AGENTDREAMER: AGENTIC 3D MOTION REASONING FOR TEXT-CONTROLLABLE PHYSICS SIMULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose AgentDreamer, an agent-guided optimization framework that aligns text instructions with simulated motion through explicit 3D motion planning. Given a text instruction and scene geometry, an off-the-shelf agent translates the instruction into a structured semantic motion prior, specifying what should move, in which direction, and in what temporal order. The agent represents this prior using parametric functions over a series of temporal chunks, yielding dense 3D trajectories. We then optimize the parameters of a differentiable MPM simulator such that its rollout best matches the prior while remaining constrained by the scene dynamics. This allows the agent to provide semantic guidance from language, while the simulator grounds it in physically feasible motion. We further introduce Motion-IG, an information-theoretic metric for text–motion alignment based on masked text recovery. By comparing the likelihood of recovering masked motion-related content with and without the simulated motion, Motion-IG measures the information contributed by the motion beyond that available from the remaining text and scene context. Experiments show that AgentDreamer improves text alignment over video-based guidance, producing simulations that more closely follow the requested physical behavior.

1 INTRODUCTION

Controlling physical motion in simulated scenes is important for creating dynamic environments for embodied AI and interactive content generation. Without an intuitive control interface, users must manually tune simulation parameters through repeated simulation and adjustment to obtain the desired motion. Text-controllable physics simulation offers a promising alternative by allowing users to directly describe the intended behavior in natural language. Recent approaches, including PhysFlow (Liu et al., 2025) and MotionPhysics (Wang et al., 2026), leverage video generative priors to guide the optimization of simulation parameters. In this way, they connect high-level user specifications to the dynamics of a 3D scene.

Despite this progress, the resulting simulation may not reproduce the physical behavior described by the text. A central difficulty lies in the motion reference itself: visually plausible video generation does not ensure that the requested dynamics are represented correctly. For example, a request for a rolling ball may produce sliding, omitting the rotation that distinguishes the two behaviors. Optimizing a simulator to follow such a reference can therefore yield physically plausible motion that remains inconsistent with the text. This discrepancy motivates explicit 3D motion planning that describes how the object should move to reproduce the requested behavior.

We propose AgentDreamer, a framework for text-controllable physics simulation that bridges text and simulated motion through agent-based 3D motion planning. Our key idea is to formulate motion planning as a physical reasoning problem. Given a text instruction and scene geometry, an off-the-shelf multimodal agent draws on its language understanding and physical knowledge to infer how the object should move, as shown in Figure 1. The agent thus takes a direct role in constructing the motion plan, extending its use beyond estimating material parameters (Liu et al., 2025; Wang et al., 2026). This plan provides explicit 3D guidance for translating the text into simulated motion.

To support motion reasoning, we introduce automatic semantic injection that associates sampled control points with visible object parts. We provide a scene image with colored point markers corresponding to the 3D point input, together with coordinate-system information. This correspondence

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

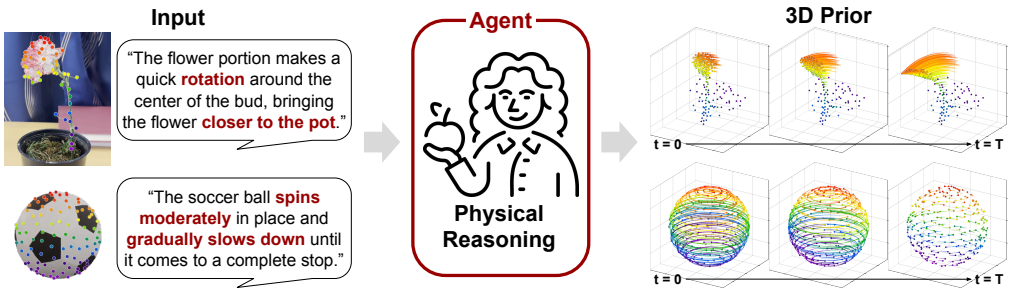

Figure 1: 3D motion planning as physical reasoning. Given a text instruction and scene geometry, the agent reasons about the physical behavior required by the instruction and constructs a corresponding parametric motion plan. The plan produces dense 3D trajectories that guide physics optimization toward the requested motion.

helps the agent interpret the role of each point in the requested motion. The agent then divides the time horizon into temporal chunks and infers a parametric motion function for each. These functions describe distinct motion stages and provide dense trajectories without predicting each future position independently. The resulting trajectories serve as pseudo supervision for a differentiable Material Point Method (MPM) simulator. We optimize simulation parameters to match these trajectories, guiding the simulator toward physically plausible motion aligned with the text.

We further introduce Motion-IG, an information-gain-based metric for measuring text-motion alignment through masked text recovery. An MLLM-as-a-judge score (Zheng et al., 2023; Chen et al., 2024), read as the model's verbalized confidence, considers the complete text and simulated motion together, without isolating the information provided by the motion. We instead mask motion-related content in the text and compare its recovery likelihoods with and without simulated motion under a fixed evaluation model. The difference measures how much the motion contributes to recovering the masked content beyond the remaining text.

Experiments demonstrate that AgentDreamer improves text alignment over video-based guidance, producing simulations that more closely follow the requested physical behavior. Our contributions are summarized as follows:

- We propose AgentDreamer, a framework for text-controllable physics simulation that bridges text and simulated motion through agent-based 3D motion planning.
- We formulate motion planning as physical reasoning through parametric functions over temporal chunks, providing dense trajectories as pseudo supervision for differentiable physics optimization.
- We introduce Motion-IG, an information-gain-based metric for text-motion alignment that compares masked text recovery likelihoods with and without simulated motion.

2 Related Works

Dynamic 3D representations Recent methods extend 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) to reconstruct dynamic scenes from video sequences. These methods (Luiten et al., 2024; Wu et al., 2024; Yang et al., 2024; Lin et al., 2024a; Katsumata et al., 2024; Jiang et al., 2025a; Lee et al., 2024) represent temporal changes through persistent Gaussian trajectories, deformation fields, or space-time primitives. The particle-based structure of Gaussian primitives further enables these representations to be extended to physics-based dynamics. PhysGaussian (Xie et al., 2024) couples Gaussian primitives with Material Point Method (MPM) simulation, using a shared representation for physical dynamics and rendering. Subsequent works develop elastic deformation models (Zhong et al., 2024), estimate material properties (Zhang et al., 2024; Cai et al., 2024), and support diverse material behaviors and interactions (Lin et al., 2025; Jiang et al., 2025b).

Text-conditional physics simulation Text provides an intuitive interface for specifying object dynamics and interactions. Recent approaches (Wang et al., 2026; Liu et al., 2025; 2024; Lin et al.,

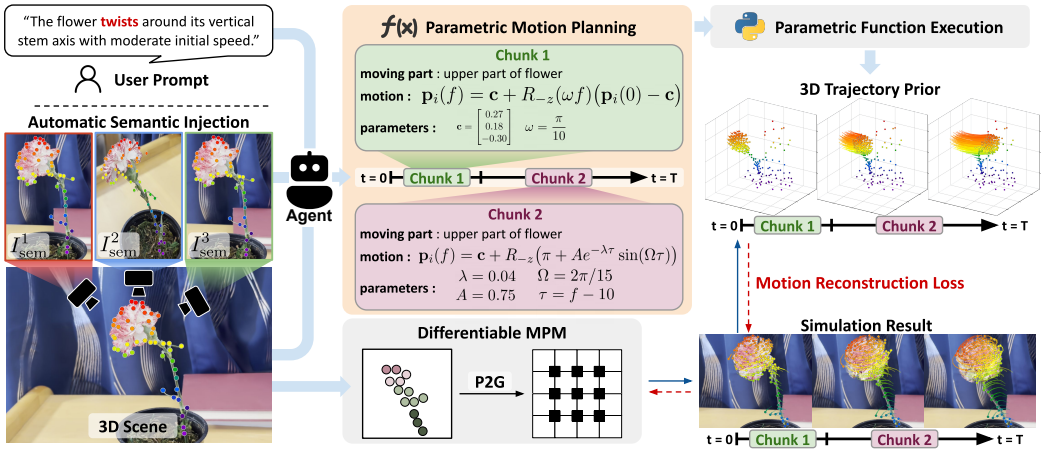

Figure 2: Overview of AgentDreamer. Given text and initial scene geometry, an agent constructs a parametric 3D motion plan. The resulting trajectories provide pseudo supervision for optimizing a differentiable MPM simulator to reproduce the requested behavior.

2025) translate these instructions into simulation guidance through video diffusion priors. PhysFlow (Liu et al., 2025) uses optical-flow guidance to optimize physical parameters, while Motion-Physics (Wang et al., 2026) employs learnable motion distillation to guide simulation toward text-specified dynamics. These approaches primarily derive motion guidance in 2D space from generated videos. In contrast, our method constructs an explicit 3D motion prior through physical reasoning with an off-the-shelf multimodal agent, directly specifying the desired scene dynamics. Prior works have also used multimodal models to construct explicit motion priors, including bounding-box plans in VLIPP (Yang et al., 2025) and 2D point trajectories in C-Drag (Li et al., 2025). However, these representations remain sparse and confined to image space. Our approach instead produces dense 3D trajectories over the scene geometry, which serve as pseudo supervision for differentiable physics simulation.

Agent-guided physics-aware motion evaluation Vision-language models provide semantic and physical feedback for evaluating and improving generated motion (Lin et al., 2024b; Bansal et al., 2025; Meng et al., 2024). PhyGDPO (Cai et al., 2026) uses VLM-based scores of semantic adherence and physical commonsense to guide groupwise preference optimization. PhyCo (Narayanan et al., 2026) evaluates generated videos through targeted physical questions and converts response logits into differentiable rewards for refining video generation. These approaches assess motion through semantic or physical judgments (Zheng et al., 2023; Chen et al., 2024), but do not explicitly separate the evidence supplied by motion from information available in the textual context alone. We instead formulate text-motion alignment as conditional information gain (Wyner, 1978), measuring how much a simulated trajectory improves the recovery of masked motion semantics over the remaining text.

3 METHOD

We present AgentDreamer, a framework for text-controllable physics simulation through agent-generated 3D motion planning, as shown in Figure 2. The agent translates the requested behavior into parametric motion functions, whose trajectories guide physics optimization. We further introduce Motion-IG to measure text–motion alignment through masked text recovery.

3.1 TEXT-CONTROLLABLE PHYSICS SIMULATION

Given a text instruction X and initial scene geometry $\mathcal{G}$, we construct a 3D motion plan and optimize simulation parameters to follow its trajectories.

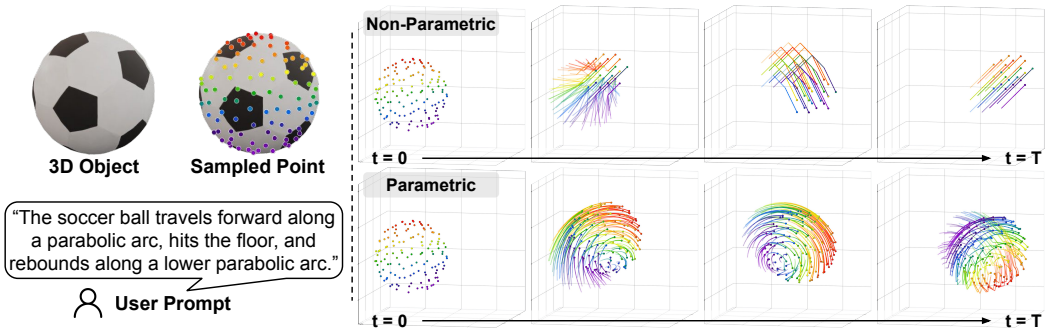

Figure 3: Effect of parametric motion planning. Direct prediction of future point positions produces irregular trajectories and distorts the object structure (top). Parametric functions yield coherent trajectories that better preserve spatial relationships between points (bottom). Colors identify corresponding points across time.

3.1.1 Agent-Based Motion Planning

Automatic semantic injection Point coordinates alone provide limited semantic information about the object parts relevant to the text X. Let $\mathcal{C} = \{\mathbf{c}_k\}_{k=1}^{K}$ denote control points sampled from $\mathcal{G}$. For each initial scene view I^v, we project the points using the camera projection π^v and annotate their locations as follows:

$$I^v_{\text{sem}} = \text{Annotate}\left(I^v, \{\pi^v(\mathbf{c}_k)\}_{k=1}^{K}\right), \qquad v = 1, \ldots, V. \tag{1}$$

Each marker displays the point's index k consistently across views, allowing the agent to associate its coordinates with visible object parts. We denote the annotated images by $\mathcal{I}_{\text{sem}} = \{I^v_{\text{sem}}\}_{v=1}^{V}$, where V is the number of views. We also provide the coordinate-system origin, axis directions, and spatial units. Examples of annotated inputs are provided in the Appendix.

Parametric motion planning Directly predicting point positions at each timestep causes the predicted motion to collapse over time, as shown in Figure 3. To address this issue, we instead ask the agent to infer mathematical functions and their parameters. Since a single function may be insufficient to capture different motion stages, we instruct the agent to divide the time horizon into temporal chunks and specify a function for each. The agent $\mathcal{A}$ constructs a motion plan as follows:

$$\Pi = \mathcal{A}\left(X, \mathcal{C}, \mathcal{I}_{\text{sem}}\right), \qquad \Pi = \{(\phi_j, t_{j-1}, t_j)\}_{j=1}^{J}, \tag{2}$$

where J denotes the number of temporal chunks, ϕ_j describes the motion in the j-th chunk, and $0 = t_0 < \cdots < t_J = T$. For each sampled point $\mathbf{c}_k \in \mathcal{C}$, its target position at time t is

$$\bar{\mathbf{p}}_k(t) = \phi_j(\mathbf{c}_k, t), \qquad t \in [t_{j-1}, t_j), \tag{3}$$

where the final chunk includes $t = T$. Evaluating these functions over time yields target trajectories for physics optimization.

3.1.2 Physics Optimization

MPM-based physics simulation We use the agent-generated trajectories as pseudo supervision for a differentiable Material Point Method (MPM) simulator. We optimize simulation parameters $\boldsymbol{\theta}$, including object-level material parameters and per-particle initial velocities, to follow these trajectories. Let $\mathbf{s}^n$ denote the MPM state at timestep n, including particle positions, velocities, and deformation-related quantities. Starting from the initial scene geometry $\mathcal{G}$, the state evolves as

$$\mathbf{s}^{n+1} = \mathcal{S}_{\text{MPM}}\left(\mathbf{s}^n; \boldsymbol{\theta}, \Delta t\right), \tag{4}$$

where $\mathcal{S}_{\text{MPM}}$ denotes the differentiable MPM update and Δt is the simulation timestep. We denote the initial MPM particle positions by $\mathbf{p}_i^0$ and their simulated positions at time $t_n = n\Delta t$ by $\mathbf{q}_i^n(\boldsymbol{\theta})$, where i indexes simulation particles.

Table 1: Examples of motion related masking for Motion-IG. Bold words in the original text X are replaced by [MASK] in X_m, while object and part names are retained.

Text	Instruction
X	The **soft** fox **sways** from **side** to side while **remaining attached** at the rear.
X_m	The [MASK] fox [MASK] from [MASK] to side while [MASK] [MASK] at the rear.
X	The basketball **rolls forward** in a **straight line**, **rotating naturally without slipping**.
X_m	The basketball [MASK] [MASK] in a [MASK] [MASK], [MASK] [MASK] [MASK] [MASK].
X	The **rigid** playdoh ball **falls straight down** and **bounces several times** without **deforming**.
X_m	The [MASK] playdoh ball [MASK] [MASK] [MASK] and [MASK] [MASK] [MASK] without [MASK].

Motion reconstruction loss Following Eq. 3, we obtain each target position as $\bar{\mathbf{p}}_i^n = \phi_j(\mathbf{p}_i^0, t_n)$ using the chunk containing t_n. We optimize the simulation parameters by minimizing the discrepancy between the simulated and target positions as follows:

$$\mathcal{L}_{\text{motion}}(\boldsymbol{\theta}) = \frac{1}{N_p N_t} \sum_{i=1}^{N_p} \sum_{n=1}^{N_t} \|\mathbf{q}_i^n(\boldsymbol{\theta}) - \bar{\mathbf{p}}_i^n\|_2^2, \tag{5}$$

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{motion}}(\boldsymbol{\theta}), \tag{6}$$

where N_p and N_t denote the numbers of particles and timesteps used to compute the loss, respectively. During optimization, the motion plan remains fixed while gradients propagate through the MPM rollout to update $\boldsymbol{\theta}$.

3.2 Motion-Conditioned Information Gain

We introduce Motion-Conditioned Information Gain (Motion-IG) to measure text–motion alignment through masked text recovery. It measures how much observing the simulated motion improves recovery of motion-related content beyond the remaining text.

Notation Let X_m denote the masked text and $X^{\mathcal{M}}$ the content removed at positions $\mathcal{M}$. We mask the core behavior and its attributes while retaining object and part names, as illustrated in Table 1. Let Y denote the simulated motion, represented by the rendered rollout and serialized 3D trajectories. The masked text and initial scene image I remain identical in both conditions. For brevity, we omit the shared conditioning on I in the equations below.

Information gain The masked content may be partially inferred from the remaining text alone. We therefore measure the additional information provided by the motion through conditional mutual information as follows:

$$I(X^{\mathcal{M}}; Y \mid X_m) = \mathbb{E}\left[\log \frac{p(X^{\mathcal{M}} \mid X_m, Y)}{p(X^{\mathcal{M}} \mid X_m)}\right]. \tag{7}$$

Since the underlying distributions are unknown, we use a fixed vision-language model p_ψ to compute a pointwise information score as follows:

$$s_\psi(X, Y) = \log \frac{p_\psi(X^{\mathcal{M}} \mid X_m, Y)}{p_\psi(X^{\mathcal{M}} \mid X_m)}. \tag{8}$$

The denominator provides the recovery likelihood without the simulated motion. A higher score indicates that observing the motion increases the likelihood of the masked content by a larger factor. The score can be negative when the motion reduces this likelihood.

Score computation We provide the masked text as the query and teacher-force the original text as the response. For the i-th text–motion pair (X_i, Y_i) among N evaluated pairs, let $(z_{i,1}, \ldots, z_{i,L_i})$ denote the response tokens, where L_i is the sequence length. The change in log-probability at position ℓ is given by

$$\delta_\psi(i, \ell) = \log p_\psi\left(z_{i,\ell} \mid X_{m,i}, Y_i, z_{i,<\ell}\right) - \log p_\psi\left(z_{i,\ell} \mid X_{m,i}, z_{i,<\ell}\right), \tag{9}$$

where the preceding response tokens $z_{i,<\ell}$ are identical in both conditions.

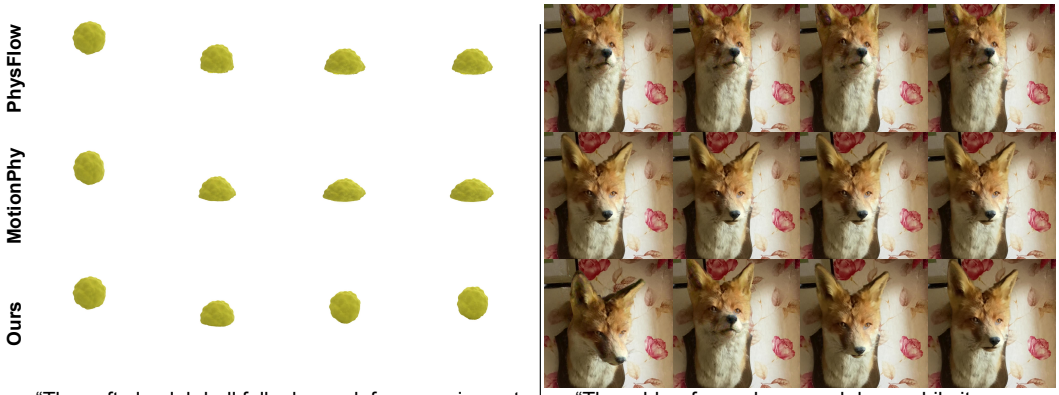

Figure 4: Qualitative comparison of text-controllable physics simulation. AgentDreamer more closely follows the requested motion than PhysFlow and MotionPhysics in the shown examples. These results illustrate the benefit of explicit agentic motion planning over 2D video-based guidance in capturing the motion specified by the instructions.

Let $\mathcal{W}_i$ denote the words selected from $X_i^{\mathcal{M}}$ for scoring, excluding words such as "is" and "the" but retaining negations such as "not". For each word w, we score its first subword token at position $\ell_i(w)$. We average these gains over selected words within each text and then over all texts:

$$\text{Motion-IG} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|\mathcal{W}_i|}\sum_{w\in\mathcal{W}_i}\delta_\psi\big(i,\ell_i(w)\big). \tag{10}$$

We compute this score for each method using its simulated motions as Y_i, while keeping the texts, masks, initial scene images, and evaluation models identical across methods.

4 Experiments

We evaluate AgentDreamer on text-controllable physics simulation with instructions covering diverse object motions. We use GPT as the off-the-shelf agent for 3D motion planning throughout our experiments. We compare the generated simulations with existing methods using Motion-IG and human evaluation, considering both text alignment and physical plausibility.

4.1 Experimental Setup

Scenes and instructions We conduct experiments on eight scenes with 20 text instructions. The dataset comprises four real-world scenes used by PhysFlow (Liu et al., 2025) (Alocasia, Carnation, Telephone and Fox) and four synthetic scenes, one from PAC-NeRF (Li et al., 2023) (Playdoh) and three from GaussianVerse (Xiang et al., 2025) (Basketball, Soccer Ball and Can). The instructions cover falling and bouncing, rolling, slowing rotation, oscillation, and bending or deformation, with 10 applied to the real-world scenes and 10 to the synthetic scenes.

Evaluation protocols We compare AgentDreamer with PhysFlow (Liu et al., 2025) and MotionPhysics (Wang et al., 2026), which use text-conditioned video diffusion models to enable text control over physics simulation. We measure the text alignment of simulated motion using the proposed Motion-IG with four open-weight evaluators: InternVL3.5-8B (Wang et al., 2025), Qwen3-VL-8B (Qwen Team, 2025), Qwen3.5-9B (Qwen Team, 2026), and Gemma-4-12B (Team et al., 2026). We additionally report results with five API-based evaluators: GPT-5.6 Sol and GPT-5.6 Terra (OpenAI, 2026a), GPT-6 Astra (OpenAI, 2026b), Gemini 3.1 Pro (Google DeepMind, 2026a), and Gemini 3.8 Flash (Google DeepMind, 2026b); since these models expose no teacher-forced log-probabilities, we query each masked word's probability in words and use the stated value in place of p_ψ in Eq. 8. For human evaluation, participants see three blinded simulations of the same instruction and rank them by how well the motion matches the requested behavior.

Table 2: Comparison of text–motion alignment using Motion-IG with (a) token likelihoods from open-weight evaluators and (b) verbalized probabilities from black-box API evaluators (higher is better). **Best in** counts the evaluator–instruction pairs in which each method ranks first.

(a) Open-weight evaluators

Method	Gemma-4	Qwen3.5	Qwen3-VL	InternVL3.5	Average	Best in
PhysFlow (Liu et al., 2025)	0.36	0.62	0.58	0.52	0.52	15/80
MotionPhysics (Wang et al., 2026)	0.13	0.58	0.81	0.56	0.52	22/80
AgentDreamer (ours)	**0.63**	**0.67**	**0.92**	**0.64**	**0.71**	**43/80**

(b) Black-box evaluators

	GPT			Gemini			
Method	6 Astra	5.6 Sol	5.6 Terra	3.8 Flash	3.1 Pro	Average	Best in
PhysFlow (Liu et al., 2025)	−0.13	−0.45	+0.04	−0.69	+0.14	−0.22	23/100
MotionPhysics (Wang et al., 2026)	−0.49	−0.96	−0.44	−0.93	−0.32	−0.63	15/100
AgentDreamer (ours)	**+0.56**	**+0.25**	**+0.48**	**+0.12**	**+1.00**	**+0.48**	**62/100**

4.2 Comparison with Existing Methods

4.2.1 Qualitative Results

Figure 4 compares the simulated motion of AgentDreamer, PhysFlow, and MotionPhysics under the same text instructions. In the shown examples, both baselines generate motion that differs from the requested behavior, whereas AgentDreamer more closely follows the description. For the deformable bouncing instruction, PhysFlow and MotionPhysics produce deformation upon impact but remain near the ground without a clear rebound. In contrast, AgentDreamer generates impact deformation followed by an upward rebound, consistent with the instruction. These examples illustrate how explicit 3D motion priors help preserve the motion specified by the text during physics optimization.

4.2.2 Quantitative Results

Motion-IG with open-weight evaluators Table 2a compares text–motion alignment using four open-weight evaluators. On the 20 evaluated instructions, AgentDreamer achieves the highest score on all four different evaluators. Its positive scores indicate that providing our simulated motion increases the probability assigned to the correct masked words, on average, compared with the text-only condition. These results suggest that motion generated by AgentDreamer provides more useful information for recovering the requested behavior than the compared methods.

Motion-IG with API-based evaluators Table 2b reports results using verbalized probabilities from five API-based evaluators. The lower average scores may partly reflect stronger recovery from text alone: if an evaluator already predicts the masked words with high probability, observing motion provides less additional gain. Nevertheless, AgentDreamer achieves the highest score under all five evaluators and ranks first in $62/100$ evaluator–instruction pairs. This consistent ranking suggests that our simulated motions provide more useful evidence for recovering the requested behavior across both evaluation approaches.

Human evaluation We conduct a human evaluation of text–motion alignment with 25 participants. For each instruction, participants view the results of all three methods and rank them by how well they match the requested behavior. AgentDreamer is ranked first most often (73.6%), compared with PhysFlow (14.0%) and MotionPhysics (12.4%). Figure 5 further shows that AgentDreamer is preferred over each baseline in pairwise comparisons. This agrees with the Motion-IG ranking, providing further evidence that AgentDreamer follows the text instructions more closely than the baselines.

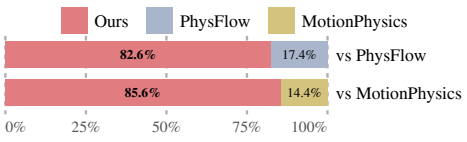

Figure 5: Pairwise human preference for text–motion alignment. Win rate is the percentage of trials in which one method is ranked above the other. AgentDreamer is preferred over PhysFlow in 82.6% of trials and over MotionPhysics in 85.6% of trials.

Table 3: Correspondence between requested motions and agent-generated function forms across 20 instructions. Expressions show representative components of the motion functions. **Matched / Total** reports the number of plans whose function forms match the expected behavior within each instruction group.

Motion cue	Function family	Representative expression	Matched / Total
Roll	Coupled translation and rotation	$d(\tau) = v\tau, \quad \theta(\tau) = (v/R)\tau$	2/2
Spin, slow down, stop	Quadratic	$\theta(\tau) = \omega_0\tau + \frac{1}{2}\alpha\tau^2, \quad \alpha < 0$	2/2
Fall, bounce, jump	Quadratic	$z(\tau) = z_0 + v_{z,0}\tau - \frac{1}{2}g\tau^2$	4/4
Sway, oscillate, rock	Sinusoid	$\theta(\tau) = Ae^{-\lambda\tau}\sin(\omega\tau)$	8/8
Pull, bend, squash, deform	Smoothstep interpolation	$s(u) = 3u^2 - 2u^3$	4/4

τ: elapsed time within a chunk; $u = \tau/(t_j - t_{j-1})$: normalized time.
d: travel distance; θ: rotation angle; R: ball radius.

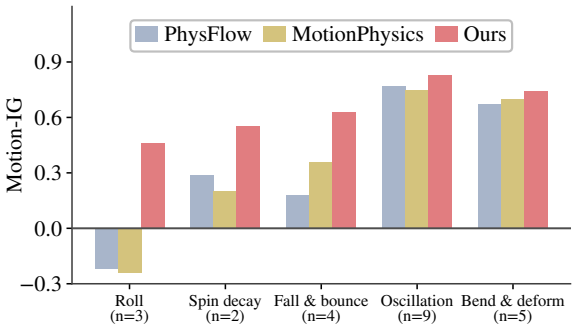

Figure 6: Text–motion alignment across motion categories measured by Motion-IG (higher is better). AgentDreamer achieves the highest score in every category; for rolling, both baselines have negative information gain.

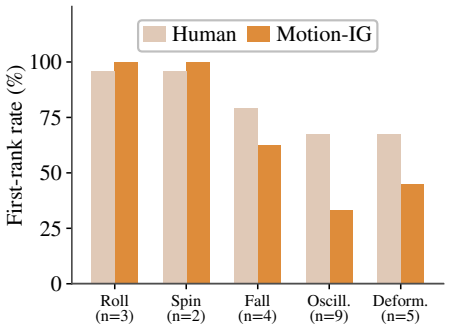

Figure 7: Consistency of Motion-IG with human evaluation across motion categories. Both show higher first-rank rates for rolling and spinning and lower rates for oscillation and deformation.

4.3 Analysis

Analysis of physical reasoning in motion planning. We examine whether the agent-generated plans reflect physical reasoning about the requested motion. We group the 20 instructions into five categories based on their characteristic motion patterns and inspect whether the generated functions encode the relevant physical relationships and temporal properties. Table 3 summarizes the function families observed in each category, with counts indicating how many plans use each family. The plans encode relevant physical relationships such as quadratic height functions for falling and bouncing. They also capture slowing rotation and decreasing oscillation amplitude through angular deceleration and damped sinusoids. These correspondences provide qualitative evidence of physical reasoning in motion planning.

Benefits of 3D motion planning across motion categories. To examine its benefit over 2D video-based guidance, we compare Motion-IG across five motion categories in Figure 6. Our method achieves the highest score in every category, with the largest improvements for rolling, spin decay, and falling and bouncing. Notably, both baselines obtain negative scores for rolling, indicating that their simulated motions make the masked behavior harder to recover than from text alone. These gains suggest that directly planning trajectories in 3D helps capture behaviors whose realization depends on understanding the object's geometry and spatial motion. For oscillation and deformation, all three methods achieve relatively high scores and the improvements are smaller. Together, these results suggest that agent-generated 3D planning is particularly beneficial when reproducing the requested behavior requires reasoning about motion in 3D space.

Consistency with human evaluation. We examine whether Motion-IG reflects human preferences by comparing how often AgentDreamer ranks first among the evaluated methods in each motion category. As shown in Figure 7, both human evaluation and Motion-IG select our method most frequently for rolling and spinning. The first-rank rates decrease for oscillation and deformation, where the baselines are selected more often. Although the absolute rates differ, this shared

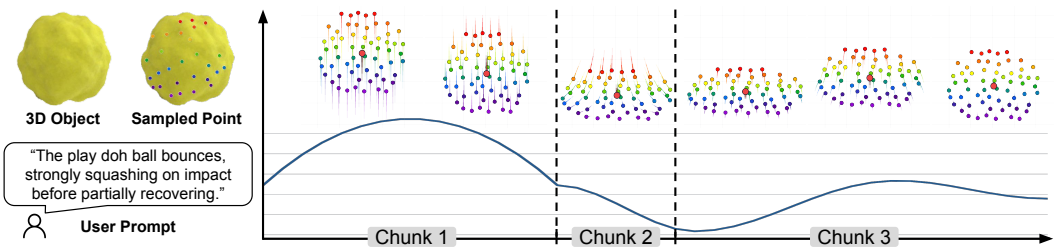

Figure 8: Temporal decomposition of an agent-generated motion plan. The agent uses separate parametric functions to describe the ball movement, compression on impact, and partial shape recovery. Dashed lines mark temporal chunks, and colored points show the predicted position over time.

Table 4: Ablation study with average score from open-weight evaluators. Parametric planning provides the largest individual gain, while combining both components achieves the highest score.

Semantic injection	Parametric planning	Motion-IG ↑
		0.37
✓		0.38
	✓	0.63
✓	✓	**0.72**

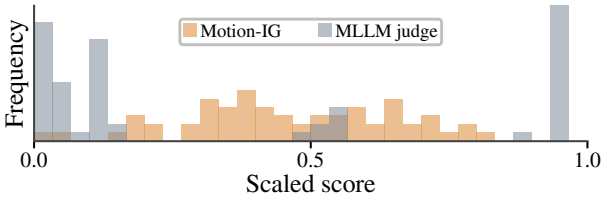

Figure 9: Score distributions on identical text–motion pairs, scored by gemma-4. Our conditional metric more clearly separates score; MLLM judge assigns overlapping scores to the two groups.

trend suggests that Motion-IG captures category-dependent differences in text-motion alignment observed by human evaluators.

Visualization of temporal decomposition of motion planning Figure 8 shows how the agent divides the requested motion into temporal chunks and constructs a parametric function for each. The predicted sequence captures the ball's motion, strong compression on impact, and subsequent partial shape recovery. The chunks correspond to these behavioral changes, suggesting that the agent identifies distinct motion stages within the instruction. By describing each stage separately, the plan captures changes in physical motion that are difficult to represent with a single parametric function.

Ablation study Table 4 shows that both components contribute to text alignment. Removing automatic semantic injection substantially lowers the score, suggesting that explicit point annotations help the agent associate scene geometry with meaningful object parts. Without parametric planning, the predicted motion loses coherence shortly after the initial movement. This suggests that inferring a motion function encourages the agent to reason about the underlying physical behavior. The resulting function also provides dense trajectories over the target horizon through direct evaluation.

Comparison with MLLM judge We compare Motion-IG with an MLLM judge (Zheng et al., 2023; Chen et al., 2024) that states how likely the instruction and the simulated motion match as a number in $[0, 1]$, taken greedily as its verbalized confidence. Figure 9 compares the two scores on the same 60 text-motion pairs, each scaled to its own range. Although a real number is asked for, the judge's confidence takes only 11 distinct values out of 60 (2.5 bits of entropy over 20 equal bins, against 4.0 for Motion-IG, which never ties two samples). This leaves little separation between simulations and limits the resolution of their ranking.

5 Conclusion

We presented AgentDreamer, a framework that uses an agent to generate 3D motion priors for text-controllable physics simulation. Given a text instruction and scene geometry, the agent reasons about the underlying physics and constructs parametric functions to describe motion consistent with the instruction. The resulting trajectories guide differentiable physics optimization to reproduce the requested motion. We have also introduced Motion-IG to measure how much simulated motion helps recover masked motion words beyond the remaining text. Experiments show improved text alignment over video-guided baselines, demonstrating the effectiveness of agentic motion priors.

AI USE STATEMENT

In this work, we used generative AI tools to help develop theoretical models or conceptual frameworks, design or provide feedback on research methodology or experiments, implement methods, and assist with translation. We have not used generative AI tools to generate synthetic datasets, formulate mathematical claims, clean or reformat datasets, or interpret results, and tasks related to providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, and supporting qualitative or thematic data analysis are not applicable to this work.

Additionally, we used generative AI tools to create or modify scientific figures or images, suggest experimental parameters, create or edit software code and research artifacts, draft parts of the manuscript, summarize or analyze existing literature, discover research topics or identify gaps, brainstorm research ideas, search for information, improve manuscript readability, identify relevant literature, format references, and suggest manuscript structure.

We have reviewed all AI-assisted work. AI-assisted code was verified through execution and testing, and all methodological suggestions and translations were manually reviewed by the authors before use. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses existing 3D scenes to study text-controlled physical simulation. The human evaluation involved 25 participants who ranked simulated motions by their alignment with text instructions. Method identities were hidden as A, B, and C, their presentation positions were balanced across instructions, and results are reported in aggregate. Because the motion plans and automated evaluations rely on pretrained multimodal models, they may reflect model errors or biases. We therefore report human judgments alongside model-based scores and do not treat the latter as a sole measure of physical correctness.

REPRODUCIBILITY STATEMENT

Sections 3.1 and 3.2 describe the motion-planning and physics-optimization procedure and define the Motion-IG metric. Section 4.1 identifies the scene sources, text instructions, baselines, and evaluation models. The appendix provides additional results (Appendices A and B), examples of semantic injection (Appendix C), the prompts used for motion planning, code generation, and evaluation (Appendix D), and the human evaluation procedure (Appendix E).

REFERENCES

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

Junhao Cai, Yuji Yang, Weihao Yuan, Yisheng He, Zilong Dong, Liefeng Bo, Hui Cheng, and Qifeng Chen. Gic: Gaussian-informed continuum for physical property identification and simulation. *Advances in Neural Information Processing Systems*, 2024.

Yuanhao Cai, Kunpeng Li, Menglin Jia, Jialiang Wang, Junzhe Sun, Feng Liang, Weifeng Chen, Felix Juefei-Xu, Chu Wang, Ali Thabet, et al. Phygdpo: Physics-aware groupwise direct preference optimization for physically consistent text-to-video generation. In *ECCV*, 2026.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. MLLM-as-a-judge: Assessing multimodal LLM-as-a-judge with vision-language benchmark. In *ICML*, 2024.

Google DeepMind. Gemini 3.1 Pro: A smarter model for your most complex tasks, 2026a. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/.

Google DeepMind. Gemini 3.8 Flash, 2026b. URL https://ai.google.dev/gemini-api/docs/latest-model.

Dadong Jiang, Zhi Hou, Zhihui Ke, Xianghui Yang, Xiaobo Zhou, and Tie Qiu. Timeformer: Capturing temporal relationships of deformable 3d gaussians for robust reconstruction. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8721–8732. IEEE, 2025a.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos. In *ICCV*, 2025b.

Kai Katsumata, Duc Minh Vo, and Hideki Nakayama. A compact dynamic 3d gaussian representation for real-time dynamic view synthesis. In *ECCV*, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 2023.

Junoh Lee, ChangYeon Won, Hyunjun Jung, Inhwan Bae, and Hae-Gon Jeon. Fully explicit dynamic gaussian splatting. *NeurIPS*, 2024.

Xuan Li, Yi-Ling Qiao, Peter Yichen Chen, Krishna Murthy Jatavallabhula, Ming Lin, Chenfanfu Jiang, and Chuang Gan. Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *ICLR*, 2023.

Yuhao Li, Mirana Claire Angel, Salman Khan, Yu Zhu, Jinqiu Sun, Yanning Zhang, and Fahad Shahbaz Khan. C-drag: chain-of-thought driven motion controller for video generation. *arXiv preprint arXiv:2502.19868*, 2025.

Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *CVPR*, 2024a.

Yuchen Lin, Chenguo Lin, Jianjin Xu, and Yadong Mu. Omniphysgs: 3d constitutive gaussians for general physics-based dynamics generation. In *ICLR*, volume 2025, 2025.

Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *ECCV*, 2024b.

Fangfu Liu, Hanyang Wang, Shunyu Yao, Shengjun Zhang, Jie Zhou, and Yueqi Duan. Physics3d: Learning physical properties of 3d gaussians via video diffusion. *arXiv preprint arXiv:2406.04338*, 2024.

Zhuoman Liu, Weicai Ye, Yan Luximon, Pengfei Wan, and Di Zhang. Unleashing the potential of multi-modal foundation models and video diffusion for 4d dynamic physical scene simulation. In *CVPR*, 2025.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024.

Fanqing Meng, Jiaqi Liao, Xinyu Tan, Wenqi Shao, Quanfeng Lu, Kaipeng Zhang, Yu Cheng, Dianqi Li, Yu Qiao, and Ping Luo. Towards world simulator: Crafting physical commonsense-based benchmark for video generation. *arXiv preprint arXiv:2410.05363*, 2024.

Sriram Narayanan, Ziyu Jiang, Srinivasa Narasimhan, and Manmohan Chandraker. Phyco: Learning controllable physical priors for generative motion. In *CVPR*, 2026.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition, 2026a. URL https://openai.com/index/gpt-5-6/.

OpenAI. GPT-6 Astra: A new generation of intelligence, 2026b. URL https://openai.com/index/gpt-6-astra/.

Qwen Team. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, 2026. URL https://qwen.ai/blog?id=qwen3.5.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Miaowei Wang, Jakub Zadrożny, Oisin Mac Aodha, and Amir Vaxman. Motionphysics: Learnable motion distillation for text-guided simulation. In *AAAI*, 2026.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024.

Aaron D Wyner. A definition of conditional mutual information for arbitrary ensembles. *Information and Control*, 1978.

Tiange Xiang, Kai Li, Chengjiang Long, Christian Häne, Peihong Guo, Scott Delp, Ehsan Adeli, and Li Fei-Fei. Repurposing 2d diffusion models with gaussian atlas for 3d generation. In *ICCV*, 2025.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *CVPR*, 2024.

Xindi Yang, Baolu Li, Yiming Zhang, Zhenfei Yin, Lei Bai, Liqian Ma, Zhiyong Wang, Jianfei Cai, Tien-Tsin Wong, Huchuan Lu, et al. Vlipp: Towards physically plausible video generation with vision and language informed physical prior. In *ICCV*, 2025.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024.

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. In *ECCV*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *NeurIPS Datasets and Benchmarks Track*, 2023.

Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and simulation of elastic objects with spring-mass 3d gaussians. In *ECCV*, 2024.

A Additional Qualitative Results

Figure 10 and Figure 11 present additional comparisons of PhysFlow, MotionPhysics and AgentDreamer under the same text instructions. In the shown examples, AgentDreamer produces motion that more closely follows the instructions than the baselines. These results complement the main-paper comparisons and further support the use of agent-generated 3D motion priors for text-controllable physics simulation.

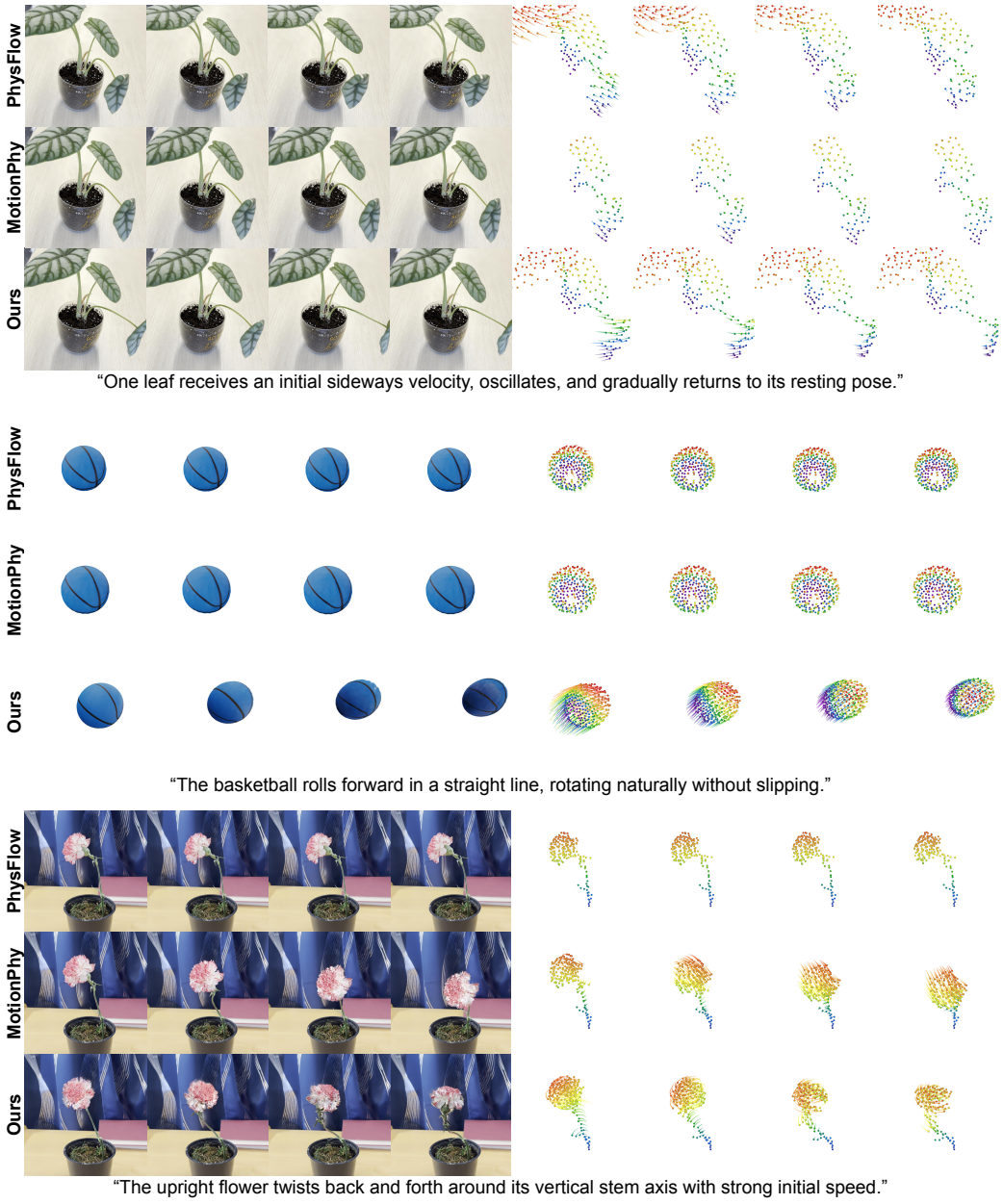

Figure 10: Additional qualitative comparisons with PhysFlow (Liu et al., 2025) and MotionPhysics (Wang et al., 2026) under the same text instructions. The left panels show frames from the simulated videos, while the right panels show the corresponding control-point trajectories. In the shown examples, AgentDreamer produces simulated motion that more closely follows the requested behavior.

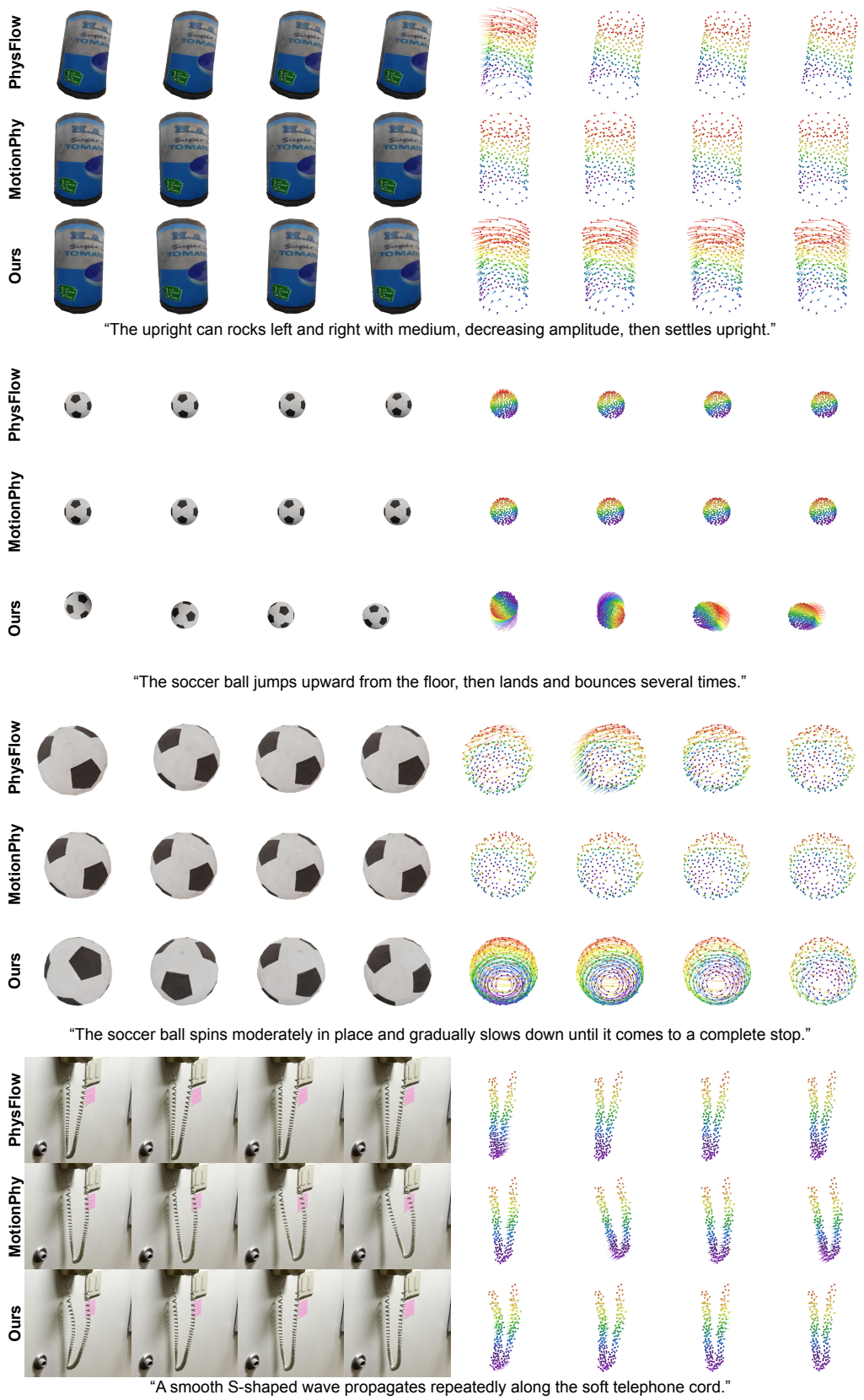

Figure 11: Additional qualitative comparisons with PhysFlow (Liu et al., 2025) and MotionPhysics (Wang et al., 2026) under the same text instructions. The left panels show frames from the simulated videos, while the right panels show the corresponding control-point trajectories.

B Additional Quantitative Results

Text–motion alignment across motion categories Table 5 breaks the Motion-IG scores down by the motion categories of the main paper. AgentDreamer ranks first in every category under both open-weight and black-box evaluators. The gap is clear for simple rigid-body motions governed by a single physical law, namely rolling, falling and bouncing, and spin decay: on rolling, AgentDreamer reaches $+0.46$ with open-weight evaluators while both baselines stay negative (-0.22 and -0.24). This is where the benefit of planning shows most, since the plan can state the motion by its governing law. On more complex motions that repeat or change shape, oscillation and bending and deformation, AgentDreamer still scores highest but by a narrower margin; with black-box evaluators it reaches $+0.35$ on oscillation, close to PhysFlow at $+0.33$.

Table 5: Text–motion alignment per motion category measured by Motion-IG (higher is better), averaged over four open-weight and five black-box evaluators.

		Open-weight (avg.)			Black-box (avg.)		
Category	n	**PhysFlow**	**MotionPhysics**	**Ours**	**PhysFlow**	**MotionPhysics**	**Ours**
Fall and bounce	4	$+0.18$	$+0.36$	$\mathbf{+0.63}$	-1.46	-0.87	$\mathbf{+0.77}$
Roll	3	-0.22	-0.24	$\mathbf{+0.46}$	-1.23	-2.01	$\mathbf{+0.92}$
Spin decay	2	$+0.29$	$+0.20$	$\mathbf{+0.55}$	-0.15	-0.23	$\mathbf{+0.62}$
Oscillation	9	$+0.77$	$+0.75$	$\mathbf{+0.83}$	$+0.33$	-0.50	$\mathbf{+0.35}$
Bend and deform	5	$+0.67$	$+0.70$	$\mathbf{+0.74}$	$+0.34$	-0.12	$\mathbf{+0.39}$

Human evaluation details We collect rankings from 25 participants on 20 instructions, for 500 trials in total. In each trial, participants read the instruction and watch the results of the three methods, stacked vertically and labeled A, B, and C without method names. The position of each method is balanced across instructions. Participants rank the three results from first to third by how well the motion matches the instruction. Table 6 reports how often each method receives each rank. AgentDreamer is ranked first in 73.6% of trials and last in only 5.4%, whereas PhysFlow and MotionPhysics are ranked last in 40.2% and 54.4% of trials, respectively.

Table 6: Rank distribution in the human evaluation (25 participants $\times$ 20 instructions $=$ 500 trials). Each entry is the percentage of trials in which the method receives the rank.

Method	**1st (%)** $\uparrow$	**2nd (%)**	**3rd (%)** $\downarrow$
PhysFlow	14.0	45.8	40.2
MotionPhysics	12.4	33.2	54.4
Ours	**73.6**	21.0	**5.4**

C SEMANTIC INJECTION EXAMPLES

Figure 12 presents examples of annotated images for two objects. The numerical identifiers show where the sampled control points appear in each initial scene image, helping the agent associate the points with visible object parts relevant to the text instruction.



Figure 12: Examples of annotated images used for automatic semantic injection. Control points sampled from the initial scene geometry are marked with numerical identifiers in two views of a fox (top) and a soccer ball (bottom).

D Details of Agent Prompts

D.1 Prompts for Motion planning

Figure 13 presents the prompt used to construct the motion plan. The agent groups the sampled control points, divides the time horizon into temporal chunks, and specifies a center function and a local function for each group in each chunk. Figure 14 presents the prompt used to translate the resulting motion plan into deterministic Python code. The code computes target point positions from the specified functions without modifying the motion plan.

```
Create an explicit grouped parametric 3D motion plan from the supplied object data.

Inputs:
- Clean multi-view images: {{IMAGES}}
- Matching point-ID overlay images: {{OVERLAY_IMAGES}}
- Point and camera data: {{POINTS_JSON}}
- Motion instruction: {{MOTION_PROMPT}}
- Maximum frame: {{MAX_FRAMES}}

Treat the clean images and overlay images as multimodal image inputs. Match each
clean image to its overlay using the camera entries and filenames in points.json.
Use them to understand the scene geometry, semantic parts, coordinate system,
and requested motion.

Point-group rules:

- Partition every supplied point ID into exactly one point group.
- Do not omit or duplicate any point ID.
- Keep the same point groups throughout the complete timeline.
- Group points that share one coherent center-motion and local-motion function.
- Prefer a small number of meaningful groups instead of one group per point.
- Use the clean and indexed multi-view images to distinguish moving material
  from static support or background.
- A static group must use motion_state static, function_type none, function
  none, empty parameter arrays, and composition p_i(tau)=p_i,start.

Timeline rules:

- Interpret {{MAX_FRAMES}} as the final frame index; frame 0 is the input state.
- Treat frame 0 through frame 40 as exactly 2.0 seconds of motion: 40 equal
  frame intervals and 41 trajectory states.
- Use dt_seconds = 0.05 seconds per frame interval, so frame f corresponds to
  physical time t_seconds = f * dt_seconds.
- Create a new segment whenever any point group's center-motion or local-motion
  function changes.
- Adjacent segments must share a boundary and together cover frame 0 through
  MAX_FRAMES.
- Set state_reference to initial_state for the first segment and
  previous_segment_end for every later segment.
- Every segment must contain exactly one group_motion for every declared group.

Function rules:

- Define center motion, local motion, and composition independently per group.
- Use local frame offset tau = f - start_frame; tau is an integer frame offset,
  not a time in seconds.
- For every time-dependent center or local function, declare
  dt_seconds = 0.05 as a numerical parameter and use the local physical time
  dt_seconds * tau explicitly in the function.
- Express linear velocity in normalized-coordinate units per second, angular
  velocity in radians per second, and acceleration in normalized-coordinate
  units per second squared.
- Let p_i,start be the group member's point position at segment start.
- The same group function is evaluated separately with each member's p_i,start.
- Every moving group function must be explicit, plain ASCII, single-line, and
  use only finite numerical parameters declared as name/value records.
- A moving center function returns absolute center c(tau); a moving local
  function returns relative q_i(tau); composition is p_i(tau)=c(tau)+q_i(tau).
- Every group function at tau = 0 must reproduce p_i,start exactly.
- Preserve continuity between adjacent segments.
- Do not output frame-by-frame coordinates.

Return valid JSON only, with no Markdown or commentary, in this shape:

{
 "point_groups": [
  {
   "group_id": "stable_group_id",
   "semantic_part": "Brief semantic description.",
   "point_ids": ["p000", "p001"]
  }
 ],
 "segments": [
  {
   "start_frame": 0,
   "end_frame": 40,
   "state_reference": "initial_state",
   "group_motions": [
    {
     "group_id": "stable_group_id",
     "motion_state": "moving",
     "motion_description": "Brief motion explanation.",
     "center_motion": {
      "function_type": "smoothstep",
      "function": "c(tau)=...",
      "parameters": [{"name": "parameter_name", "value": 0.0}]
     },
     "local_motion": {
      "function_type": "smoothstep",
      "function": "q_i(tau)=...",
      "parameters": [{"name": "parameter_name", "value": 0.0}]
     },
     "composition": "p_i(tau)=c(tau)+q_i(tau)"
    }
   ]
  }
 ]
}
```

Figure 13: Prompt for agent-based motion planning. The agent divides the time horizon into temporal chunks and specifies parametric motion functions for groups of sampled control points.

Translate the supplied grouped parametric motion plan into deterministic Python code.

Inputs:
- Point and camera data: {{POINTS_JSON}}
- Grouped parametric motion plan: {{PARAMETRIC_MOTION_PLAN}}
- Motion-plan SHA-256: {{PLAN_SHA256}}
- Segment count: {{SEGMENT_COUNT}}
- Group count: {{GROUP_COUNT}}
- Group index table: {{GROUP_INDEX_TABLE}}
- Maximum frame: {{MAX_FRAMES}}

The validated plan is final. Do not change group membership, functions, parameters, segment boundaries, motion states, directions, or compositions.

Generate exactly one private helper for every segment/group pair:

```
def _segment_SEGMENT_group_GROUP(
    point_id: str,
    segment_start_xyz: tuple[float, float, float],
    tau: int,
) -> tuple[float, float, float]:
```

Generate exactly one public dispatcher:

```
def segment_position(
    segment_index: int,
    group_index: int,
    point_id: str,
    segment_start_xyz: tuple[float, float, float],
    tau: int,
) -> tuple[float, float, float]:
```

Requirements:

- Generate SEGMENT_COUNT multiplied by GROUP_COUNT private helpers.
- The dispatcher must select helpers using segment_index and group_index.
- For motion_state static, return segment_start_xyz unchanged for every tau.
- For motion_state moving, implement the supplied group center function, local function, parameters, and composition exactly.
- At tau zero every helper must reproduce segment_start_xyz within floating point precision.
- Do not chain segments or round coordinates; the trusted runner does both.
- Return exactly three finite Python numbers.
- If math functions are needed, use exactly `import math`.
- Call math functions only as public `math.NAME(...)` calls.
- The code may optionally import future annotations and may define literal constants.
- Do not use files, environment variables, network, subprocesses, dynamic imports, eval, exec, compile, classes, loops, recursion, tests, printing, or import-time computation.

Return a JSON object containing only plan_sha256 and source_code.

Figure 14: Prompt for translating the motion plan into deterministic Python code. The code implements the specified motion functions to compute target point positions.

D.2 PROMPTS FOR EVALUATION

Figure 15 presents the prompt given to the open-weight evaluators. The evaluator receives the initial render, the generated motion as a trajectory and as rendered frames, and the description with its motion words masked; the masked words are scored by teacher forcing rather than by sampling the completion. Figure 16 presents the prompt given to the black-box evaluators. Since current API models expose no teacher-forced log-probabilities, the evaluator states a probability for each masked word, and these ratings replace the log-probabilities in the same gain.

```
The image shows the object in its initial state.

[ the block below is present only in the query with the generated motion ]

Coordinates are in simulation units in a right-handed frame: {{AXES_NOTE}}.
Trajectory over 11 time steps: the object's center (mean of all 256 tracked surface
points), then 8 points on the object's surface, each given as id (x, y, z).
t=0.000s: center (x, y, z); p000 (x, y, z); ... ; p007 (x, y, z)
   ... one row every 0.2 s ...
t=2.000s: center (x, y, z); p000 (x, y, z); ... ; p007 (x, y, z)

Rendered video of the object's motion from t=0.200s to t=1.800s, 9 frames at
5 frames per second: <<video>>

[ end of the block ]

Below is a description of the object's motion. Some words are replaced with <MASK>;
each <MASK> stands for exactly one word.
"{{MASKED_DESCRIPTION}}"

Rewrite the description with every <MASK> filled in. Respond only with the completed
description.
```

Figure 15: Prompt for the open-weight evaluators. The masked words are scored by teacher forcing, and the same prompt without the motion gives the reference term.

```
The image shows the object in its initial state.

[ the block below is present only in the query with the generated motion ]

The images after the first one are renders of the object's motion at
t = 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8 s, in that order.

Coordinates are in simulation units in a right-handed frame: {{AXES_NOTE}}.
Trajectory over 11 time steps: the object's center (mean of all 256 tracked surface
points), then 8 points on the object's surface, each given as id (x, y, z).
t=0.000s: center (x, y, z); p000 (x, y, z); ... ; p007 (x, y, z)
   ... one row every 0.2 s ...
t=2.000s: center (x, y, z); p000 (x, y, z); ... ; p007 (x, y, z)

[ end of the block ]

Below is a description of the object's motion. Some words are replaced with <MASK>;
each <MASK> stands for exactly one word.
"{{MASKED_DESCRIPTION}}"

Proposed completion:
"{{ORIGINAL_DESCRIPTION}}"

For each word listed below, estimate the probability (0 to 1) that it is the original
word at its masked position, that is, that it correctly describes how the object
actually moved. Base your estimate only on the evidence given above. The proposed
completion is only one possible reading: many other words could fill each <MASK> and
still give a fluent sentence. Do not rate a word high just because the sentence reads
naturally. Consider which other words could fill the same position, and judge how
strongly the evidence favors this word over them.
Words: {{MASKED_WORDS}}
Respond only with a JSON object that maps each listed word to its probability.
```

Figure 16: Prompt for the black-box evaluators. Each masked word receives a stated probability, collected in one JSON response per query.

E Details of User Study

We conducted a user study to assess how closely the simulated motions follow their text instructions. For each of 20 instructions, participants viewed three videos of the same object, corresponding to PhysFlow, MotionPhysics, and AgentDreamer, and ranked them from first to third according to their text–motion alignment. At the beginning of the survey, we present the following instructions:

> *In each of the 20 questions, you will see three short videos (A, B, C) of the same object and a text description below them. The videos loop automatically. For each question, please rank the three videos 1st, 2nd, and 3rd: 1st is the one whose motion best matches the text, and 3rd is the one that matches it the least. Each rank can be used only once.*
>
> *Please judge only how well the motion follows the description: what moves, in which way, and how strongly. Do not judge image quality, lighting, or rendering artifacts. The key words in each description are highlighted in orange. Please watch all three videos at least once before answering.*

Figure 17 shows an example question presented to participants in the user study.

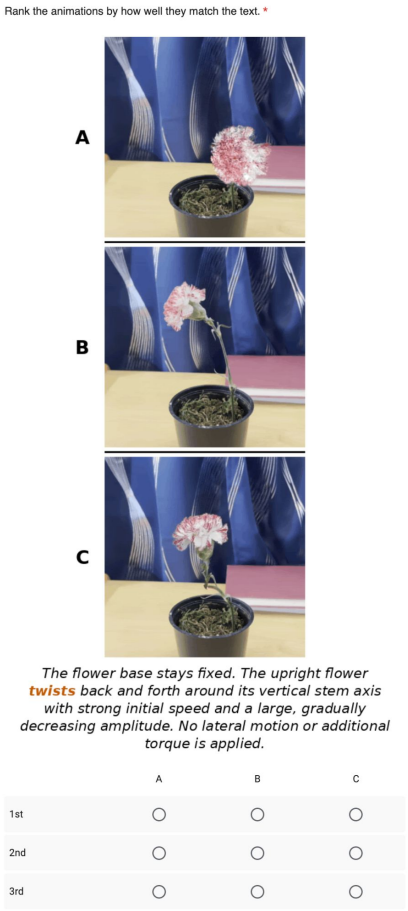

Figure 17: Example question in the human evaluation of text–motion alignment. Given a text instruction and three simulation results for the same object, participants rank the motions by how closely they follow the requested behavior.