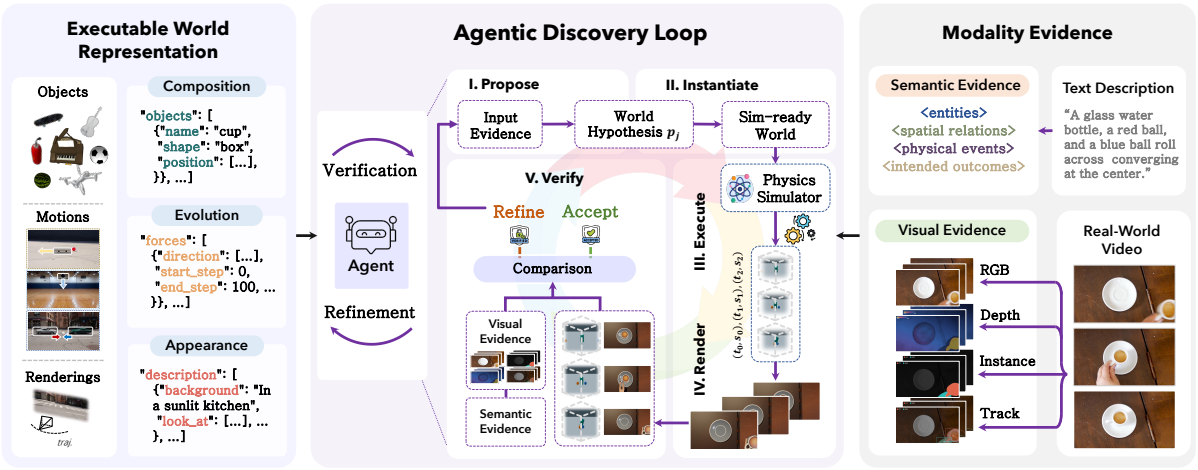



**Figure 3** **Agentic discovery loop of executable world representations.** Given a text prompt or real video, modality-specific processors transform the input into semantic or visual evidence. A shared propose–instantiate–execute–render–verify loop then optimizes an executable world representation over physical composition, dynamic evolution, and visual appearance.

## 4.2 Agentic Discovery Loop

Given modality-specific evidence $\eta$, the discovery agent proposes an initial EWR and progressively refines it through multiple iterations of the *propose–instantiate–execute–render–verify* loop, searching the executable hypothesis space for an EWR that best explains the input evidence while remaining as parsimonious as possible. Algorithm 1 summarizes the whole procedure of the agentic discovery loop.

At each iteration, the agent proposes or updates an EWR $p = (\mathcal{C}, \mathcal{E}, \mathcal{A})$ based on $\eta$, the current hypothesis, and structured feedback $\Delta$ from the previous iteration. The EWR is then instantiated as simulator-ready parameters $\theta$ that conform to a specific simulator interface, which the simulator executes to produce a complete state trajectory $\tau$. This trajectory explicitly records object states, contacts, collisions, and event outcomes, providing the world hypothesis with temporal and causal consequences that can be directly inspected.

The agent subsequently renders $\tau$ into predicted visual observations. For video input, it additionally projects the simulated states into depth maps, instance masks, and image-plane trajectories. During verification, the predicted and input evidence are compared at selected key frames. For text input, the verifier primarily evaluates semantic and physical constraints; for video input, it jointly compares RGB appearance, depth, masks, and trajectories. Frame-level discrepancies are aggregated into $\Delta$, which guides the agent in locally revising the relevant component in the next iteration. The loop terminates when the current EWR explains the input sufficiently well and parsimoniously. Otherwise, refinement continues until the iteration budget is exhausted, at which point the hypothesis is rejected.

## 4.3 Evaluation

### 4.3.1 Experimental Setup

**Data Sources and Filtering.** For text-driven construction, language specifications are generated by LLMs and subsequently reviewed by human annotators. For video-driven abstraction, candidate observations are selected from WISA-80K [48] through a motion-focused filtering pipeline. We first retain clips whose metadata indicates salient rigid-body motion or collision-like interactions and remove clips that mix unrelated physical phenomena. We then reject videos with substantial camera translation or rotation, insufficient object motion, severe editing, or incomplete physical events. The remaining clips undergo manual review for temporal continuity, object visibility, and suitability for executable reconstruction.

**Processing Tools.** For each retained video, SAM3 [49] supplies instance masks and image-plane tracks, VGGT-Omega [50] estimates scene depth and camera geometry, and SAM3D [47] provides object geometry. These observations constrain the agentic discovery loop, and a resulting world is retained only if it

**Algorithm 1** Agentic Discovery of Executable World Representations

**Require:** Input evidence $\xi$; LLM $A$; iteration budget $K$

**Notation.** $\eta$ denotes the preprocessed evidence. The current EWR is $p = (\mathcal{C}, \mathcal{E}, \mathcal{A})$, while $\theta$ denotes its simulator-ready instantiation and $\tau$ its executed world-state trajectory. The structured trace $z$ records the hypotheses and verification outcomes produced during discovery.

**Ensure:** An evidence-supported EWR $p$, or rejection, together with trace $z$

**if** $\xi$ is a text specification **then**
  $m \leftarrow$ text; $\eta \leftarrow \text{InterpretText}(A, \xi)$ ▷ Semantic evidence
**else if** $\xi$ is a video **then**
  $m \leftarrow$ video; $D \leftarrow \text{Depth}(\xi)$; $M \leftarrow \text{Segment}(\xi)$; $Q \leftarrow \text{Track}(\xi)$ ▷ Visual evidence
  $\eta \leftarrow (\xi, D, M, Q)$
**end if**
$p \leftarrow \emptyset$; $\Delta \leftarrow \emptyset$; $z \leftarrow \emptyset$
**for** $k = 1$ to $K$ **do**
  $p \leftarrow \text{ModifyEWR}(A, \eta, p, \Delta)$ ▷ Propose | Update
  $\theta \leftarrow \text{CompileEWR}(p)$ ▷ Instantiate
  $\tau \leftarrow \text{RunSimulation}(\theta)$ ▷ Execute
  $\hat{X} \leftarrow \text{Render}(\tau, \theta)$; $(\hat{D}, \hat{M}, \hat{Q}) \leftarrow \text{Project}(\tau, \theta)$ ▷ Render & Project
  $\hat{\eta} \leftarrow (\hat{X}, \hat{D}, \hat{M}, \hat{Q})$
  $F \leftarrow \text{SelectFrames}(m, \eta)$; $\Delta \leftarrow \emptyset$ ▷ Verify
  **for all** $f \in F$ **do**
    $\delta \leftarrow \text{CompareAndDiagnose}(A, \hat{\eta}[f], \eta)$
    $\Delta \leftarrow \Delta \oplus \delta$
  **end for**
  $z \leftarrow z \oplus \text{RecordRound}(k, p, \tau, \Delta)$
  **if** $\text{Accept}(A, p, \Delta)$ **then**
    **return** $(p, z)$
  **end if**
**end for**
**return** $(\text{Reject}, z)$

can be rendered and verified against the source video. Across both input modalities, Code-as-World uses the same simulation interface to produce synchronized videos and physical states.

**Evaluation Protocol and Metrics.** We evaluate video-driven world reconstruction for up to $K = 5$ agentic discovery rounds and compare iterative refinement with one-shot generation and Best-of-5 sampling under matched evaluation budgets. Visual Alignment and Object IoU [51] measure full-video visual agreement and object-region overlap between an input video and its simulator reconstruction. Motivated by [52, 53], we use Traj-ADE, Velocity-ADE, and Accuracy@2%$D$ to measure object-position error, inter-frame displacement error, and the percentage of valid observations within $0.02D$, respectively, with distances normalized by the frame diagonal $D$.

### 4.3.2 Qualitative Analysis

We evaluate the quality of data curated by Code-as-World from both text and video inputs. Figures 6 and 7 visualize representative inputs and outputs for the two modalities. These examples demonstrate that Code-as-World can both turn semantic descriptions into physically grounded videos and recover executable worlds from authentic visual observations.

Beyond the observed evidence, Code-as-World exposes recovered executable worlds as editable programs for controllable generation. As shown in Figure 4, we can resimulate a world after changing physical quantities or initial conditions. For example, changing the bowling ball's initial velocity direction produces distinct trajectories. We can also change the camera configuration to render the same collision from a global view or from either car. Our internal video generation model renders each edited rollout as a realistic video while preserving the specified physical evolution and viewpoint. Separating world editing from appearance synthesis enables coherent counterfactual videos without reconstructing every variant.

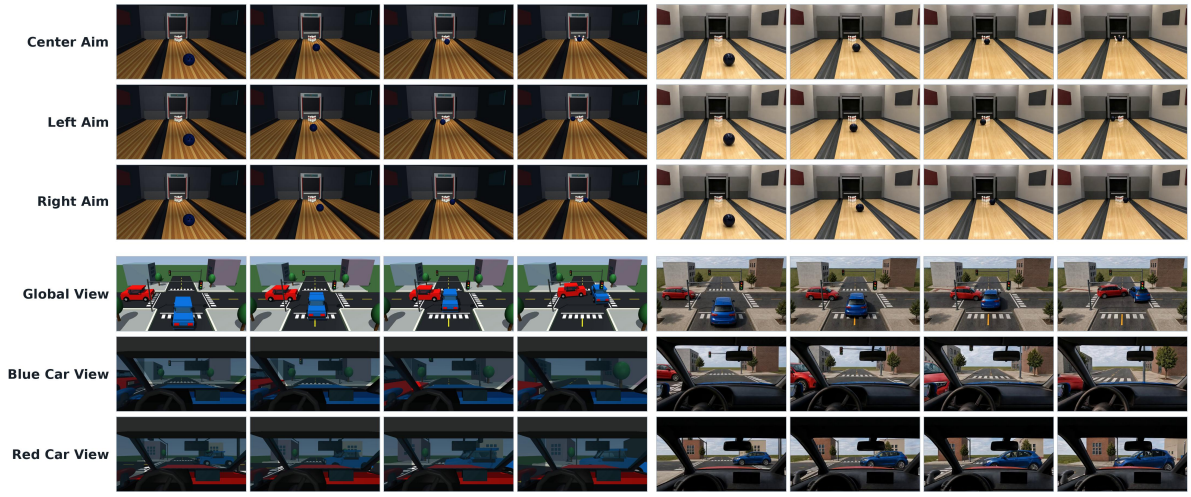



**Figure 4 Controllable resimulation with Code-as-World.** In each row, the left four columns show an edited simulator rollout, and the right four show temporally aligned frames from its realistic video.

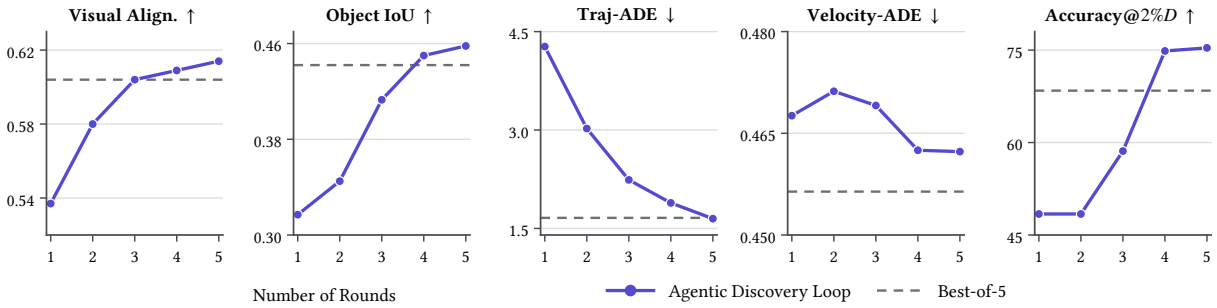



**Figure 5 Agentic discovery across loop rounds.** Solid curves connect one-shot to agentic loop rounds two through five. Dashed gray lines mark Best-of-5 from independent sampling. At a matched five-evaluation budget, agentic loop outperforms on Visual Alignment, Object IoU, Traj-ADE, and Accuracy@2%$D$. Here, $D$ denotes the frame diagonal. Traj-ADE is reported in %$D$, Velocity-ADE in %$D$/step, and accuracy at 2%$D$.

### 4.3.3 Agentic Discovery Loop

To evaluate the effectiveness of the agentic discovery loop, we set the maximum number of loop rounds to $K = 5$ and compare different rounds for video-driven world reconstruction. Candidate selection and refinement are driven by the verifier in Code-as-World, whereas the results are measured using independent metrics that are not included in the verification signal. All reported metrics compare the input video with the simulator rendering of the reconstructed EWR. Figure 5 plots each metric as a function of the loop rounds. Each solid curve begins with the one-shot result at round one and continues with four successive agentic discovery loop updates at rounds two through five. The dashed gray line marks the Best-of-5 reference. Across these refinements, static quality and most motion-fidelity metrics improve overall. At the matched five-evaluation budget, the agentic discovery loop outperforms Best-of-5 on most aspects, demonstrating more effective use of compute than independent sampling.

## 4.4 Limitations

Real-world environments are highly complex, and current simulators cannot faithfully capture the full range of physical conditions. Even seemingly simple rigid-body motion can be sensitive to small variations in terrain, contact geometry, material properties, or other latent factors. When the underlying process falls outside the simulator's modeling scope, the agentic discovery loop may converge to a locally plausible EWR without recovering a mechanistically accurate explanation of the physical process.

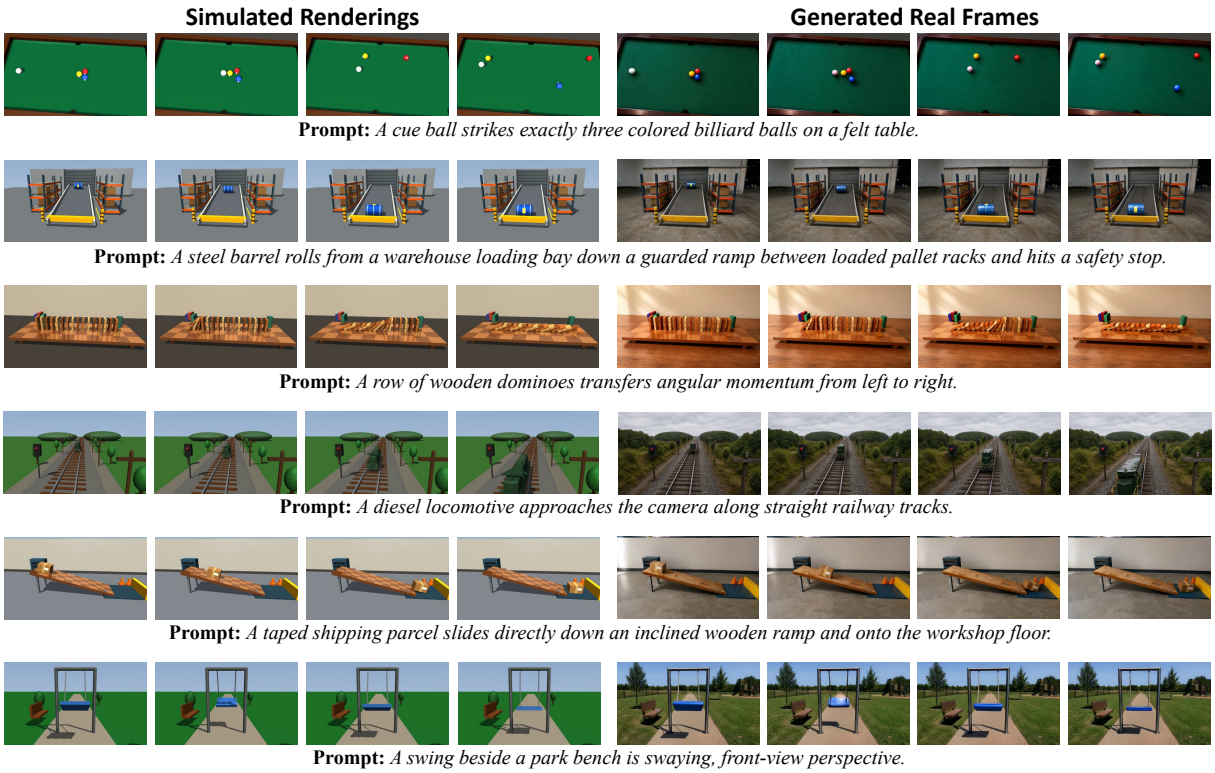



**Figure 6** **Visual results of text-driven construction**: from executable simulation to realistic video. For each example, the left side shows matched frames from a simulator rollout generated from a text specification, while the right side shows the corresponding final video synthesized by the video generation model. The sim-to-real transformation introduces rich and diverse objects, materials, backgrounds, and textures while preserving the motion trajectories and physical evolution encoded by the executable world.

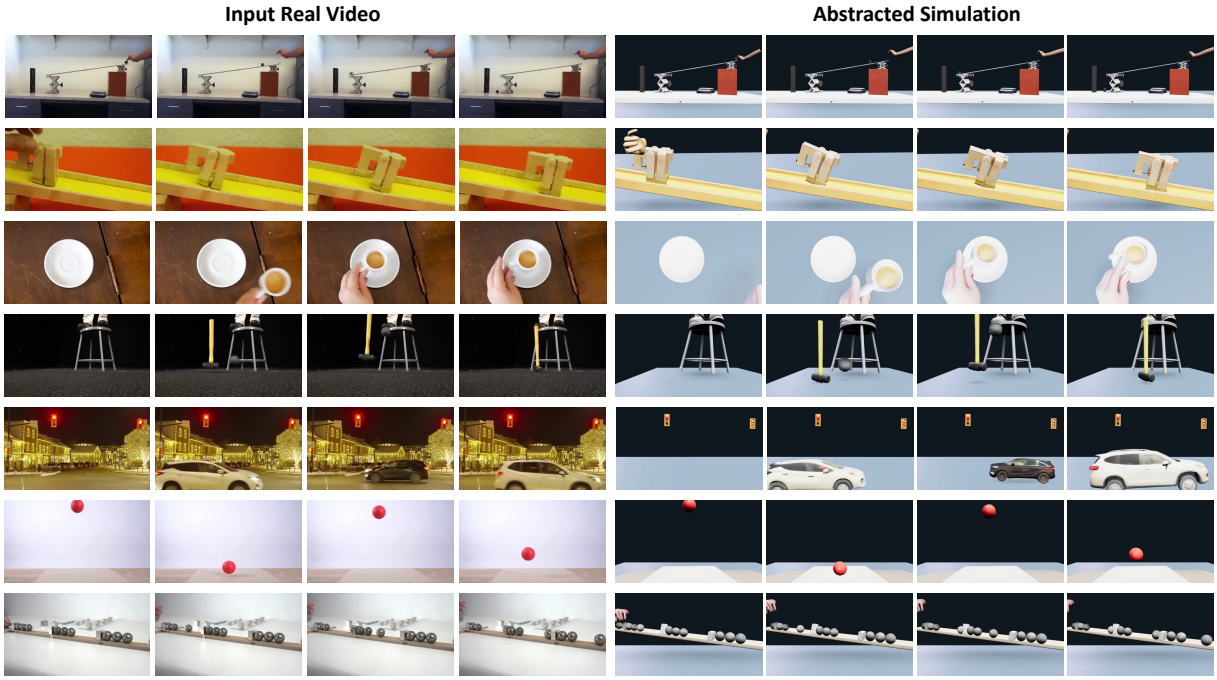



**Figure 7** **Visual results of video-driven abstraction**: from real video to executable simulation. For each example, the left side shows matched frames from the input real video, while the right side shows the corresponding rollout of the abstracted executable world. The recovered simulation aligns with the observed scene composition, object motion, and physical interactions, capturing the underlying physical mechanism as a coherent, executable process.

# 5 Application: Learning Quantitative Physical Reasoning

As a concrete downstream application of Code-as-World, we study quantitative physical reasoning in vision-language models [11]. Unlike semantic physical question answering, which can often be solved through visual patterns or commonsense priors, this task requires a VLM to infer physical quantities hidden beneath pixel observations in monocular videos, such as an object's real-world size, velocity, and acceleration. However, real-world videos rarely provide annotations of such underlying physical quantities. Code-as-World addresses this fundamental challenge by providing verified executable worlds that expose physical states and trajectories, enabling scalable physical supervision.

## 5.1 Problem Formulation

Given a monocular video $V$ and a quantitative physical question $q$, the model predicts a numerical answer $\hat{y} = f_\theta(V, q) \in \mathbb{R}$. The question specifies a target object, relevant timestamps, a queried quantity—such as size, velocity, or acceleration—and an output unit in either pixel space or world space. For world-space queries, the question additionally provides a reference quantity with a known world-space value $\rho$ to enable metric calibration.

The video provides measurements in pixel space but does not reveal their world-space scale. Let $y^{\text{pix}}$ and $\rho^{\text{pix}}$ denote the target and reference measurements in pixel units. Following QuantiPhy [11], the relative scale is estimated as $\gamma = \frac{\rho}{\rho^{\text{pix}}}$ and $y = \gamma y^{\text{pix}}$ is the ground-truth answer of the world-space quantity. The same calibration converts pixel measurements into world units for size, displacement, velocity, and acceleration. For 3D settings, depth information provides additional geometric cues that help relate image-space measurements to their corresponding world-space quantities.

## 5.2 Image-Space Measurement Grounding

Quantitative physical reasoning requires reliable image-space measurements as a foundation [54, 55]. We construct pixel-level supervision by converting bounding boxes, masks, and object tracks from existing visual datasets into quantitative question-answer pairs. These questions involve object extent, position, displacement, velocity, and acceleration, and require no world-space calibration. Specifically, for an object trajectory $\{\mathbf{c}_t\}_{t=1}^{T}$ sampled at interval $\Delta t$, displacement, velocity, and acceleration are computed as

$$\mathbf{d} = \mathbf{c}_{t_2} - \mathbf{c}_{t_1}, \qquad \mathbf{v}_t = \frac{\mathbf{c}_{t+1} - \mathbf{c}_{t-1}}{2\Delta t}, \qquad \mathbf{a}_t = \frac{\mathbf{c}_{t+1} - 2\mathbf{c}_t + \mathbf{c}_{t-1}}{\Delta t^2}, \tag{2}$$

where $t_1$ and $t_2$ are the timestamps specified by a displacement question. Object extent and position are read directly from bounding boxes or masks, while motion quantities are derived from tracks.

We train the model on the resulting pixel-level dataset $\mathcal{D}_{\text{pix}}$ with supervised fine-tuning:

$$\mathcal{L}_{\text{pix}} = -\mathbb{E}_{(V,q,y)\sim\mathcal{D}_{\text{pix}}} \log \pi_\theta(y \mid V, q). \tag{3}$$

This stage teaches the model to localize, measure, and track objects, providing the visual foundation for subsequent world-space physical reasoning.

## 5.3 World-Space Physical Calibration from Verified Executable Worlds

After image-space grounding, we further use verified executable worlds to construct physical supervision. Each EWR provides a synchronized video and simulated state trajectory that records object geometry, camera parameters, timestamps, and time-varying physical states. This enables direct generation of quantitative question-answer pairs with exact world-space labels. Specifically, we sample a target object, relevant timestamps, a physical quantity, and a requested world unit from this record, optionally providing a reference quantity with a known world-space value as a scale prior. The answer $y$ is obtained directly from the same EWR: object size is read from the scene geometry, while displacement, velocity, and acceleration are computed from the state trajectory. A question is retained only when its target object, temporal range, and optional reference are valid in the corresponding observation, yielding a VQA training instance $(V, q, y)$. Text-driven and video-driven executable worlds use the same format, forming $\mathcal{D}_{\text{text}}$ and $\mathcal{D}_{\text{video}}$, respectively, which we combine into unified world-level training data.

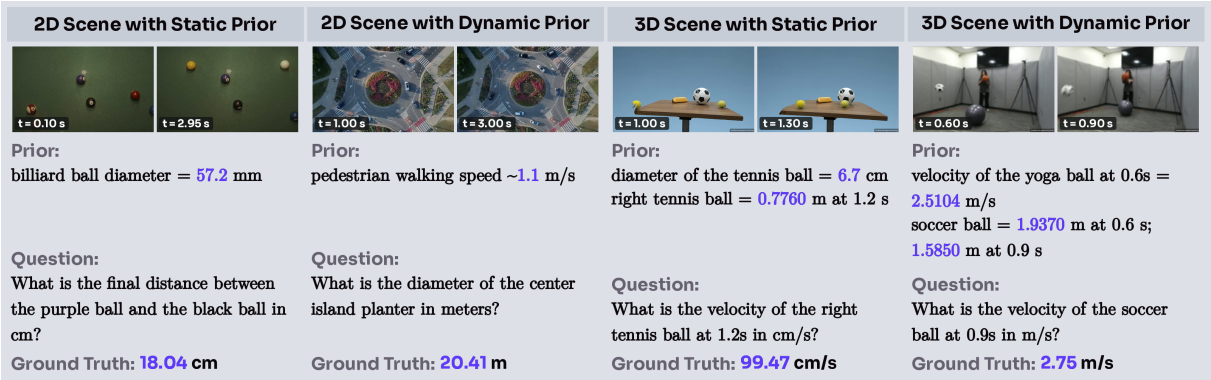



**Figure 8 Illustration of quantitative physical reasoning.** A video and a question are given as input. For 3D scenes, an additional depth prior provides spatial context. The model needs to predict an answer grounded in the video. Examples adapted from [11].

We optimize the model on these executable-world examples using Group Relative Policy Optimization (GRPO) [56, 57]. The reward combines scale-normalized numerical accuracy with auxiliary rewards for unit correctness and response format:

$$r_{\text{num}} = \exp\left(-\frac{|\hat{y} - y|}{|y| + \epsilon}\right), \qquad r = r_{\text{num}} + \lambda_u r_{\text{unit}} + \lambda_f r_{\text{fmt}}. \tag{4}$$

The two sources of executable-world supervision provide complementary benefits. Text-driven worlds offer fully observable simulator states and numerically exact physical supervision, while video-driven worlds better match the appearance and motion distributions of real observations. Joint training therefore combines accurate physical supervision with visual generalization to real-world videos.

## 5.4 Evaluation

### 5.4.1 Experimental Setup

**Models and Training.** We train controlled direct-answer variants of Code-as-World-VL at 4B and 9B using eight NVIDIA H100 GPUs and the two-phase curriculum described in Sec. 5. The first phase establishes image-space measurement through supervised fine-tuning; the second applies GRPO [57] to world-level VQA derived from text-driven and video-driven executable worlds. We refer to the checkpoints after the first phase as the *Image-Space* variants and to the checkpoints after both phases as Code-as-World-VL-4B and Code-as-World-VL-9B. To examine whether the framework extends to reasoning models at a larger scale, we additionally train Code-as-World-VL-27B (Reasoning), which produces a chain-of-thought reasoning before its final answer. During training and evaluation, all variants uniformly sample 16 temporally ordered frames from each video. Appendix B.5 gives the separate 27B protocol.

**Training and Evaluation Data.** Our image-space supervision is constructed from four referring-expression datasets—RefCOCO [54], RefCOCO+ [54], RefCOCOg [72], and RefCLEF [73]—together with GOT-10K [74]. The referring-expression datasets provide natural-language descriptions and ground-truth bounding boxes, from which we generate questions about the referred object's width, height, and diagonal length in raw pixels, as well as grounding questions requiring its bounding-box coordinates. GOT-10K provides dense object tracks over video, from which we construct questions about the target object's image-space extent, velocity, and acceleration at randomly sampled timestamps, together with timestamp-specific video grounding questions. Our world-level supervision contains text-driven and video-driven executable worlds produced by Code-as-World, from which we derive synchronized videos, physical states, and world-level VQA examples. We use the corresponding held-out datasets for pixel-level evaluation and the open-source QuantiPhy-validation set [11] for metric physical reasoning evaluation.

**Metrics.** For image-space measurement evaluation, we evaluate on five datasets: RefCOCO, RefCOCO+, RefCOCOg, RefCLEF, and GOT-10K. For each dataset, we compute Mean Relative Accuracy (MRA) [75] by comparing the model's pixel-valued numerical answers with the corresponding ground-truth pixel

**Table 1 Quantitative physical reasoning results.** We report MRA on the 2S, 2D, 3S, and 3D subsets and their macro-average on QuantiPhy-validation. The 4B and 9B variants produce direct answers, whereas the 27B reasoning variant is scored only on the answer emitted after its reasoning trace. Rows are grouped by model family, and Code-as-World-VL variants are highlighted in purple.

| Models | Size | Kinematic Categories | | | | Average Score |
|---|---|---|---|---|---|---|
| | | 2S | 2D | 3S | 3D | |
| *Proprietary models* | | | | | | |
| Gemini-3.1 Flash [3] | – | 49.4 | 47.5 | **61.4** | 61.1 | 54.8 |
| ChatGPT-5.1 [58] | – | **56.9** | 34.6 | 45.6 | 56.4 | 48.4 |
| Gemini-2.5 Pro [59] | – | 45.9 | 38.6 | 40.7 | 60.2 | 46.4 |
| Gemini-2.5 Flash [59] | – | 42.8 | 31.9 | 47.0 | 54.7 | 44.1 |
| Grok 4.1 (Fast Reasoning) [60] | – | 23.4 | 30.5 | 46.3 | 46.8 | 36.8 |
| ChatGPT-5 [61] | – | 32.2 | 24.6 | 35.6 | 38.1 | 32.6 |
| ChatGPT-5 Pro [61] | – | 16.2 | 22.7 | 20.2 | 18.9 | 19.5 |
| *Open-weight models* | | | | | | |
| Qwen3-VL-32B-Instruct [62] | 32B | 38.1 | 39.7 | 39.8 | 43.0 | 40.2 |
| InternVL-3.5-30B [63] | 30B | 33.1 | 33.0 | 31.4 | 44.7 | 35.5 |
| Qwen3-VL-8B-Instruct [62] | 8B | 17.2 | 27.6 | 36.0 | 48.3 | 32.3 |
| Qwen3.5-4B [6] | 4B | 26.6 | 35.7 | 19.8 | 41.3 | 31.2 |
| InternVL-3.5-8B [63] | 8B | 26.9 | 23.2 | 38.4 | 31.7 | 30.0 |
| Qwen3-VL-2B-Instruct [62] | 2B | 25.0 | 28.6 | 16.0 | 39.1 | 27.2 |
| Phi-4-Multimodal-Instruct [64] | 5.6B | 26.6 | 26.5 | 31.6 | 23.8 | 27.1 |
| SmolVLM-Instruct [65] | 0.26B | 30.0 | 21.4 | 22.8 | 33.0 | 26.8 |
| Qwen3.5-2B [6] | 2B | 28.7 | 29.7 | 17.9 | 25.9 | 25.6 |
| InternVL-3.5-2B [63] | 2B | 24.4 | 22.7 | 15.3 | 34.9 | 24.3 |
| Molmo-7B [66] | 7B | 13.8 | 20.3 | 19.1 | 41.7 | 23.7 |
| CogVLM2 Video [67] | 12B | 18.1 | 16.8 | 16.3 | 24.0 | 18.8 |
| VILA-7B [68] | 7B | 14.4 | 19.5 | 8.8 | 30.4 | 18.3 |
| Phi-3-Mini-128K-Instruct [69] | 3.8B | 14.7 | 12.4 | 21.6 | 19.6 | 17.1 |
| MiniCPM-V 4.5 [70] | 8B | 27.5 | 33.2 | 0.0 | 0.0 | 15.2 |
| LLaVA-13B [71] | 13B | 11.2 | 10.8 | 8.1 | 21.9 | 13.0 |
| *Ours* | | | | | | |
| **Code-as-World-VL-4B** | **4B** | 45.4 | 55.4 | 45.8 | 56.0 | 50.6 |
| **Code-as-World-VL-9B** | **9B** | 55.0 | 52.9 | 55.6 | 58.1 | 55.4 |
| **Code-as-World-VL-27B (Reasoning)** | **27B** | 48.7 | **62.4** | 60.5 | **62.8** | **58.6** |

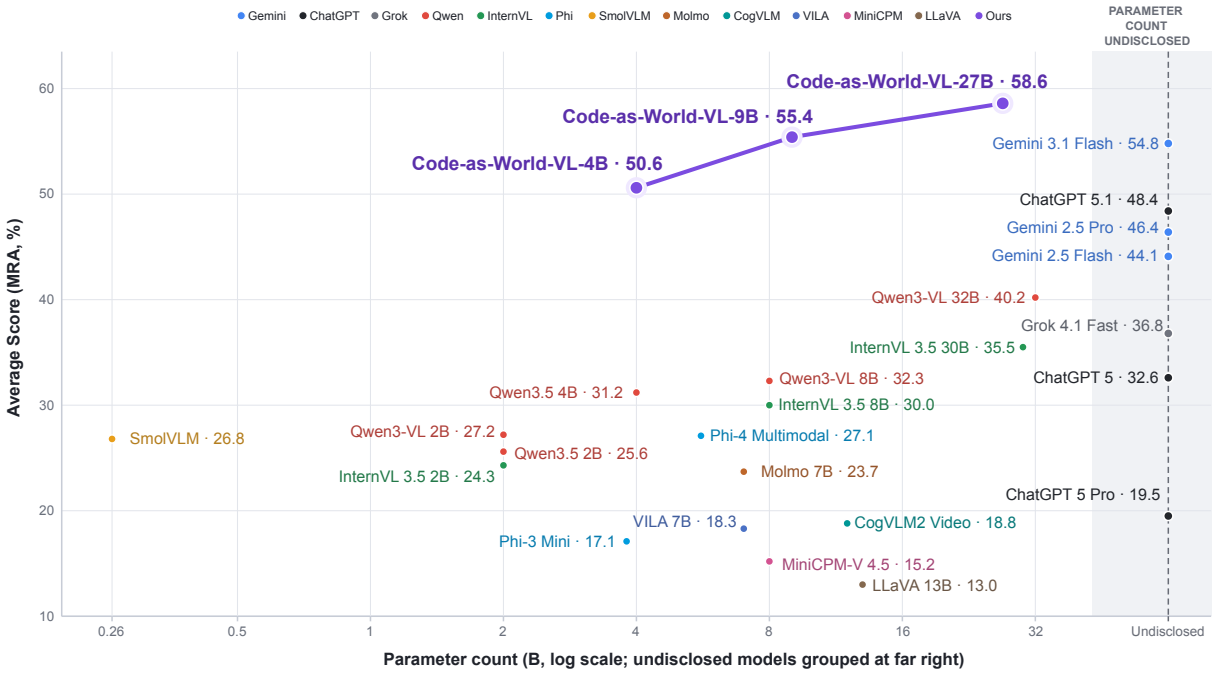
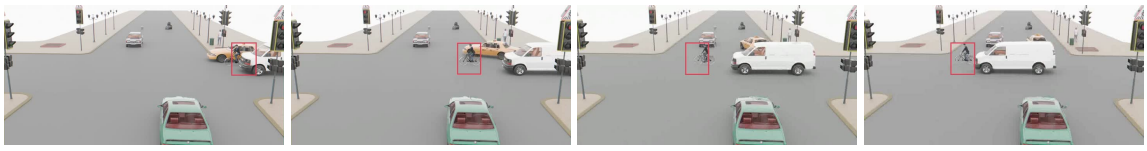



**Figure 9 Parameter scaling on QuantiPhy.** Models with undisclosed parameter counts are grouped at the far right.



$t = 0.1\,\text{s}$ $t = 0.9\,\text{s}$ $t = 1.7\,\text{s}$ $t = 2.1\,\text{s}$

**Prior:** Bicycle-camera distances are $d = 23.5534\,\text{m}$ at $0.58\,\text{s}$ and $d = 23.2262\,\text{m}$ at $1.58\,\text{s}$.
**Question:** What is the average speed of the bicycle in m/s?

<think> **(abridged).**
1. **Identify the motion.** The bicycle moves from right to left.
2. **Compute the radial component.** $v_r = |23.2262 - 23.5534|/(1.58 - 0.58) = 0.33\,\text{m/s}$.
3. **Track and calibrate.** The lateral track is about 210 pixels/s; mean depth $\bar{z} \approx 23.4\,\text{m}$ gives $v_\perp \approx 5.5$–$6.8\,\text{m/s}$.
4. **Check.** Since $v_r \ll v_\perp$, the total speed is about $6\,\text{m/s}$.
</think>

**Predicted Answer:** 6 m/s **GT:** 5.33 m/s **MRA:** 0.8

$t = 1.1\,\text{s}$ $t = 1.4\,\text{s}$ $t = 1.7\,\text{s}$ $t = 2.0\,\text{s}$

**Prior:** Billiard-ball diameter = 57.2 mm.
**Question:** What is the white ball's average velocity from 1.00 to 2.00 s in cm/s?

<think> **(abridged).**
1. **Locate the target.** Track the white ball (red boxes) across the table.
2. **Measure in Image-Space.** Its center displacement over 1.00 s is approximately 248 pixels; the ball spans about 53 pixels.
3. **Calibrate to World-Space.** The 57.2-mm diameter gives the local scale; perspective-normalized displacement yields $v \approx 26.8\,\text{cm/s}$.
4. **Check.** The direction and nearly constant track agree with the observed roll.
</think>

**Predicted Answer:** 26.8 cm/s **GT:** 26.82 cm/s **MRA:** 1.0

**Figure 10 Reasoning process of Code-as-World-VL-27B on QuantiPhy.** Two representative cases are shown vertically: an executable-world bicycle example and a real-video billiards example from `internet_0027`. Red boxes mark the target in the selected frames. The abridged traces preserve the key Image-Space measurements and World-Space calibration steps.

quantities. The four referring-expression datasets evaluate object extent, while GOT-10K evaluates object extent and motion quantities. For QuantiPhy evaluation in world-space, we evaluate metric physical reasoning on the open-source QuantiPhy-validation set [11] and compute MRA against its released ground-truth numerical answers. We additionally report the 2S, 2D, 3S, and 3D subsets under the official protocol.

### 5.4.2 Image-Space Measurement

Both the 4B and 9B Image-Space variants achieve strong image-plane measurement performance despite their compact model sizes, as shown in Table 3. The complete Code-as-World-VL models further improve over their Image-Space counterparts on every benchmark, showing that World-Space training preserves and strengthens the underlying Image-Space grounding capability. Further evaluation details and complete results are provided in Appendix B.3.

### 5.4.3 Quantitative Physical Reasoning

QuantiPhy evaluates whether a model can use a physical prior to calibrate monocular video evidence into world-unit estimates of object size, displacement, velocity, and acceleration. We follow the official protocol and compare all models under the same video input and question format.

As shown in Table 1, Code-as-World-VL delivers strong quantitative physical reasoning in the controlled direct-answer comparison. Code-as-World-VL-4B substantially outperforms larger open-weight baselines and remains competitive with leading proprietary systems, while Code-as-World-VL-9B achieves the best

average performance among the direct-answer variants. Its consistent strength across the benchmark subsets shows that the gain is not confined to a particular quantity or scene configuration. Instead, Code-as-World-VL more reliably connects visual measurements with metric scale and motion, demonstrating the value of executable-world supervision for general quantitative reasoning from video.

To further validate this physical reasoning capability, we train a larger, reasoning-enabled Code-as-World-VL-27B that exposes its measurement-and-calibration reasoning trace before producing the final scalar answer. As shown in Table 1, the model exceeds the direct-answer 9B variant and the strongest proprietary baseline. Because model scale and response protocol change together, we view the 27B result as evidence that the framework extends to larger reasoning models, rather than as a controlled estimate of the effect of reasoning alone. Figure 10 illustrates this process on a QuantiPhy example, while Appendix B.5 details the reasoning-model setting. Figure 9 further summarizes the scaling behavior, with average MRA increasing from the 4B to 9B direct-answer variants and reaching its highest value for the reasoning-enabled 27B model.

### 5.5 Limitations

The empirical study in this section has two main limitations. First, QuantiPhy evaluates only a limited subset of physical understanding, focusing primarily on monocular scale calibration for size, displacement, velocity, and acceleration under relatively constrained motion settings. It therefore covers only a small part of real-world physics. Natural scenes can contain camera motion, rotation, deformation, occlusion, contact, collision, friction, fluids, rigid-body interactions, and long-horizon multi-object dynamics.

Second, Code-as-World-VL currently learns from supervision derived from verified executable worlds, but does not internalize the agentic discovery process itself. In other words, the model is trained on the outcomes of world representation and verification, while hypothesis construction, simulation, diagnosis, and iterative revision remain external to the model. Extending physical reasoning to broader mechanisms and turning this discovery loop into a native model capability are important directions for future work.

## 6 Related Work

**Physical understanding and reasoning.** Physical intelligence has been explored from increasingly richer forms of reasoning over the physical world. Spatial reasoning focuses on recovering geometric structures and relations between objects from visual observations [75–77]. Beyond spatial structure, physical question answering studies whether models can understand and reason about object properties, interactions, and measurable physical quantities from images and videos [9–11, 23, 78]. Another line of work investigates intuitive physics, evaluating whether models can acquire human-like expectations about object permanence, dynamics, and physical plausibility beyond surface-level visual correlations [79, 80]. Increasingly, interactive physical environments have been used to study whether agents can apply physical knowledge to solve novel tasks through action and intervention [1, 81–84]. Despite these advances, existing approaches primarily evaluate physical understanding through task-specific outputs, while the underlying representations of physical entities, states, and mechanisms remain largely implicit.

**Code as world representations.** Code has emerged as a promising medium for representing executable worlds. In virtual environments, recent works explore code-based world models for interactive agents, where programs serve as executable engines of environments that can be generated, modified, and evaluated through interaction [85–87]. Beyond virtual worlds, researchers have also investigated code-based representations for physical content creation and simulation. At the object level, recent benchmarks study the generation of 3D assets and procedural objects through code [88, 89]; at the scene level, programmatic representations have been used to describe editable 3D environments and indoor scenes [90–93]. These approaches demonstrate the advantages of code in providing structured, compositional, and editable representations, but they primarily focus on static scene structures. More recent work extends code-driven representations to physical dynamics through executable simulations and physics-aware reasoning [94–96]. However, these approaches do not address the more fundamental problem of abstracting physical mechanisms from real-world observations into executable representations.

**Agentic optimization and discovery.** Recent advances in agentic systems have explored how agents can autonomously discover improved solutions by iteratively proposing, evaluating, and refining external artifacts. Coding agents have demonstrated the ability to discover improved algorithms through evolutionary search and automated evaluation [22], while similar ideas have been extended to automating academic experiments [97], discovering reusable skills for robotics [98], and generating executable representations of visual environments [91, 93]. These approaches show that agents can move beyond executing predefined procedures toward actively searching for artifacts that better satisfy task objectives. Our work echoes this trend in the context of physical world representations: it treats an executable world hypothesis as the artifact to be discovered and iteratively refines it through simulation and verification against language or visual evidence.

# 7 Conclusion

Physical understanding requires more than predicting or describing observations; it requires representations that capture the mechanisms underlying how the world is composed, evolves, and can be manipulated. In this work, we introduced Code-as-World, which explores executable world representations as a bridge between visual observations and mechanism-grounded reasoning. By combining structured code representations with an evolving agentic discovery process, Code-as-World enables agents to construct, verify, and improve executable hypotheses of the observed physical world. We further showed that these verified executable worlds can provide scalable physical supervision for training vision-language models on quantitative physical reasoning. Our results suggest that executable code representations offer a promising direction toward more explicit, verifiable, and generalizable physical intelligence.

## 7.1 Broader Physical Phenomena

Our current implementation focuses primarily on rigid-body dynamics, but the Code-as-World paradigm is not tied to any particular physical regime. Code is sufficiently expressive to provide a common interface to specialized simulators for fluids [99, 100], cloth and other deformable bodies [101, 102], combustion [103, 104], fracture [105], elasticity and plasticity [106, 107], and gas dynamics [108], potentially connecting decades of progress in graphics and computational physics to physical intelligence. The central challenge lies in operationalizing this expressiveness: coding agents must learn to correctly invoke and compose various simulation engines, while the discovery process must extract sufficiently informative evidence from observations and verify candidate worlds across different physical regimes. Extending both the executable representation and its evidence-verification mechanisms is therefore an important direction for future work.

## 7.2 Broader Physical Capabilities

Beyond its current use as a source of outcome supervision for quantitative physical reasoning, representing the physical world through code opens several broader directions for physical intelligence:

- **General and grounded physical reasoning**: Code-as-World provides a structured representation over entities, states, relations, and dynamics, enabling supervision beyond sparse question-answering objectives. Future work can explore how executable worlds can train models to ground entities, infer physical relations, simulate possible interactions, and verify their own reasoning against explicit world states [1, 85, 86].
- **Physically consistent video generation**: Executable world representations provide video generators with persistent states that explicitly maintain object identity, geometry, and temporal evolution. By separating world dynamics from visual rendering [109], future systems may achieve more coherent long-horizon generation, controllable interventions, and improved physical consistency.
- **Deliberate embodied interaction**: Executable world representations may support a System-2-like form of embodied intelligence, in which agents construct explicit world models, reason over possible futures, and deliberate over candidate actions before execution rather than relying solely on reactive policies [110]. By providing a shared abstraction across passive observations, simulated environments, and real-world interaction, they can enable physical knowledge acquired in one setting to transfer

to another, offering a path toward scalable embodied learning beyond the limits of direct real-world experience.

## Authors

Hanyang Wang, Yimo Cai, Weiliang Chen, Jiawei Chi, Haowen Sun, Qiyu Dai, Yi-Hsin Hung, Xingzhuo Guo, Jinshan Ren, Runmao Yao, Ziwei Liu, Mingsheng Long, Yueqi Duan, Jun Gao, Jiangran Lyu, Fangfu Liu, Jialong Wu[†]

## Affiliations

MirroS, Tsinghua University, Peking University, Nanyang Technological University

[†]Project lead.

## References


[1] A. Bakhtin, L. van der Maaten, J. Johnson, L. Gustafson, and R. Girshick. "Phyre: A new benchmark for physical reasoning". In: *Advances in Neural Information Processing Systems* 32 (2019).

[2] Y. Bengio, A. Courville, and P. Vincent. "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[3] Google DeepMind. *Gemini 3.1 Flash-Lite*. Tech. rep. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-1-Flash-Lite-Model-Card.pdf. Google DeepMind, 2025.

[4] B. Seed. "Seed2. 0 model card: Towards intelligence frontier for real-world complexity". In: *arXiv preprint arXiv:2607.00248* (2026).

[5] K. Team, T. Bai, Y. Bai, Y. Bao, S. Cai, Y. Cao, et al. "Kimi k2. 5: Visual agentic intelligence". In: *arXiv preprint arXiv:2602.02276* (2026).

[6] Qwen Team. *Qwen3.5: Towards Native Multimodal Agents*. Feb. 2026.

[7] P. Machamer, L. Darden, and C. F. Craver. "Thinking about mechanisms". In: *Philosophy of science* 67.1 (2000), pp. 1–25.

[8] Y. Li, Q. Gao, T. Zhao, B. Wang, H. Sun, H. Lyu, et al. "Core knowledge deficits in multi-modal language models". In: *arXiv preprint arXiv:2410.10855* (2024).

[9] K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, and J. B. Tenenbaum. "Clevrer: Collision events for video representation and reasoning". In: *arXiv preprint arXiv:1910.01442* (2019).

[10] W. Chow, J. Mao, B. Li, D. Seita, V. C. Guizilini, and Y. Wang. "PhysBench: Benchmarking and Enhancing Vision-Language Models for Physical World Understanding". In: *The Thirteenth International Conference on Learning Representations*. 2025.

[11] L. Puyin, T. Xiang, E. Mao, S. Wei, X. Chen, A. Masood, L. Fei-Fei, and E. Adeli. "Quantiphy: A quantitative benchmark evaluating physical reasoning abilities of vision-language models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2026, pp. 33174–33184.

[12] H. Sun, Q. Gao, H. Lyu, D. Luo, Y. Li, and H. Deng. "Probing mechanical reasoning in large vision language models". In: *arXiv preprint arXiv:2410.00318* (2024).

[13] D. Luo, Y. Li, M. Wang, T. Zhao, B. Wang, S. Wang, et al. "Vision language models cannot reason about physical transformation". In: *arXiv preprint arXiv:2603.07109* (2026).

[14] L. M. Schulze Buschoff, E. Akata, M. Bethge, and E. Schulz. "Visual cognition in multimodal large language models". In: *Nature Machine Intelligence* 7.1 (2025), pp. 96–106.

[15] T. Brooks, B. Peebles, C. Holmes, W. DePue, Y. Guo, L. Jing, et al. "Video generation models as world simulators". In: (2024).

[16] T. Seedance, D. Chen, L. Chen, X. Chen, Y. Chen, Z. Chen, et al. "Seedance 2.0: Advancing video generation for world complexity". In: *arXiv preprint arXiv:2604.14148* (2026).

[17] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, M. Muckley, et al. "V-jepa 2: Self-supervised video models enable understanding, prediction and planning". In: *arXiv preprint arXiv:2506.09985* (2025).

[18] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, et al. "Dinov3". In: *arXiv preprint arXiv:2508.10104* (2025).

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.

[20] H. Liu, C. Li, Q. Wu, and Y. J. Lee. "Visual instruction tuning". In: *Advances in neural information processing systems* 36 (2023), pp. 34892–34916.

[21] J. R. Josephson and S. G. Josephson. *Abductive inference: Computation, philosophy, technology*. Cambridge University Press, 1996.

[22] A. Novikov, N. Vũ, M. Eisenberger, E. Dupont, P.-S. Huang, A. Z. Wagner, et al. "Alphaevolve: A coding agent for scientific and algorithmic discovery". In: *arXiv preprint arXiv:2506.13131* (2025).

[23] F. Zhou, J. Huang, J. Li, D. Ramanan, and H. Shi. *PAI-Bench: A Comprehensive Benchmark For Physical AI*. 2025. arXiv: 2512.01989 [cs.CV].

[24] J. Wu, S. Yin, N. Feng, X. He, D. Li, J. Hao, and M. Long. "ivideogpt: Interactive videogpts are scalable world models". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 68082–68119.

[25] S. Huang, J. Wu, Q. Zhou, S. Miao, and M. Long. "Vid2world: Crafting video diffusion models to interactive world models". In: *arXiv preprint arXiv:2505.14357* (2025).

[26] W. Chen, H. Sun, J. Gao, J. Chi, H. Wang, Q. Dai, et al. "HarnessEval-W: Agentifying the Evaluation of Visual Worlds". In: *arXiv preprint arXiv:2608.16859* (2026).

[27] B. Kang, Y. Yue, R. Lu, Z. Lin, Y. Zhao, K. Wang, G. Huang, and J. Feng. "How far is video generation from world model: A physical law perspective". In: *arXiv preprint arXiv:2411.02385* (2024).

[28] S. Motamed, L. Culp, K. Swersky, P. Jaini, and R. Geirhos. "Do generative video models understand physical principles?" In: *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2026, pp. 948–958.

[29] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. "Nerf: Representing scenes as neural radiance fields for view synthesis". In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[30] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. "3D Gaussian Splatting for Real-Time Radiance Field Rendering". In: *ACM Transactions on Graphics* 42.4 (July 2023).

[31] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. "DUSt3R: Geometric 3D Vision Made Easy". In: *CVPR*. 2024.

[32] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. "VGGT: Visual Geometry Grounded Transformer". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2025.

[33] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. Lensch. "Nerd: Neural reflectance decomposition from image collections". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 12684–12694.

[34] J. Gao, C. Gu, Y. Lin, Z. Li, H. Zhu, X. Cao, L. Zhang, and Y. Yao. "Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing". In: *European conference on computer vision*. Springer. 2024, pp. 73–89.

[35] Y. Jiang, J. Tu, Y. Liu, X. Gao, X. Long, W. Wang, and Y. Ma. "Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2024, pp. 5322–5332.

[36] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron. "NeRFactor: neural factorization of shape and reflectance under an unknown illumination". In: *ACM Transactions on Graphics* 40.6 (Dec. 2021), pp. 1–18. doi: 10.1145/3478513.3480496.

[37] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. "D-NeRF: Neural Radiance Fields for Dynamic Scenes". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[38] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz. "HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields". In: *ACM Trans. Graph.* 40.6 (Dec. 2021).

[39] Z. Yang, X. Gao, W. Zhou, S. Jiao, Y. Zhang, and X. Jin. "Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2024, pp. 20331–20341.

[40] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. "4d gaussian splatting for real-time dynamic scene rendering". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2024, pp. 20310–20320.

[41] Z. Yang, H. Yang, Z. Pan, and L. Zhang. "Real-time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting". In: *International Conference on Learning Representations (ICLR)*. 2024.

[42] Y. Duan, F. Wei, Q. Dai, Y. He, W. Chen, and B. Chen. "4D-Rotor Gaussian Splatting: Towards Efficient Novel-View Synthesis for Dynamic Scenes". In: *Proc. SIGGRAPH*. 2024.

[43] J. Berg, C. Zhu, Y. Bao, I. Durugkar, and A. Gupta. "Semantic world models". In: *arXiv preprint arXiv:2510.19818* (2025).

[44] D. Chen, T. Moutakanni, W. Chung, Y. Bang, Z. Ji, A. Bolourchi, and P. Fung. "Planning with reasoning using vision language world model". In: *arXiv preprint arXiv:2509.02722* (2025).

[45] Q. Han, K. Hu, L. Qiu, C. Wu, and K. He. *VISTA: A Visual Harness for Reasoning in an Interactive World*. Aug. 2026.

[46] J. Wu, X. Zhang, H. Yuan, X. Zhang, T. Huang, C. He, et al. "Visual generation unlocks human-like reasoning through multimodal world models". In: *arXiv preprint arXiv:2601.19834* (2026).

[47] X. Chen, F.-J. Chu, P. Gleize, K. J. Liang, A. Sax, H. Tang, et al. "Sam 3d: 3dfy anything in images". In: *arXiv preprint arXiv:2511.16624* (2025).

[48] J. Wang, A. Ma, K. Cao, J. Zheng, J. Feng, Z. Zhang, W. Pang, and X. Liang. "Wisa: World simulator assistant for physics-aware text-to-video generation". In: *Advances in Neural Information Processing Systems* 38 (2026), pp. 5388–5416.

[49] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, et al. "Sam 3: Segment anything with concepts". In: *arXiv preprint arXiv:2511.16719* (2025).

[50] J. Wang, M. Chen, S. Zhang, N. Karaev, J. Schönberger, P. Labatut, et al. "VGGT-Omega". In: *arXiv preprint arXiv:2605.15195* (2026).

[51] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. "A benchmark dataset and evaluation methodology for video object segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 724–732.

[52] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi. "Social gan: Socially acceptable trajectories with generative adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2255–2264.

[53] W. Jiang, E. Trulls, J. Hosang, A. Tagliasacchi, and K. M. Yi. "Cotr: Correspondence transformer for matching across images". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 6207–6217.

[54] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg. "Modeling Context in Referring Expressions". In: *European Conference on Computer Vision (ECCV)*. Springer. 2016, pp. 69–85.

[55] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei. "Kosmos-2: Grounding multimodal large language models to the world". In: *arXiv preprint arXiv:2306.14824* (2023).

[56] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, et al. "Deepseekmath: Pushing the limits of mathematical reasoning in open language models". In: *arXiv preprint arXiv:2402.03300* (2024).

[57] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning". In: *arXiv preprint arXiv:2501.12948* (2025).

[58] OpenAI. *GPT-5.1: A Smarter, More Conversational ChatGPT*. https://openai.com/index/gpt-5-1/. Accessed: 2026-07-29. Nov. 2025.

[59] G. Comanici, E. Bieber, M. Schaekermann, I. Pasupat, N. Sachdeva, I. Dhillon, et al. "Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities". In: *arXiv preprint arXiv:2507.06261* (2025).

[60] xAI. *Grok 4.1 Model Card*. Model Card. Accessed: 2026-07-29. xAI, Nov. 2025.

[61] OpenAI. *GPT-5 System Card*. System Card. Accessed: 2026-07-29. OpenAI, Aug. 2025.

[62] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, et al. "Qwen3-vl technical report". In: *arXiv preprint arXiv:2511.21631* (2025).

[63] W. Wang, Z. Gao, L. Gu, H. Pu, L. Cui, X. Wei, et al. "Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency". In: *arXiv preprint arXiv:2508.18265* (2025).

[64] A. Abouelenin, A. Ashfaq, A. Atkinson, H. Awadalla, N. Bach, J. Bao, et al. "Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras". In: *arXiv preprint arXiv:2503.01743* (2025).

[65] A. Marafioti, O. Zohar, M. Farré, M. Noyan, E. Bakouch, P. Cuenca, et al. "Smolvlm: Redefining small and efficient multimodal models". In: *arXiv preprint arXiv:2504.05299* (2025).

[66] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, et al. "Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 91–104.

[67] W. Hong, W. Wang, M. Ding, W. Yu, Q. Lv, Y. Wang, et al. "Cogvlm2: Visual language models for image and video understanding". In: *arXiv preprint arXiv:2408.16500* (2024).

[68] J. Lin, H. Yin, W. Ping, P. Molchanov, M. Shoeybi, and S. Han. "Vila: On pre-training for visual language models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 26689–26699.

[69] M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. H. Awadalla, et al. "Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone". In: *ArXiv* abs/2404.14219 (2024).

[70] T. Yu, Z. Wang, C. Wang, F. Huang, W. Ma, Z. He, et al. "Minicpm-v 4.5: Cooking efficient mllms via architecture, data, and training recipe". In: *arXiv preprint arXiv:2509.18154* (2025).

[71] H. Liu, C. Li, Q. Wu, and Y. J. Lee. "Visual instruction tuning". In: *Advances in neural information processing systems* 36 (2023), pp. 34892–34916.

[72] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy. "Generation and Comprehension of Unambiguous Object Descriptions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 11–20.

[73] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg. "ReferItGame: Referring to Objects in Photographs of Natural Scenes". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 787–798.

[74] L. Huang, X. Zhao, and K. Huang. "GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.5 (2019), pp. 1562–1577.

[75] J. Yang, S. Yang, A. W. Gupta, R. Han, L. Fei-Fei, and S. Xie. "Thinking in space: How multimodal large language models see, remember, and recall spaces". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 10632–10643.

[76] D. Wu, F. Liu, Y.-H. Hung, and Y. Duan. "Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence". In: *Advances in neural information processing systems* 38 (2026), pp. 13569–13597.

[77] F. Liu, D. Wu, J. Chi, Y. Cai, Y.-H. Hung, X. Yu, et al. "Spatial-TTT: Streaming Visual-based Spatial Intelligence with Test-Time Training". In: *arXiv preprint arXiv:2603.12255* (2026).

[78] F. Liu, H. Wang, S. Yao, S. Zhang, J. Zhou, and Y. Duan. "Physics3d: Learning physical properties of 3d gaussians via video diffusion". In: *arXiv preprint arXiv:2406.04338* (2024).

[79] R. Riochet, M. Y. Castro, M. Bernard, A. Lerer, R. Fergus, V. Izard, and E. Dupoux. "Intphys: A framework and benchmark for visual intuitive physics reasoning". In: *arXiv preprint arXiv:1803.07616* (2018).

[80] F. Bordes, Q. Garrido, J. T. Kao, A. Williams, M. Rabbat, and E. Dupoux. "Intphys 2: Benchmarking intuitive physics understanding in complex synthetic environments". In: *arXiv preprint arXiv:2506.09849* (2025).

[81] X. Xu, P. Bu, Y. Wang, B. F. Karlsson, Z. Wang, T. Song, et al. "DeepPHY: Benchmarking agentic VLMs on physical reasoning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 40. 40. 2026, pp. 34160–34168.

[82] C. Xue, V. Pinto, C. Gamage, E. Nikonova, P. Zhang, and J. Renz. "Phy-Q as a measure for physical reasoning intelligence". In: *Nature Machine Intelligence* 5.1 (2023), pp. 83–93.

[83] Y. Wu, M. Song, Y. Lan, L. Wang, Z. Hu, Y. Xiao, et al. "From Perception to Action: An Interactive Benchmark for Vision Reasoning". In: *arXiv preprint arXiv:2602.21015* (2026).

[84] R. Yao, K. Hu, Y. Cao, R. Wang, S. Tian, Z. Cao, et al. "Apple-pi: Benchmarking Thinking with Video Towards Law-Grounded Physical Intelligence". In: *arXiv preprint arXiv:2607.16401* (2026).

[85] W. Lehrach, D. Hennes, M. Lazaro-Gredilla, X. Lou, C. Wendelken, Z. Li, et al. "Code world models for general game playing". In: *International Conference on Learning Representations*. Vol. 2026. 2026, pp. 133870–133927.

[86] S. Rodionov. "Executable World Models for ARC-AGI-3 in the Era of Coding Agents". In: *International Conference on Artificial General Intelligence*. Springer. 2026, pp. 198–210.

[87] H. Tang, D. Key, and K. Ellis. "Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 70148–70212.

[88] Y. Zheng and F. Bordes. "Voxelcodebench: Benchmarking 3d world modeling through code generation". In: *arXiv preprint arXiv:2604.02580* (2026).

[89] Y. Gao, L. Shu, G. Ye, X. Xiong, A. Makadia, M. Guo, L. Itti, and J. Chen. "3DCodeBench: Benchmarking Agentic Procedural 3D Modeling Via Code". In: *arXiv preprint arXiv:2606.01057* (2026).

[90] Y. Zhang, Z. Li, M. Zhou, S. Wu, and J. Wu. "The scene language: Representing scenes with programs, words, and embeddings". In: *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2025, pp. 24625–24634.

[91] P. Wang, Y. Wang, L. Li, Z. Yang, K. Q. Lin, Y. Li, and Y. Cheng. "SceneCode: Executable World Programs for Editable Indoor Scenes with Articulated Objects". In: *arXiv preprint arXiv:2605.19587* (2026).

[92] Y. Yang, Z. Luo, W. Gan, J. Hao, J. Lu, J. Yan, Z. Lyu, and X. Xu. "Code-as-Room: Generating 3D Rooms from Top-Down View Images via Agentic Code Synthesis". In: *arXiv preprint arXiv:2605.18451* (2026).

[93] S. Yin, J. Ge, Z. Z. Wang, C. Wang, X. Li, M. J. Black, T. Darrell, A. Kanazawa, and H. Feng. "Vision-as-inverse-graphics agent via interleaved multimodal reasoning". In: *arXiv preprint arXiv:2601.11109* (2026).

[94] J. Liang, M. Ku, K.-H. Hui, P. Nie, and W. Chen. "VisPhyWorld: Probing Physical Reasoning via Code-Driven Video Reconstruction". In: *arXiv preprint arXiv:2602.13294* (2026).

[95] Ž. Kovačič and K. Ellis. "MPMWorlds: Material-Point-Method Simulations for Inferring and Extrapolating Physical Dynamics". In: *arXiv preprint arXiv:2606.01538* (2026).

[96] T. Xie, P. Wang, Y. Qian, Y. Wang, R. Ma, Y. Tai, et al. "PhysCodeBench: Benchmarking Physics-Aware Symbolic Simulation of 3D Scenes via Self-Corrective Multi-Agent Refinement". In: *arXiv preprint arXiv:2604.23580* (2026).

[97] A. Karpathy. *autoresearch*. https://github.com/karpathy/autoresearch. 2026.

[98] R. Lu, Y. Wu, E. Kou, L. Fu, W. Xiao, A. Mandlekar, et al. "ASPIRE: Agentic /Skills Discovery for Robotics". In: *arXiv preprint arXiv:2607.00272* (2026).

[99] J. Stam. "Stable fluids". In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '99. USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 121–128. doi: 10.1145/311535.311548.

[100] Q. Dai, X. Ni, qianfan Shen, W. Chen, B. Chen, and M. Chu. "RainyGS: Efficient Rain Synthesis with Physically-Based Gaussian Splatting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2025.

[101] D. Baraff and A. Witkin. "Large steps in cloth simulation". In: *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. New York, NY, USA: Association for Computing Machinery, 1998, pp. 43–54. doi: 10.1145/280814.280821.

[102] R. Narain, A. Samii, and J. F. O'brien. "Adaptive anisotropic remeshing for cloth simulation". In: *ACM transactions on graphics (TOG)* 31.6 (2012), pp. 1–10.

[103] D. Q. Nguyen, R. Fedkiw, and H. W. Jensen. "Physically based modeling and animation of fire". In: *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 2002, pp. 721–728.

[104] Q. Shen, N. Tao, Q. Dai, T. Chen, M. Qin, Y. Zhang, M. Chu, W. Chen, and B. Chen. "FieryGS: In-the-Wild Fire Synthesis with Physics-Integrated Gaussian Splatting". In: *The Fourteenth International Conference on Learning Representations*. 2026.

[105] J. F. O'brien and J. K. Hodgins. "Graphical modeling and animation of brittle fracture". In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 1999, pp. 137–146.

[106] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. "Elastically deformable models". In: *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. 1987, pp. 205–214.

[107] G. Irving, J. Teran, and R. Fedkiw. "Invertible finite elements for robust simulation of large deformation". In: *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 2004, pp. 131–140.

[108] R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher. "A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method)". In: *Journal of computational physics* 152.2 (1999), pp. 457–492.

[109] H. A. Alhaija, J. Alvarez, M. Bala, T. Cai, T. Cao, L. Cha, et al. "Cosmos-transfer1: Conditional world generation with adaptive multimodal control". In: *arXiv preprint arXiv:2503.14492* (2025).

[110] P. Intelligence, B. Ai, A. Amin, R. Aniceto, A. Balakrishna, G. Balke, et al. "pi0.7: a Steerable Generalist Robotic Foundation Model with Emergent Capabilities". In: *arXiv preprint arXiv:2604.15483* (2026).

[111] J. Y. Lim, R. Shah, Z. Ikram, S. Yu, H. Ma, T.-Y. Leong, and D. Liu. "JEDI: Joint Embedding Diffusion World Model for Online Model-Based Reinforcement Learning". In: *arXiv preprint arXiv:2605.13013* (2026).

[112] A. Bardes, Q. Garrido, J. Ponce, X. Chen, M. Rabbat, Y. LeCun, M. Assran, and N. Ballas. "V-jepa: Latent video prediction for visual representation learning". In: (2024).

[113] K. Allen, C. Doersch, G. Zhou, M. Suhail, D. Driess, I. Rocco, et al. "Direct motion models for assessing generated videos". In: *arXiv preprint arXiv:2505.00209* (2025).

[114] C. Doersch, P. Luc, Y. Yang, D. Gokay, S. Koppula, A. Gupta, et al. "Bootstap: Bootstrapped training for tracking-any-point". In: *Asian Conference on Computer Vision*. Springer. 2024, pp. 483–500.

[115] E. Todorov, T. Erez, and Y. Tassa. "MuJoCo: A Physics Engine for Model-Based Control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 5026–5033. doi: 10.1109/IROS.2012.6386109.

[116] T. Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, et al. "Wan: Open and advanced large-scale video generative models". In: *arXiv preprint arXiv:2503.20314* (2025).

[117] Z. Jiang, Z. Han, C. Mao, J. Zhang, Y. Pan, and Y. Liu. "Vace: All-in-one video creation and editing". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2025, pp. 17191–17202.

# Appendix

## A Dataset Details

We provide additional details on the datasets summarized in Sec. 5.4.1. The Image-Space datasets provide direct image-plane measurement supervision, whereas the Code-as-World executable-world dataset provides synchronized videos, physical states, and World-Space VQA examples from verified EWRs.

### A.1 Image-Space Datasets

**Image sources.** We construct image-based examples from RefCOCO and RefCOCO+ [54], RefCOCOg [72], and RefCLEF [73]. Each annotation associates a natural-language referring expression with a ground-truth bounding box. From the box coordinates, we derive questions about object width, height, diagonal extent, and relative image-plane position. We retain the original grounding task as auxiliary supervision by requiring the model to return the box of the referred object.

**Video source.** GOT-10K [74] provides dense, frame-level bounding boxes for a tracked object. We uniformly sample 16 temporally ordered frames from each clip and use the box centers and extents to construct questions about object size, displacement, speed, and acceleration in pixel units. We additionally construct timestamp-specific grounding questions from the sampled boxes. A short semantic description of the tracked object is included in each question to specify the target unambiguously.

**Statistics.** After removing samples whose images overlap with the evaluation data, the Image-Space training set contains 73,335 question–answer pairs. Of these, 46,763 are derived from GOT-10K: 20,858 speed, 8,399 velocity, 12,169 acceleration, 3,995 grounding, and 1,342 size-related examples. The remaining 26,572 examples come from the four referring-expression datasets, including 16,989 grounding and 9,583 size-related examples.

### A.2 World-Space Executable-World Dataset

The Code-as-World executable-world dataset unifies World-Space supervision constructed from reviewed language specifications and real-video observations. It contains 1,585 text-driven and 988 video-driven VQA samples.

## B Experiment Protocols

### B.1 Executable-World Fidelity and Realism Evaluation

We evaluate an executable world along three complementary axes: agreement with the source observation, preservation of object motion, and proximity to the distribution of real videos.

**Visual Alignment.** For frame $t$, let $M_t$ and $\widehat{M}_t$ denote the union of observed and rendered object masks, respectively. The silhouette component is their IoU, $s_t^{\mathrm{sil}} = |M_t \cap \widehat{M}_t| / |M_t \cup \widehat{M}_t|$, with a value of one when both masks are empty. Let $z_t$ and $\hat{z}_t$ be the observed and rendered depth maps, and let $I_t$ and $\hat{I}_t$ be their 8-bit RGB frames. For a pixel $p$ in the mask overlap $\Omega_t = M_t \cap \widehat{M}_t$, let $\Omega_t^{\mathrm{dep}}$ be the subset with finite observed/rendered depth and positive observed depth, and let $\Omega_t^{\mathrm{rgb}}$ be the valid RGB pixels in $\Omega_t$. We compute the median absolute relative depth error and normalized RGB error,

$$e_t^{\mathrm{dep}} = \underset{p \in \Omega_t^{\mathrm{dep}}}{\mathrm{median}} \frac{|\hat{z}_t(p) - z_t(p)|}{\max(|z_t(p)|, 10^{-6})}, \qquad e_t^{\mathrm{rgb}} = \underset{p \in \Omega_t^{\mathrm{rgb}}}{\mathrm{mean}} \frac{\|\hat{I}_t(p) - I_t(p)\|_1}{3 \times 255}. \tag{5}$$

The corresponding similarities are $s_t^{\mathrm{dep}} = (1 + e_t^{\mathrm{dep}})^{-1}$ and $s_t^{\mathrm{rgb}} = \max(0, 1 - e_t^{\mathrm{rgb}})$. Define the active component set $\mathcal{A}_t$ to always contain sil, to contain dep iff $\Omega_t^{\mathrm{dep}} \neq \emptyset$, and to contain rgb iff $\Omega_t^{\mathrm{rgb}} \neq \emptyset$. If $T$ is the number of native evaluated frames, Visual Alignment is the full-video mean of the active-weight-

renormalized frame score,

$$\mathrm{VA} = \frac{1}{T}\sum_{t=1}^{T}\frac{\sum_{c\in\mathcal{A}_t} w_c s_t^c}{\sum_{c\in\mathcal{A}_t} w_c}, \qquad (w_{\mathrm{sil}}, w_{\mathrm{dep}}, w_{\mathrm{rgb}}) = (0.60, 0.25, 0.15). \tag{6}$$

Thus, unavailable depth or RGB components are omitted before the weights are renormalized, while the silhouette component remains active on every frame.

**Object IoU.** Let $M_{t,o}$ and $\widehat{M}_{t,o}$ be the observed and rendered masks for object $o$ in frame $t$. We compute

$$\mathrm{IoU}_{t,o} = \frac{|M_{t,o} \cap \widehat{M}_{t,o}|}{|M_{t,o} \cup \widehat{M}_{t,o}|}, \tag{7}$$

and report Object IoU as its mean over all annotated object–frame pairs. We set IoU to one when both masks are empty. Rendered visibility is resolved with the final scene z-buffer so that occluded geometry is not penalized as an additional visible region. Higher Visual Alignment and Object IoU indicate closer visual agreement [51].

**Trajectory fidelity.** For an object, let $\mathbf{p}_t$ be its simulator ground-truth image-plane center at sampled frame $t$, and let $\hat{\mathbf{p}}_t$ be its CoTracker estimate in the video being evaluated. Both the simulator render and its sim-to-real video are initialized from $\mathbf{p}_0$ and compared against the same ground-truth trajectory. For each video, we uniformly sample 16 frames and resize it to width 512 before tracking. For a frame of width $W$ and height $H$, let $D = \sqrt{W^2 + H^2}$ be its diagonal. To keep the notation compact, $\langle\cdot\rangle$ denotes the mean over all valid objects and sampled frames, excluding the initialization frame; for velocity, it denotes the mean over valid consecutive frame pairs. Following standard trajectory-error evaluation [52, 53], we define

$$\text{Traj-ADE} = \frac{100}{D}\left\langle \|\hat{\mathbf{p}}_t - \mathbf{p}_t\|_2 \right\rangle, \tag{8}$$

$$\text{Velocity-ADE} = \frac{100}{D}\left\langle \|(\hat{\mathbf{p}}_t - \hat{\mathbf{p}}_{t-1}) - (\mathbf{p}_t - \mathbf{p}_{t-1})\|_2 \right\rangle, \tag{9}$$

$$\text{Accuracy@2\%D} = 100\left\langle \mathbb{1}[\|\hat{\mathbf{p}}_t - \mathbf{p}_t\|_2 \le 0.02D] \right\rangle, \tag{10}$$

Traj-ADE is reported in $\%D$ and Velocity-ADE in $\%D$/step. We retain absolute image positions and do not subtract the initial tracking offset. Dataset-level results give each executable world equal weight; its sim-to-real variants are averaged before the final cross-world mean.

**JEDi.** JEDi [111] measures distributional video realism using V-JEPA [112] video features. Let $\{\mathbf{z}_i^r\}_{i=1}^n$ be features from real reference videos and $\{\mathbf{z}_j^c\}_{j=1}^m$ those from candidate videos. With feature dimension $d$ and the degree-two polynomial kernel $k(\mathbf{u}, \mathbf{v}) = (\mathbf{u}^\top \mathbf{v}/d)^2$, we report the biased maximum mean discrepancy

$$\mathrm{JEDi} = 100\left[\frac{1}{n^2}\sum_{i,i'} k(\mathbf{z}_i^r, \mathbf{z}_{i'}^r) + \frac{1}{m^2}\sum_{j,j'} k(\mathbf{z}_j^c, \mathbf{z}_{j'}^c) - \frac{2}{nm}\sum_{i,j} k(\mathbf{z}_i^r, \mathbf{z}_j^c)\right]. \tag{11}$$

Lower JEDi indicates that the candidate video distribution is closer to the real reference distribution.

**TRAJAN.** TRAJAN [113] isolates motion realism. We extract point tracks with BootsTAPIR [114], encode them with the TRAJAN TrackAutoEncoder, and flatten the resulting $128 \times 64$ latent into an 8192-dimensional feature. From each feature set, we estimate the sample mean and unbiased sample covariance $(\mu, \Sigma)$. Given $(\mu_r, \Sigma_r)$ and $(\mu_c, \Sigma_c)$ for real and candidate videos, the reported distance is

$$\mathrm{TRAJAN} = \|\mu_r - \mu_c\|_2^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_c - 2\left(\Sigma_r^{1/2}\Sigma_c\Sigma_r^{1/2}\right)^{1/2}\right). \tag{12}$$

Lower TRAJAN indicates closer agreement with real-video motion statistics. Here Tr denotes the matrix trace, and each matrix square root is the principal positive-semidefinite square root.

## B.2 Executable Engine Configurations

We use MuJoCo [115] as the simulation platform for constructing and executing EWRs. Within this platform, we support two interchangeable execution engines. The animation engine describes motion kinematically through time-varying poses and trajectories, emphasizing what moves and how it moves, whereas the physics engine describes mechanical bodies and interactions, deriving motion from properties such as forces, contacts, and constraints. Both engines implement the same executable-world interface and can therefore be plugged into the agentic discovery process independently of input evidence, permitting flexible evidence–engine pairings according to the desired motion control and physical detail. The main paper illustrates the agentic discovery process using video evidence with the animation engine, while Section C.1 reports results using video evidence with the physics engine. After simulation, we apply a realism-oriented re-rendering stage to the rendered videos using Wan2.2-VACE [116, 117] together with an internal video generation model. This stage conditions on the simulator render to improve photorealism while preserving the scene structure and temporal dynamics of the executable rollout.

## B.3 Image-Space Measurement Evaluation

The Image-Space evaluation isolates direct measurement from World-Space calibration. For an axis-aligned box $B_t = (x_t^{\min}, y_t^{\min}, x_t^{\max}, y_t^{\max})$, its width, height, and diagonal extent are

$$w_t = x_t^{\max} - x_t^{\min}, \qquad h_t = y_t^{\max} - y_t^{\min}, \qquad \ell_t = \sqrt{w_t^2 + h_t^2}. \tag{13}$$

For video, let $\mathbf{c}_t$ be the center of the tracked box at a uniformly sampled timestamp with interval $\Delta t$. We compute image-plane velocity and acceleration by central differences,

$$\mathbf{v}_t = \frac{\mathbf{c}_{t+1} - \mathbf{c}_{t-1}}{2\Delta t}, \qquad \mathbf{a}_t = \frac{\mathbf{c}_{t+1} - 2\mathbf{c}_t + \mathbf{c}_{t-1}}{\Delta t^2}. \tag{14}$$

Scalar speed and acceleration targets are the corresponding vector magnitudes. We generate questions only at timestamps for which all required boxes are valid. The targets remain in raw pixels, pixels per second, or pixels per second squared; no object-size prior or world-unit scale is supplied.

Scalar predictions are evaluated with Eq. (15), using the same parsing and relative-error computation. We average sample-level MRA independently within each benchmark, yielding the five dataset scores reported in Table 3. Grounding outputs are parsed separately as bounding boxes; grounding examples are not included in the scalar MRA reported in the table.

## B.4 World-Space QuantiPhy Evaluation

We evaluate on the open-source QuantiPhy validation set [11]. It contains 159 quantitative question–answer pairs with visible ground-truth answers. Each example provides a monocular video, a question, and a physical prior from which the model must infer a scalar World-Space quantity. We follow the official categorization and evaluation protocol.

**Kinematic categories.** The four reported subsets are denoted 2S, 2D, 3S, and 3D. The numeral specifies the spatial setting: 2 denotes planar kinematics, whereas 3 denotes a depth-aware three-dimensional setting. The letter specifies the type of the provided source prior: S denotes a static quantity, such as object size, and D denotes a dynamic quantity, such as velocity or acceleration. The queried target may itself be static or dynamic; the subset is determined by the source prior and spatial setting.

**Mean Relative Accuracy.** We use Mean Relative Accuracy (MRA) [75]. For sample $i$, let $\hat{y}_i$ denote the raw model response. The evaluator parses it as a scalar and canonicalizes its sign as $\tilde{y}_i = |\,\mathrm{parse}(\hat{y}_i)|$. Given ground-truth $y_i > 0$, the relative error is $r_i = |\tilde{y}_i - y_i|/y_i$. Let $\Theta = \{0.1, 0.2, \ldots, 0.9, 0.95\}$. The sample-level score is

$$\mathrm{MRA}_i = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \mathbb{1}[r_i < 1 - \theta]\,. \tag{15}$$

Thus, a prediction receives credit at each of ten increasingly strict relative error thresholds. A response that cannot be parsed as a finite number fails all thresholds and contributes zero.

**Table 2** Distributional realism and motion fidelity of text-driven sim-to-real video generation. Realism metrics use held-out real videos as the reference distribution, whereas motion metrics compare CoTracker estimates against simulator ground-truth trajectories.

| | Distributional Realism | | Motion Fidelity | | |
|---|---|---|---|---|---|
| **Video Type** | **JEDi MMD** ↓ | **TRAJAN Fréchet** ↓ | **Traj-ADE** ↓ (%$D$) | **Velocity-ADE** ↓ (%$D$/step) | **Accuracy@2%$D$** ↑ (%) |
| Simulator Render | 3.000 | 406.872 | 1.682 | 0.404 | 78.81 |
| Sim-to-Real Video | 1.484 | 185.321 | 1.677 | 0.472 | 77.49 |

Within each of 2S, 2D, 3S, and 3D, we average the sample-level MRA values. The overall score is the unweighted macro-average of these four subset scores, rather than a sample-weighted average. All MRA values in the main paper are multiplied by 100 for presentation.

### B.5 Reasoning-Model Setting

**Scope and input.** Code-as-World-VL-27B (Reasoning) is a separate outcome-supervised reasoning variant, rather than a third point in the controlled 4B/9B direct-answer comparison. It receives the same 16 temporally ordered frames, question, unit, and benchmark-provided physical prior as the direct-answer models. For depth-aware questions, it also receives the same benchmark-provided depth context. It has no access to the generating EWR, simulator states, object tracks, external tools, retrieval, or ground-truth measurements at test time.

**Training protocol.** The 27B variant is optimized with GRPO on the merged text-driven and video-driven World-Space questions constructed from verified executable worlds. Each prompt produces 16 rollouts; the global update batch size is 16, and optimization uses AdamW with learning rate $5 \times 10^{-6}$, weight decay 0.01, five warm-up steps, and bfloat16 parameters. We enable the model's medium-effort thinking mode and impose no KL penalty. The reward parser discards the reasoning trace and computes MRA only from the final numerical answer. No process label or reward supervises the content of the trace, keeping the reasoning training outcome-based.

**Response schema.** The system prompt supplies the opening `<think>` tag and asks the model to close the trace with `</think>`, then emit only one scalar and the requested unit on a single line. Within the trace, a complete solution can identify the target and timestamps, obtain an Image-Space measurement, resolve depth and perspective, and calibrate the measurement into a World-Space quantity. The trace is free-form rather than a sequence of separately supervised fields. Figure 10 presents a representative generated trace from the reported model.

**Inference and parsing.** For the result in Table 1, we evaluate the model at training step 60. We sample one response per QuantiPhy example with temperature 1.0, top-$p = 0.95$, and a maximum response length of 6,144 tokens. Subset MRA values and their unweighted macro-average in the table are reported from this step-60 validation run. We do not use self-consistency, majority voting, best-of-$N$ selection, or tool calls. The evaluator takes only the text after the final `</think>` tag and extracts its first scalar; a short one-line direct answer is accepted as a fallback, while an unclosed multi-line reasoning trace is unparseable and receives zero. Numbers inside the reasoning trace are thus never visible to the numerical evaluator.

## C Additional Experiments

### C.1 Text-Driven Sim-to-Real Evaluation

Having established that an EWR can faithfully explain its input, we next evaluate whether it can produce videos that are both realistic and reliably labeled in Table 2. For text-driven worlds, the simulator provides exact states, trajectories, and physical labels, while the video generator should improve visual realism without changing this information. We therefore evaluate sim-to-real generation along two axes: distributional realism and motion fidelity.

The generated videos are closer to authentic videos in both overall video-feature and motion-feature dis-

tributions, while retaining motion agreement comparable to the original simulator renders. Sim-to-real generation therefore improves visual appearance without materially changing the physical evolution specified by the EWR. Taken together, the two evaluations first establish that the executable representation remains faithful to its source evidence and then show that it can produce realistic observations without sacrificing the reliability of simulator-derived states and labels.

## C.2 Additional Agentic Discovery Loop

To assess whether iterative discovery remains effective beyond the animation engine used in the main analysis, we repeat the experiment with the physics engine described in Appendix B.2. As shown in Figure 11, performance improves consistently over the five discovery rounds: Visual Alignment, Object IoU, and Accuracy@2%$D$ increase, while Traj-ADE and Velocity-ADE decrease. By the fifth round, the agentic discovery loop also outperforms the matched-budget Best-of-5 baseline on all five metrics, showing that its iterative gains persist under a different executable engine.

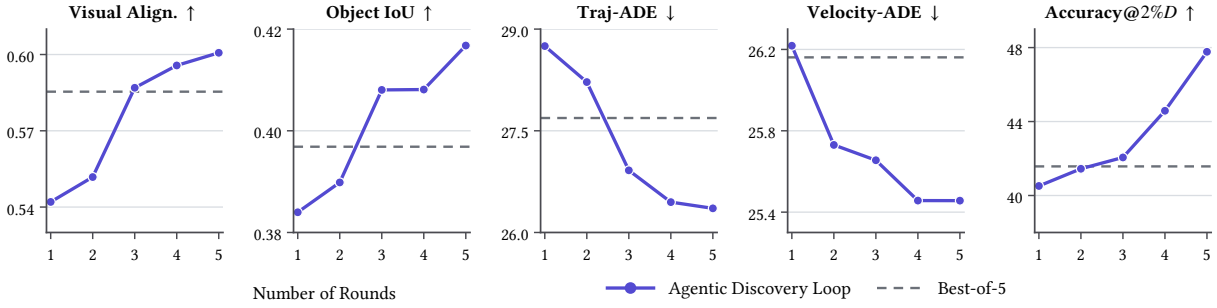



**Figure 11 Agentic discovery with the physics engine.** Solid curves report performance over five iterative discovery rounds, while dashed gray lines mark Best-of-5 under the same five-evaluation budget. Higher is better for Visual Alignment, Object IoU, and Accuracy@2%$D$; lower is better for Traj-ADE and Velocity-ADE.

## C.3 Image-Space Measurement Results

We compare Code-as-World-VL with open-weight models on four referring-expression datasets and GOT-10K. These benchmarks cover object extent and grounding in images as well as object extent and motion in video. Table 3 reports the quantitative comparison. Figure 12 complements the quantitative results in Table 3 with representative predictions for grounding, length, speed, and acceleration. In the examples shown, Code-as-World-VL identifies the intended target and produces measurements close to the ground truth, whereas the comparison models exhibit larger localization or numerical errors.

**Table 3 Pixel-level measurement evaluation.** Referring-expression datasets evaluate object extent in images, while GOT-10K evaluates object extent and motion in video. Image-Space variants use only the first training phase; full Code-as-World-VL models additionally receive executable-world reinforcement learning.

| Models | Size | RefCOCO† | RefCOCOg† | RefCOCO+† | RefCLEF† | GOT-10K* |
|---|---|---|---|---|---|---|
| *Open-weight models* | | | | | | |
| Qwen3-VL-32B-Instruct [62] | 32B | 49.1 | 40.3 | 44.9 | 32.5 | 15.8 |
| Qwen3.5-27B [6] | 27B | 47.0 | 44.8 | 46.8 | 31.6 | 6.1 |
| Qwen3.5-9B [6] | 9B | 38.6 | 33.0 | 38.6 | 37.8 | 15.9 |
| MiniCPM-V 4.5 [70] | 8B | 55.6 | 53.2 | 54.1 | 38.6 | 15.9 |
| InternVL-3.5-8B [63] | 8B | 36.4 | 30.9 | 35.2 | 24.2 | 14.4 |
| Qwen3-VL-8B-Instruct [62] | 8B | 43.4 | 43.0 | 41.8 | 40.0 | 18.9 |
| Qwen3.5-4B [6] | 4B | 28.6 | 26.1 | 30.4 | 20.9 | 3.0 |
| InternVL-3.5-4B [63] | 4B | 36.9 | 32.5 | 35.9 | 27.5 | 8.4 |
| *Ours* | | | | | | |
| **Code-as-World-VL-4B (Image-Space)** | **4B** | 56.1 | 51.6 | 56.2 | 31.2 | 18.7 |
| **Code-as-World-VL-4B** | **4B** | 62.9 | 60.2 | 63.3 | 39.8 | 20.5 |
| **Code-as-World-VL-9B (Image-Space)** | **9B** | 63.7 | 61.1 | 61.5 | 47.2 | 20.1 |
| **Code-as-World-VL-9B** | **9B** | **68.3** | **65.6** | **66.4** | **61.9** | **26.6** |

The Image-Space variants establish reliable visual grounding and pixel-level measurement, which provide the perceptual basis for subsequent World-Space learning. After adding World-Space supervision, the full 4B and 9B models improve on all five Image-Space benchmarks over their corresponding Image-Space variants. The two training stages therefore reinforce one another: Image-Space supervision grounds physical reasoning in observable measurements, while World-Space supervision feeds physical consistency back into grounding and quantitative measurement.

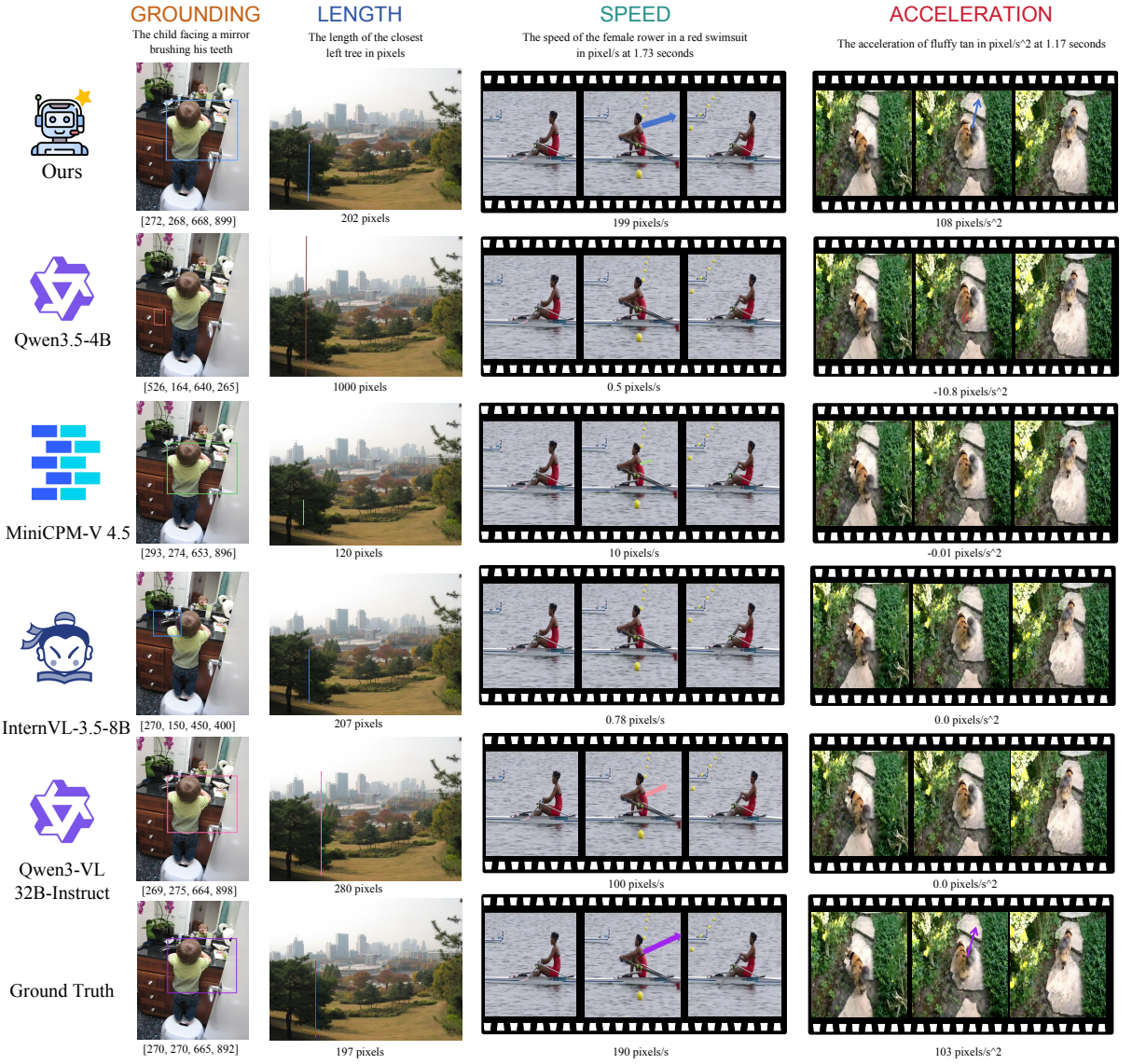



**Figure 12** Qualitative comparison of Image-Space grounding and measurement. The four columns evaluate target grounding, object length, speed, and acceleration, respectively. All numerical quantities are expressed in the pixel-space units specified by the question.

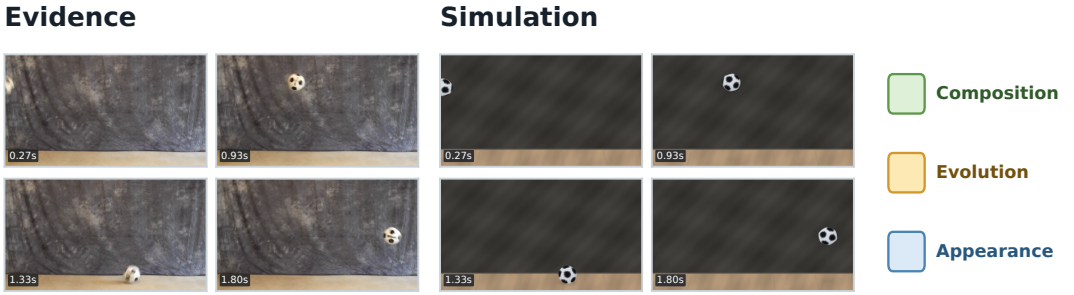
## C.4 Ablation of Data Sources

We ablate the three data sources in our two-phase curriculum: Image-Space measurement data $\mathcal{D}_{\text{pix}}$, text-driven executable worlds $\mathcal{D}_{\text{text}}$, and video-driven executable worlds $\mathcal{D}_{\text{video}}$. We perform this ablation on our compact 4B and 9B models. All variants first use $\mathcal{D}_{\text{pix}}$ to establish Image-Space measurement, after which the two World-Space sources are introduced separately or jointly.

**Table 4 Ablation of data sources.** We report MRA on the QuantiPhy subsets and their macro-average for the three data sources in our two-phase curriculum.

| Training Variants | Size | $\mathcal{D}_{\text{pix}}$ | $\mathcal{D}_{\text{text}}$ | $\mathcal{D}_{\text{video}}$ | 2S | 2D | 3S | 3D | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| Image-Space Only | 4B | ✓ | | | 45.3 | 45.1 | 31.6 | 54.6 | 44.2 |
| + Text-Driven Worlds | 4B | ✓ | ✓ | | 47.5 | 49.4 | 40.0 | 57.0 | 48.5 |
| + Video-Driven Worlds | 4B | ✓ | | ✓ | 42.2 | 52.7 | 46.5 | 49.8 | 47.8 |
| **Full Code-as-World-VL** | 4B | ✓ | ✓ | ✓ | 45.4 | **55.4** | 45.8 | 56.0 | 50.6 |
| Image-Space Only | 9B | ✓ | | | 44.3 | 51.9 | 51.4 | 60.0 | 50.9 |
| + Text-Driven Worlds | 9B | ✓ | ✓ | | 47.2 | 48.1 | 53.5 | 61.1 | 52.5 |
| + Video-Driven Worlds | 9B | ✓ | | ✓ | 43.1 | 50.5 | **57.4** | **61.5** | 53.1 |
| **Full Code-as-World-VL** | 9B | ✓ | ✓ | ✓ | **55.0** | 52.9 | 55.6 | 58.1 | **55.4** |

As shown in Table 4, adding either World-Space source improves the 4B Image-Space model, and combining both sources achieves the best average of 50.6, demonstrating their complementary benefits. The same trend holds at the larger scale, where the full 9B model improves from 50.9 to 56.8. Together, these results show that exact simulator supervision and real-video alignment contribute complementary signals beyond Image-Space grounding.




**scene.json**

```
{
  "schema_version": 1,
  "coordinate_system": {"handedness": "right",
    "up_axis": "y", "length_unit": "meter"},
  "camera": {
    "model": "perspective",
    ...
    "binding": {"frame_count": 60,
      "image_size": [280, 504],
      ...
      "source_image_size": [1080, 1920]}
  },
  ...
  "execution_mode": "physics",
  "collision_objects": ["ball", "floor"],
  "expected_collisions": true,
  "external_wrenches": [],
  "contact_events": [
    {"frame_index": 14, "time_seconds": 0.533133,
      "objects": ["ball", "floor"],
      ...
      "kind": "native_contact"},
    {"frame_index": 40, "time_seconds": 1.400667,
      "objects": ["ball", "floor"],
      ...
      "kind": "native_contact"}
  ],
  ...
  "objects": [
    {
      "id": "ball", "semantic_label": "\u8db3\u7403",
      "role": "dynamic_rigid",
      "motion_execution": "physics",
      "extent": [0.2196042402061299,
        0.2196042402061299, 0.2196042402061299],
      "mesh": {"primitive_type": "sphere",
        "procedural_primitive": true,
        ...
        "appearance": "soccer_panels"},
      "initial_state": {
        "position": [-1.563215794156141,
          0.8104922155577554, -2.4312423355321666],
        ...
        "linear_velocity": [1.3552338287520653,
          -0.6983782887575889, 0.005079997441856851],
        "angular_velocity": [0.0, 0.0, 0.0]
      },
      "physics": {
        "body_type": "dynamic", "collision": {
          "shape": "sphere", "radius": 0.10980212010306495},
        "friction": 0.0, "mass": 3.6768789664225365,
        ...
        "restitution": 0.8059094118124696
      }
      ...
    },
    {
      "id": "floor", "semantic_label": "\u5730\u9762",
      "role": "static_rigid",
      "extent": [2.222890615463257,
        0.030757149681448936, 0.31273001432418823],
      ...
      "physics": {"body_type": "fixed",
        "collision": {"shape": "oriented_box_proxy"},
        ...
        "friction": 0.0, "restitution": 0.1}
    }
  ],
  ...
  "relations": [
    {"subject": "ball", "object": "floor",
      "type": "above", "confidence": 0.98}
  ],
  "rendering": {
    "background_kind": "synthetic",
    ...
    "preview_video": "renders/final_simulation.mp4"
  },
  "simulator": {
    "backend": "mujoco", "integrator": "implicitfast",
    "gravity": [0.0,
      -9.228704973337999, 0.0],
    "time_step": 0.001, "solver_iterations": 100
    ...
  },
  ...
  "timeline": {
    "fps": 29.969730572122224, "frame_count": 60,
    "time_basis": "source PTS seconds"
  }
  ...
}
```

**simulation_sdk.py**

```
...
    scene = {
        "schema_version": 1,
        # Object-level execution is semantic; all numeric fitting remains SDK-owned.
        "execution_mode": effective_mode,
        "physics_recovered": False,
        "expected_collisions": expected_collisions,
        "collision_objects": collision_objects,
        "coordinate_system": {
            "handedness": "right",
            "up_axis": "y",
            "length_unit": "meter" if scale_status == "metric" else "scene_unit",
            "scale_status": scale_status,
        },
        "timeline": timeline,
        "camera": {
            "model": "perspective",
            "trajectory": _relative(camera_path, run_root),
            "binding": _load_camera_trajectory(camera_path),
        },
        "objects": objects,
        "relations": relations,
        "simulator": {
            "backend": "mujoco",
            "time_step": _positive_number(
                request["parameters"].get("time_step", 1.0 / 240.0), "time_step"
            ),
            "gravity": gravity.tolist(),
            "gravity_units": "m/s2" if scale_status == "metric" else "scene_units/s2",
            "scene_path": _relative(xml_path, run_root),
        },
        "ground": ground,
        "boundaries": environment["boundaries"] if environment is not None else {},
        "tracking_source": _relative(poses_path, run_root),
        "trajectory_source": _relative(poses_path, run_root),
...
    (
        optimized_objects,
        optimized_boundaries,
        optimized_wrenches,
        gravity_scale,
    ) = _apply_scene_parameter_vector(
        original,
        original_boundaries,
        original_wrenches,
        specs,
        best_values,
        normal,
    )
    scene["objects"] = optimized_objects
    scene["boundaries"] = optimized_boundaries
    scene["external_wrenches"] = optimized_wrenches
    optimized_gravity = (base_gravity * gravity_scale).astype(float).tolist()
    scene["simulator"]["gravity"] = optimized_gravity
    final_parameters = copy.deepcopy(dict(mjcf_parameters))
    final_parameters["gravity"] = optimized_gravity
...
```

**Figure 13 A concrete EWR code and simulation interface.** A video-driven scene is encoded as a structured `scene.json` specification of its composition, evolution, appearance, physical parameters, relations, simulator configuration, and timeline. The simulator SDK instantiates and executes this specification in MuJoCo, while the frame-aligned evidence and simulation outputs above illustrate the resulting reconstruction.