Seeing the Unseen: Unified Visible–Invisible Motion for Physically Consistent Video Generation

Mingyang Bao[1], **Fangda Ye**[1], **Xihao Chen**[1], **Bobo Li**[1], **Jason Li**[2], **Shengqiong Wu**[3]*
[1]National University of Singapore, [2]Nanyang Technological University, [3]University of Oxford
Project page: https://vid-gen.github.io/seeing_the_unseen

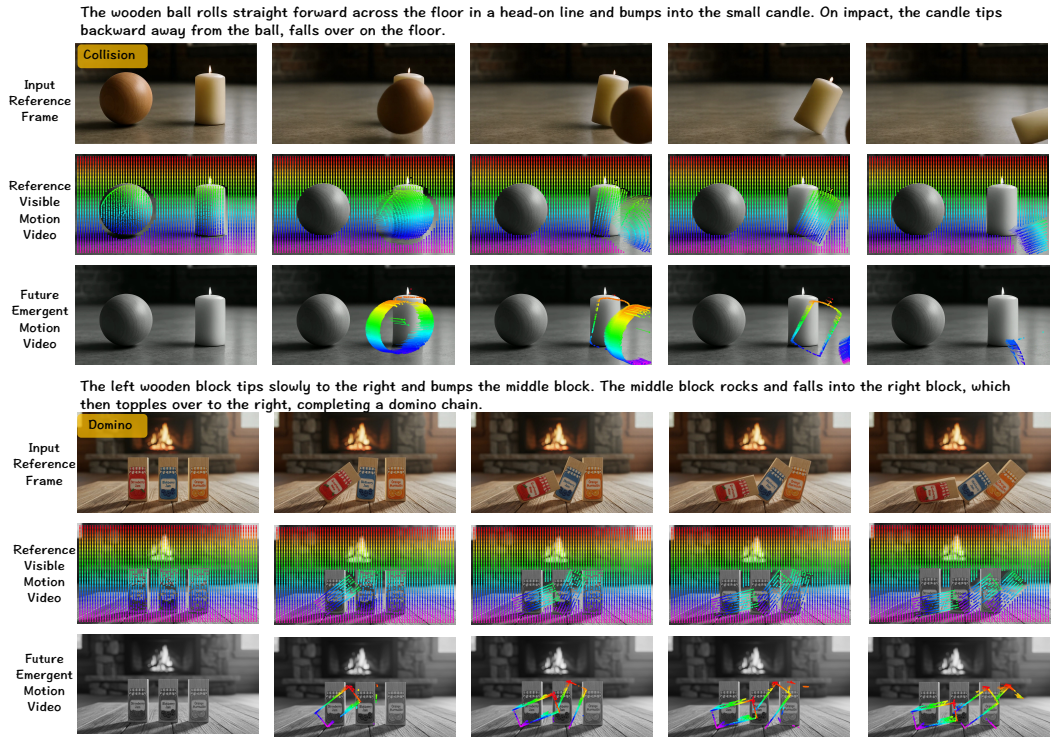

Figure 1: **Seeing the Unseen.** Our framework predicts physically consistent motion from a single image by modeling both reference-visible and future-emergent trajectories. By reasoning about occluded dynamics and structured interactions, it enables accurate simulation of complex events such as collisions and domino chains.

Abstract

Recent advances in video generation have achieved impressive visual quality, yet they often fail to produce physically consistent dynamics due to the lack of explicit modeling of underlying physical processes. Existing approaches either rely on simulators with strong assumptions or use coarse semantic guidance, and further struggle to effectively inject physical priors into generation models. In this work, we propose ***Seeing the Unseen***, a unified framework for physics-grounded video generation. Our approach decomposes the problem into two stages: learning physical dynamics and injecting them into video synthesis. First, we introduce

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

a lightweight *Particle-based Graph Dynamics Simulator (PGDS)* that learns generalizable physical interactions from data and predicts plausible 3D motion trajectories. Second, we propose a *Visible–Invisible Motion Field (VIMF)* that captures both motion observable in the input frame and motion that emerges over time due to object dynamics. This representation enables more complete and structured motion guidance compared to conventional physical signals. By integrating these trajectories into a diffusion-based generator, our method produces videos that are both visually coherent and physically consistent. Extensive experiments demonstrate improved motion accuracy, interaction consistency, and reduced physical artifacts compared to strong baselines.

1 Introduction

Video is a fundamental medium for recording dynamic processes in the real world. With the significant advancement of diffusion transformers [41], video generation [41, 61, 56, 49] has achieved remarkable progress in visual fidelity, temporal consistency, and complex scene synthesis. Recent production-level systems (e.g., Kling [47], Gen-3 [44], Pika [28], and Veo3.1 [13]) are capable of generating highly realistic dynamic videos and perform impressively in many everyday scenarios. However, the physical world is governed not only by visual appearance but also by underlying dynamics and interactions between objects, which follow well-defined physical laws. In contrast, existing video generation models predominantly learn pixel-level temporal statistics and lack explicit modeling of physical dynamics [4, 50]. As a result, they often produce physically implausible outcomes in scenarios involving complex interactions or causal relationships, such as object interpenetration, unrealistic motion, or unstable interactions [22, 10, 36, 3]. Improving physical consistency, therefore, requires the incorporation of effective physical priors [25, 31, 24]. Yet, current approaches remain limited in both the learning of such priors and their integration into video generation models.

One line of work [31, 8, 46, 58] learns physical priors through physics simulators [48, 1], which explicitly model the accurate physical constraints of rigid bodies, deformable objects, or fluids under controlled settings. However, they typically require precise 3D reconstruction, material estimation, and physical parameter specification, which are difficult to obtain in open-world visual scenarios. This strong dependence on accurate scene states and simulation parameters limits their scalability and generalization. To alleviate these requirements, some works [63, 50] adopt data-driven approaches to approximate physical dynamics, for instance, ordinary differential equations [63]. However, such methods rely on compact state representations (e.g., low-dimensional vectors), which limits their ability to capture complex interactions and multi-object dynamics. Another line of work exploits [64, 62, 24, 60, 2] the physical commonsense knowledge embedded in large vision-language models (LVLMs) [52, 39] to guide video generation at a semantic level. While such models can provide high-level physical reasoning, their guidance remains coarse-grained and language-centric. They do not directly characterize the fine-grained dynamics underlying object motion and interaction, making it difficult to impose stable and precise physical constraints during generation.

Another critical challenge is how to effectively incorporate them into video generation models. Existing methods often convert predicted motion into intermediate two-dimensional representations, such as optical flow [63, 50, 33, 58], tracking videos [24, 46], layouts [60], or sparse keypoint trajectories [21, 8], and use them as conditional signals. Although these representations can convey certain motion cues, they suffer from several limitations. First, projecting 3D trajectories into 2D inevitably discards depth and spatial structure, leading to compressed and incomplete physical information. Second, 2D motion conditions are usually restricted to regions visible from the reference view, and thus cannot describe future content that is initially occluded or unseen. Third, optical flow, tracking videos, and sparse keypoints only provide partial approximations of object motion, failing to preserve object-level and structure-level dynamics. These limitations weaken their ability to provide comprehensive physical guidance for video generation.

In this paper, we propose **Seeing the Unseen**, a unified framework for enhancing physically consistent video generation from two complementary perspectives: learning physical priors and injecting them through a richer motion representation. First, instead of relying on fully specified physics engines or purely semantic reasoning, we introduce a lightweight and learnable **Particle-based Graph Dynamics Simulator (PGDS)**, inspired by [45]. By representing dynamic scenes as particle-based systems, PGDS learns physical evolution directly from data, providing generalizable

and fine-grained priors for complex object interactions. Second, we propose a **Visible–Invisible Motion Field (VIMF)** derived from a simulated 3D motion trajectory, which jointly models both observable motion in the reference view and physically emergent motion in future unseen regions. VIMF preserves richer 3D dynamic information and provides more complete guidance to the video generation model, enabling it to synthesize motions that are not only visually coherent but also physically plausible. To support systematic evaluation, we further construct a benchmark for physics-aware video generation, covering diverse scenarios such as domino effects, free fall, collisions, and their compositional variants, with a total of 350 samples. Extensive experiments demonstrate that our method produces more accurate and stable motion trajectories, better preserves interaction consistency (e.g., contact and causality), and significantly reduces artifacts such as object interpenetration and unrealistic motion. Overall, our approach substantially improves physical plausibility compared to strong baselines.

Our contributions are summarized as follows:

- We introduce an efficient Particle-based Graph Dynamics Simulator to learn generalizable physical dynamics without manually specified physical parameters.
- We propose a Visible–Invisible Motion Field that jointly captures observable motion and physically predicted unseen motion, providing richer and more complete guidance for video generation.
- We build a benchmark for physics-aware video generation and demonstrate significant improvements in motion accuracy and physical plausibility over existing methods.

2 Related Work

Physically Grounded Video Generation. Recent efforts toward physically grounded video generation can be broadly categorized into three directions. Early approaches [31, 46, 8] rely on physics simulators to explicitly model object dynamics, including rigid body motion, deformation, and fluid behavior [48, 1]. While these methods provide structured physical signals, they are typically developed under controlled settings and depend on predefined scene representations. To improve scalability, subsequent works explore data-driven approximations of physical dynamics [63, 25, 50]. These methods relax the need for explicit simulation but often operate under constrained settings or specific scenario types, which limits their ability to generalize to complex multi-object interactions. More recently, a growing body of work [64, 2, 40] leverages the physical commonsense knowledge embedded in large vision-language models (LVLMs) [52, 39] or pre-trained video encoders [65, 25] to guide video generation. Such approaches typically introduce intermediate reasoning processes, such as predicting key motion trajectories [20, 60, 23] or generating keyframes [24], which are then used to guide video synthesis. In addition, some methods [6, 64] employ LVLMs as verifiers to assess physical consistency during or after generation (e.g., physical constraint checking or context verification). Despite these advances, existing approaches either rely on complex pipelines with limited scalability or provide coarse-grained guidance that lacks explicit modeling of underlying dynamics, making it challenging to enforce precise physical consistency.

Controllable Video Generation with Motion Guidance. Controlling motion in video generation has been extensively studied through various conditioning mechanisms. Some works [18, 38, 19, 17] incorporate motion cues through additional control branches to provide flexible and intuitive control over motion patterns. Another line of work integrates motion information directly into the generative process via attention-based mechanisms [21, 54, 29]. In addition, some methods [40, 12] explore implicit control strategies where motion guidance is encoded in textual prompts or high-level instructions. Despite these advances, existing methods predominantly rely on 2D motion representations and view-dependent signals. As a result, they often fail to capture complete 3D dynamics, particularly for regions that are initially occluded or become visible over time, restricting their ability to model complex interactions and provide physically consistent guidance for video generation.

Physical Dynamics Learning and Simulation. Modeling physical dynamics has been extensively studied in both classical simulation and learning-based frameworks. Traditional approaches rely on numerical solvers grounded in physical laws, such as finite element methods (FEM) [67], position-based dynamics (PBD) [35], material point methods (MPM) [26], smoothed particle hydrodynamics (SPH) [42], and mass-spring systems [32]. These methods provide accurate and interpretable solutions but typically require detailed scene representations and carefully specified physical parameters, limiting their applicability in data-driven visual settings. To improve flexibility, neural approaches have been proposed to approximate physical processes, such as PINNs [43] and fluids [11, 55].

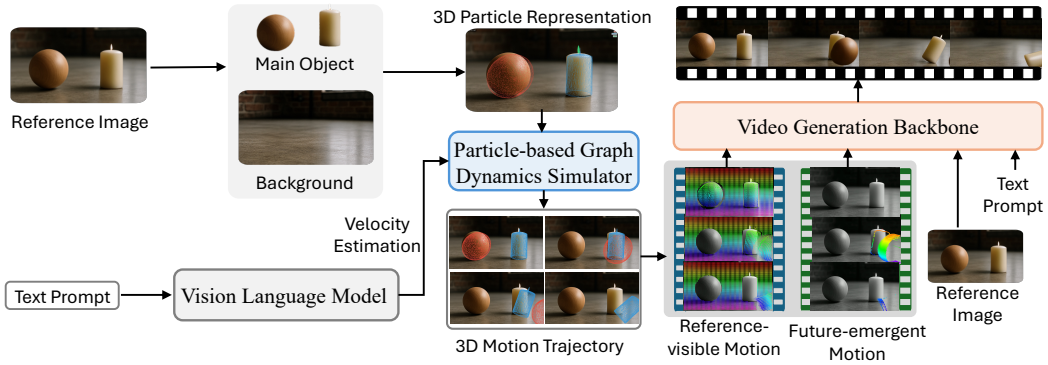

Figure 2: Overview of our proposed framework. Given an input image, we first lift it into a 3D scene particle representation. A particle-based graph dynamics simulator then predicts physically plausible 3D motion trajectories, which are rendered into reference-visible and future-emergent motions to guide physically consistent video generation.

Hybrid methods further integrate neural components into simulation pipelines to accelerate inference [15], though they often rely on structured representations or per-scene optimization. More recently, neural simulators based on graph neural networks have emerged as an effective paradigm for learning physical dynamics [45] by modeling interactions between particles or objects. Beyond these, recent works also explore diffusion models to model point-based physical dynamics [50], or neural Newtonian dynamics to predict low-dimensional latent states [63]. While these approaches improve flexibility, their compact representations struggle to capture complex multi-object interactions; we therefore adopt a particle-based graph simulator for more expressive physical modeling.

3 Method

3.1 Overview

To generate physically consistent videos, we propose a unified framework that jointly addresses physical prior learning and seamless motion-guided video synthesis. As illustrated in Figure 2, given an input image I, we first lift it into a simulatable 3D scene representation. Then, we employ a particle-based graph dynamics simulator (PGDS) to model object interactions and predict physically plausible motion trajectories in 3D space (cf. §3.2). The predicted 3D trajectories are then transformed into a unified motion representation that captures both observable motion and future motion in previously unseen regions in the given reference image. Finally, we inject this representation into a video generation model through a visible-invisible motion field (VIMF), enabling the synthesis of videos that are both visually coherent and physically plausible (cf. §3.3).

3.2 Learning Physical Dynamics with Graph-Based Simulation

Input Image Lifting. Given an input image $I \in \mathbb{R}^{H\times W\times 3}$, we first lift it into a simulatable 3D scene representation. Specifically, we begin by estimating a pixel-wise depth map and converting it into a metric 3D point map $\mathcal{B} = \{\boldsymbol{p}_i\}_{i=1}^{N}$, where $p_i \in \mathbb{R}^3$ denotes the i-th 3D position. We then segment foreground objects using SAM3 [7], obtaining instance-level binary masks. For each segmented object, we perform single-image 3D reconstruction using SAM 3D [9], conditioned on the predicted point map. This step yields object-level 3D representations with metric poses, encoding their positions and scales relative to the camera. We then transform all objects into a unified scene coordinate system and estimate the ground plane via RANSAC-based fitting [37, 66]. Finally, we aggregate all objects into a multi-object scene representation, which is then converted into a set of simulation particles, which serves as the input to our graph-based dynamics simulator.

Particle-based Graph Dynamics Simulator. Given simulation particles from the lifted 3D scene, we construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes represent particles and edges connect neighbors within a radius r. Each particle is associated with a feature $\mathbf{z}_i = [\mathbf{c}_i; \mathbf{o}_i; \mathbf{s}_i]$, encoding material attributes, object-relative geometry, and dynamic state. Node and edge features are encoded as $\mathbf{h}_i^v = \phi^v(\mathbf{z}_i)$ and $\mathbf{h}_{ij}^e = \phi^e([\mathbf{h}_i^v; \mathbf{h}_j^v; \mathbf{r}_{ij}; \|\mathbf{r}_{ij}\|])$, where $\mathbf{r}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. We perform T rounds of

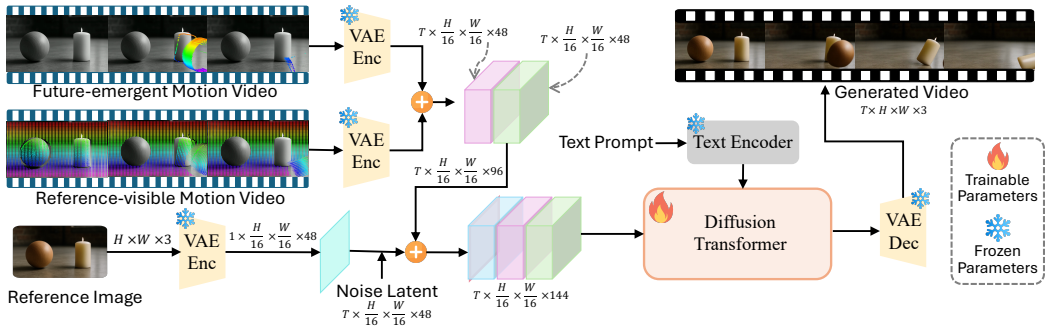

Figure 3: Architecture of the physics-guided video generation model. The visible and future visible motion fields are encoded by a frozen 3D VAE and concatenated with the noisy video latent along the channel dimension. A zero-initialized injection strategy preserves the pretrained model behavior while progressively incorporating motion guidance.

message passing to model particle interactions:

$$\mathbf{m}_{ij}^{(t)} = \psi^{e}([\mathbf{h}_{ij}^{e}; \mathbf{e}_{i}^{(t)}; \mathbf{e}_{j}^{(t)}]), \quad \mathbf{e}_{i}^{(t+1)} = \mathbf{e}_{i}^{(t)} + \psi^{v}\left([\mathbf{h}_{i}^{v}; \sum_{j \in \mathcal{N}(i)} \mathbf{m}_{ij}^{(t)}]\right), \tag{1}$$

where $\mathcal{N}(i)$ denotes the neighborhood of particle i, and $\mathbf{e}_{i}^{(0)} = \mathbf{0}$. Finally, particle velocities are predicted as $\hat{\mathbf{v}}_i = \phi^{\text{dec}}(\mathbf{e}_{i}^{(T)})$, which are used to update positions and roll out trajectories. This formulation enables PGDS to capture structured interactions and learn fine-grained physical dynamics in a particle-based representation.

Training Objective. We train the simulator by minimizing the mean squared error between predicted and ground-truth particle velocities. Given $\hat{\mathbf{v}}_i$ and $\mathbf{v}_i^*$, the loss is defined as

$$\mathcal{L}_{\text{PGDS}} = \frac{1}{B} \sum_{g=1}^{B} \frac{1}{N_g} \sum_{i \in \mathcal{G}} \frac{1}{3} \left\| \hat{\mathbf{v}}_i - \mathbf{v}_i^* \right\|_2^2, \tag{2}$$

where N_g is the number of particles in graph $\mathcal{G}$ and B is the batch size. This formulation ensures equal contribution from each scene regardless of its size.

Simulation Capability and Generalization PGDS models local particle-level interactions rather than object-specific global geometry, which enables it to generalize across unseen object shapes and spatial configurations. Its material space is also extensible: new material types can be incorporated by adding corresponding simulated data and specifying them through the material attribute vector $\mathbf{c}_i$. Prior studies [48] further show that such a simulator can remain reliable even when the number of particles increases substantially at test time, suggesting strong scalability to more complex scenes. These properties make PGDS well-suited for open-ended video generation, where object geometry, particle density, and interaction patterns may differ from those observed during training.

3.3 Physics-grounded Image-to-Video Generation

Visible–Invisible Motion Field. Existing motion-guided video generation methods typically convert predicted motion into 2D optical flow or sparse point tracks. Although effective for conveying visible motion, such representations discard 3D spatial structure and only describe regions observable from the reference view. They therefore provide limited guidance for occluded surfaces or objects that become visible in later frames. To address this limitation, we decompose physical motion into two complementary tracking signals. The first is a *reference-visible motion tracking video* $\mathcal{M}_{\text{vis}}$, which records the motion of points visible in the input frame. The second is a *future-emergent motion tracking video* $\mathcal{M}_{\text{fv}}$, which captures points that are initially occluded or outside the reference view but become visible in future frames. Together, these two signals form a visible–invisible motion representation that preserves both immediately observable motion and physically predicted future motion. More details are provided in Appendix E.4. During training, we obtain these tracking signals from videos using a learning-based tracking pipeline. During inference, we derive them by projecting the simulated 3D trajectories from PGDS into the same tracking-video format.

Unified Motion Encoding and Injection We build our physics-guided generator on top of Wan2.2-TI2V-5B [49]. To inject our dual motion guidance, we encode $\mathcal{M}_{\text{vis}}$ and $\mathcal{M}_{\text{fv}}$ using the same frozen 3D VAE to yield the latent representations $\mathbf{z}_{\text{vis}}$ and $\mathbf{z}_{\text{fv}}$, and then concatenate them with the noisy video latent along the channel dimension:

$$\mathbf{z}_t = \text{Concat}\left[\mathbf{z}_t,\, \mathbf{z}_{\text{vis}},\, \mathbf{z}_{\text{fv}}\right]. \tag{3}$$

To preserve the pretrained model behavior, we adopt a zero-initialized injection strategy to allow the model to incorporate motion guidance progressively without disrupting the original generation capability. For appearance conditioning, we follow the standard TI2V scheme by replacing the first-frame latent with a clean reference, ensuring consistent visual grounding.

Training Objective for Physics-Guided Generation. Following [30, 14, 49], we adopt the flow matching framework. Given a clean video latent $\mathbf{z}_0$ and Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the noisy latent at timestep t is:

$$\mathbf{z}_t = (1 - \sigma_t)\, \mathbf{z}_0 + \sigma_t\, \boldsymbol{\epsilon}, \tag{4}$$

where σ_t follows the noise schedule and t is uniformly sampled. The model predicts the velocity field $\boldsymbol{\epsilon} - \mathbf{z}_0$ conditioned on text, the reference image, and the dual tracking latents:

$$\mathcal{L}_{\text{gen}} = \mathbb{E}_{\mathbf{z}_0, \boldsymbol{\epsilon}, t}\left[w(\sigma_t)\, \left\| f_\theta(\mathbf{z}_t^{\text{in}}, t, \mathbf{c}_{\text{text}}) - (\boldsymbol{\epsilon} - \mathbf{z}_0) \right\|_2^2\right], \tag{5}$$

where $w(\sigma_t)$ denotes the signal-to-noise weighting, $\mathbf{c}_{\text{text}}$ is the text embedding, and $\mathbf{z}_t^{\text{in}}$ contains the noisy video latent together with the visible and future-visible tracking latents. During training, the first-frame latent is kept clean and assigned zero timestep, indicating a noise-free visual reference.

4 Settings

Dataset. For physical dynamics modeling, we train PGDS on the Physion++ dataset [48], which contains approximately 16K samples generated by the ThreeDWorld simulator [16] and covers eight types of physical interactions. For motion-guided video generation, we combine simulated data with real-world videos from MiraData [27]. We further extract motion supervision using SpatialTracker2 [57], thereby generating two trajectory sets: reference-visible and future-emergent trajectories. For evaluation, we focus on seven physics-aware interaction categories, including four primitive motion types (*free fall*, *domino*, *collision*, and *translation*) and three compositional scenarios (*collision+domino*, *free fall+domino*, and *free fall+collision*). Each category contains 50 samples, each consisting of a reference image and a paired text prompt. In addition to single-object scenes, we explicitly include multi-object compositions (e.g., multiple objects or mixed categories), enabling evaluation under both simple and complex interaction settings.

Implementation Details. PGDS operates on particle graphs with about 3000 nodes per scene, where edges connect particles within a radius r=0.08. Node and edge representations have latent dimensions of 200 and 300, respectively. We perform 10 rounds of message passing to model particle interactions. The model is trained for 1K epochs on four NVIDIA H200 GPUs with a global batch size of 32. We use the Adam optimizer with an initial learning rate of 10^{-4} and a ReduceLROnPlateau scheduler. To improve robustness, Gaussian noise with $\sigma = 3 \times 10^{-4}$ is injected into input velocities during training. During the video generation training stage, we initialize the generation backbone from Wan2.2-TI2V-5B [49]. To facilitate stable motion injection, we adopt an asymmetric learning rate scheme: the patch embedding layer is trained with a learning rate of 2×10^{-4}, while all other parameters use 2×10^{-5}. This design allows the newly introduced motion channels to adapt rapidly while preserving the pretrained generation capability. We optimize the model using AdamW [34] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 3×10^{-2}, and $\epsilon = 10^{-10}$. Training uses a constant learning rate schedule with 100 warmup steps and gradient clipping at 0.5.

Baselines and Evaluation metrics. We compare against two groups of methods. (1) General video generation models (without trajectory injection): open-source models including Wan2.2-TI2V-5B [49], Wan2.2-I2V-A14B [49], HunyuanVideo-1.5 720p I2V [56], and CogVideoX1.5-5B-I2V [61], as well as closed-source state-of-the-art models Wan2.7 and Veo3.1 [13]. (2) Physics-guided generation models: PhysCtrl [50], NewtonGen [63], and PhysT2V [59]. Following [3], we adopt two evaluation metrics. **(1) Semantic Adherence (SA):** measures the alignment between the generated video and the text prompt in content and motion (e.g., force direction and location); **(2) Physical Commonsense (PC):** evaluates whether the motion follows physically plausible dynamics consistent with the specified forces.

Table 1: Quantitative results on seven physical video generation scenarios. Best results are **bold**. "–" indicates unsupported cases.

Method	Domino		Collision		Free Fall		Coll.+Dom.		F.F.+Dom.		F.F.+Coll.		Translation	
	SA	PC	SA	PC	SA	PC	SA	PC	SA	PC	SA	PC	SA	PC
Open-source Video Generation Models														
Wan2.2-I2V-14B	2.483	3.017	2.750	3.083	2.317	3.033	2.217	3.050	2.533	2.950	2.517	3.000	3.000	4.000
Wan2.2-TI2V-5B	2.650	3.033	2.917	3.067	2.183	3.083	2.200	3.033	2.467	3.033	2.183	3.083	3.000	4.100
HunyuanVideo-1.5	2.550	3.017	2.867	3.067	2.317	3.017	2.233	3.000	2.617	2.950	2.467	3.000	3.050	3.933
CogVideoX1.5-5B	2.450	3.200	2.833	3.017	2.333	3.000	2.300	2.983	2.650	3.000	2.333	3.000	3.050	3.933
Physics-aware Models														
PhysT2V	2.267	3.867	2.533	3.400	2.050	4.283	2.283	3.267	2.383	3.333	2.117	3.933	3.033	4.400
NewtonGen	–	–	–	–	–	–	–	–	–	–	–	–	3.033	3.983
PhysCtrl	–	–	–	–	–	–	–	–	–	–	–	–	2.867	4.567
Closed-source Video Generation Models														
Wan-2.7	2.200	2.800	2.800	3.000	2.400	3.000	2.200	3.000	2.400	2.800	2.200	3.000	3.000	**4.600**
Veo 3.1	2.800	3.000	2.800	3.000	2.400	3.000	2.400	3.000	2.800	3.000	2.200	3.000	3.200	4.000
Ours	**3.976**	**4.013**	**4.160**	**4.394**	**4.347**	**4.328**	**3.854**	**4.016**	**3.985**	**4.017**	**4.327**	**4.589**	**4.013**	4.578

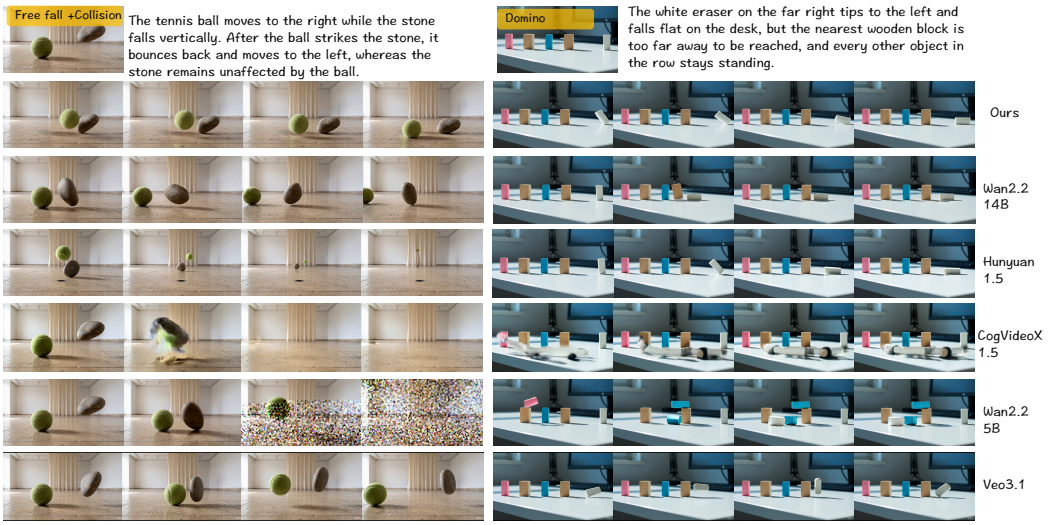

Figure 4: Qualitative comparison with existing methods. Our method produces physically plausible dynamics with accurate object interactions (e.g., correct collision responses and domino chain propagation), while baseline models frequently exhibit artifacts such as object interpenetration, implausible motion trajectories, or failure to trigger expected physical events.

5 Results and Discussion

Main Results on Physically Consistent Video Generation. We compare our method with open-source baselines, recent SOTA models, and prior physics-guided approaches on the proposed benchmarks. As shown in Table 1, our model achieves the best overall performance across most scenarios. Compared to its backbone Wan2.2-TI2V-5B, our model consistently improves both SA and PC scores, demonstrating the effectiveness of integrating learned physical dynamics and structured motion guidance. We observe that most models perform well on the *Translation* benchmark, which involves single-object motion without interactions. However, performance drops significantly as the task complexity increases, especially in compositional scenarios (e.g., *collision+domino*, *free fall+collision*). In contrast, our method maintains stable performance, indicating stronger capability in modeling multi-object interactions and causal dynamics. All methods achieve relatively lower scores on domino-related tasks that involve dense object layouts and frequent occlusions, as illustrated in Figure 4. They often fail to propagate contact dynamics correctly (e.g., incomplete domino chains) or produce unstable and inconsistent object motions. In contrast, our method generates coherent interaction patterns and accurately models contact propagation, demonstrating stronger capability in capturing structured physical dynamics under complex multi-object settings. Additional visualizations can be found in Appendix F.

We further evaluate PhysT2V, NewtonGen, and PhysCtrl. NewtonGen and PhysCtrl are limited to simple motion settings and are only applicable to the Translation benchmark. PhysT2V achieves

Table 2: Results (best in **bold**) on the compatible VideoPhy2 subset.

Model	SA ↑	PC ↑
CogVideoX-5B	3.21	3.65
Wan2.2-5B	3.14	3.46
Ours (visible-only)	3.47	3.78
Ours	**4.01**	**4.13**

Table 3: PGDS rollouts on 7 Physion++ scenarios. Position MSE: mean squared particle position error; CoM: center-of-mass error (% object size); rotation: angular deviation.

Horizon k	Position MSE (cm^2) ↓	CoM error (% size) ↓	Rotation error (°) ↓
30	0.7	0.5	1.3
60	1.7	1.2	3.8
90	3.0	2.6	7.5

Table 4: PGDS motion prediction on seven Physion++ test scenarios. Vel. RMSE is the root mean squared particle velocity error over complete rollouts. Acc-V reports the percentage of particles meeting strict (0.05 m/s or 5%) or relaxed (0.10 m/s or 10%) velocity-error thresholds. Kinetic-energy bias is $(E_{\mathrm{pred}} - E_{\mathrm{GT}})/E_{\mathrm{GT}} \times 100\%$.

Scenario	Vel. RMSE (m/s) ↓	Acc-V Strict (%) ↑	Acc-V Relaxed (%) ↑	Energy Bias (%)
Dominoes	0.0077	99.20	99.65	–4.7
Collide	0.0359	95.04	98.79	–5.5
Support	0.0209	96.16	98.36	–6.5
Link	0.0284	94.51	98.14	+2.6
Contain	0.0171	96.59	98.94	–3.8
Roll	0.0384	95.48	98.27	–5.5
Drop	0.0264	94.48	96.47	–1.2
Average	**0.0250**	**95.92**	**98.37**	**–3.5**

competitive PC scores, indicating its strength in improving physical plausibility via LLM-based refinement. However, its SA scores are generally lower, suggesting a trade-off between physical consistency and semantic fidelity. In contrast, our trajectory-based framework provides more stable and accurate guidance.

For external evaluation, we use VideoPhy2 [4] samples compatible with our setting. Table 2 reports SA/PC scores of 4.01/4.13 for our full model, the highest among all compared models. The full model also improves both scores over the visible-only variant, supporting the effectiveness of future-emergent guidance. Evaluation details and prompts are provided in Appendix E.1.

Prediction Accuracy of PGDS. As shown in Table 4, PGDS achieves high motion prediction accuracy across all seven physical scenarios in Physion++. Velocity errors remain low across scenarios, with an average strict Acc-V of 95.92%, indicating that the model accurately captures motion changes under diverse physical interactions. The small discrepancy between predicted and ground-truth kinetic energy further indicates consistency with the ground-truth dynamics.

Although position, center-of-mass, and rotation errors accumulate with the prediction horizon (Table 3), they remain small after 90 steps, indicating accurate translation and rotation predictions over multi-step rollouts. These results show that PGDS produces accurate 3D motion trajectories across diverse physical scenarios, providing reliable physical guidance for downstream video generation. We further evaluate downstream video generation under incomplete motion guidance and upstream perception errors, with detailed robustness experiments provided in Appendix G.

Physical Dynamics Simulation Efficacy. We compare the efficiency of different physical prior sources in Table 5. Simulator-based methods (e.g., PhysGen, PhysMotion) require extensive manual design (5–10 parameters) and precise 3D meshes, resulting in high setup cost and runtime (∼10 minutes), while showing limited generalization. Data-driven approaches reduce manual effort (∼2–5 parameters) and runtime (∼6–8 minutes), but their generalization remains limited or moderate. LVLM-based methods eliminate manual configuration and achieve the fastest runtime (∼2 minutes), yet lack fine-grained physical modeling. In contrast, PGDS requires no manual parameter tuning, maintains a competitive runtime (∼5 minutes), achieves strong generalization to unseen shapes, and provides a better balance between efficiency and physical modeling capability. We also report the end-to-end runtime and peak GPU memory usage of our full model, including stage-wise measurements and conditioning overhead, in Appendix E.2.

Table 5: Configuration efficiency of trajectory generation methods. We compare setup cost and generalization with time and parameter estimates from public implementations.

Method	Type	Manual Phys. Params ↓	Precise 3D Mesh	Total Runtime (incl. manual tuning) ↓	Unseen Shapes
PhysGen [31]	Simulator	∼6–10	Required	∼10 min	Limited
PhysMotion [46]	Simulator	∼6–8	Required	∼10 min	Limited
PhysGen3D [8]	Simulator	∼5–7	Required	∼10 min	Limited
PhysCtrl [50]	Data-driven	∼3–5	Partial	∼8 min	Moderate
NewtonGen [63]	Data-driven	∼2–3	Not required	∼6 min	Limited
PhysT2V [59]	LVLM-based	–	Not required	∼2 min	Moderate
PGDS (Ours)	Learned Sim.	**0**	Required	**∼5 min**	**Strong**

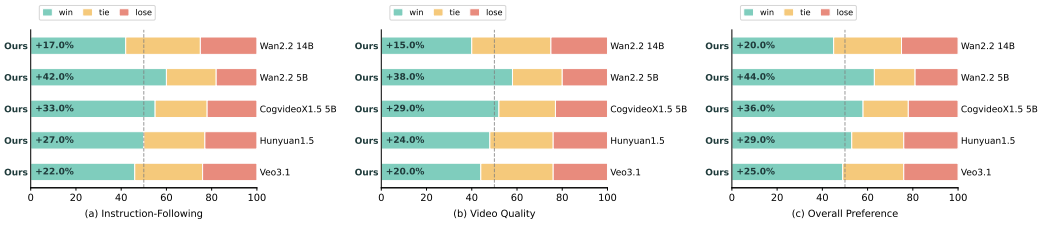

Figure 5: Pairwise human evaluation comparing our method against baseline models. For each comparison, human annotators judge which method produces better results across three criteria: (a) instruction-following, measuring how faithfully the generated video adheres to the given text prompt; (b) video quality, assessing visual fidelity, temporal coherence, and artifact severity; and (c) overall preference. Results are reported as win/tie/lose rates (%).

Table 6: Comparison of physical prior injection strategies and the visible-only ablation across four scenarios. Our visible-only and full variants use the same diffusion backbone; the full variant uses both reference-visible and future-emergent tracking videos. Best results are **bold**.

Method	Domino		Collision		Free Fall		Translation	
	SA	PC	SA	PC	SA	PC	SA	PC
Go with Flow [5]	2.583	3.389	3.275	3.691	3.670	3.928	3.568	4.419
Wan-Move [12]	2.672	3.351	3.594	3.771	3.892	4.017	**4.167**	**4.602**
Diffusion as Shader [21]	2.640	3.295	3.697	3.829	3.764	3.923	3.821	4.493
Ours (visible-only)	2.693	3.348	3.693	3.717	3.972	4.004	3.837	4.521
Ours (full)	**3.976**	**4.013**	**4.160**	**4.394**	**4.347**	**4.328**	4.013	4.578

Human Evaluation We conduct pairwise human evaluation, with results shown in Figure 5. Our method consistently outperforms all baselines across all three criteria. In instruction following, our model achieves clear gains (e.g., +42.0% over Wan2.2-5B), indicating better alignment with the input prompt. For video quality, we observe similar improvements (e.g., +38.0%), suggesting stronger visual fidelity and temporal consistency. These advantages lead to higher overall preference (up to +44.0%), demonstrating that our method produces more desirable results. Overall, the results confirm that our approach improves both physical plausibility and perceptual quality.

Comparison of Physical Prior Injections. We compare physical prior injection strategies by converting the same PGDS-predicted trajectories into different motion representations and feeding them to existing models. Specifically, we provide optical flow to Go with Flow, 10 sparse tracking points to Wan-Move, and visible tracking videos to Diffusion as Shader, while retaining the original pre-trained weights of each method. As shown in Table 6, our full model achieves the highest SA and PC scores on Domino, Collision, and Free Fall. The optical flow used by Go with Flow struggles to fully represent object motion under occlusions and complex interactions, leading to motion errors and generation artifacts during interactions, as illustrated in Figure 6. Diffusion as Shader relies solely on visible trajectories and may lack motion information for newly revealed regions when objects rotate or become occluded, limiting the effectiveness of its guidance. Wan-Move, which uses the Wan2.2-I2V-A14B backbone, performs well on Translation but provides less accurate motion control than our full model in complex interaction scenarios.

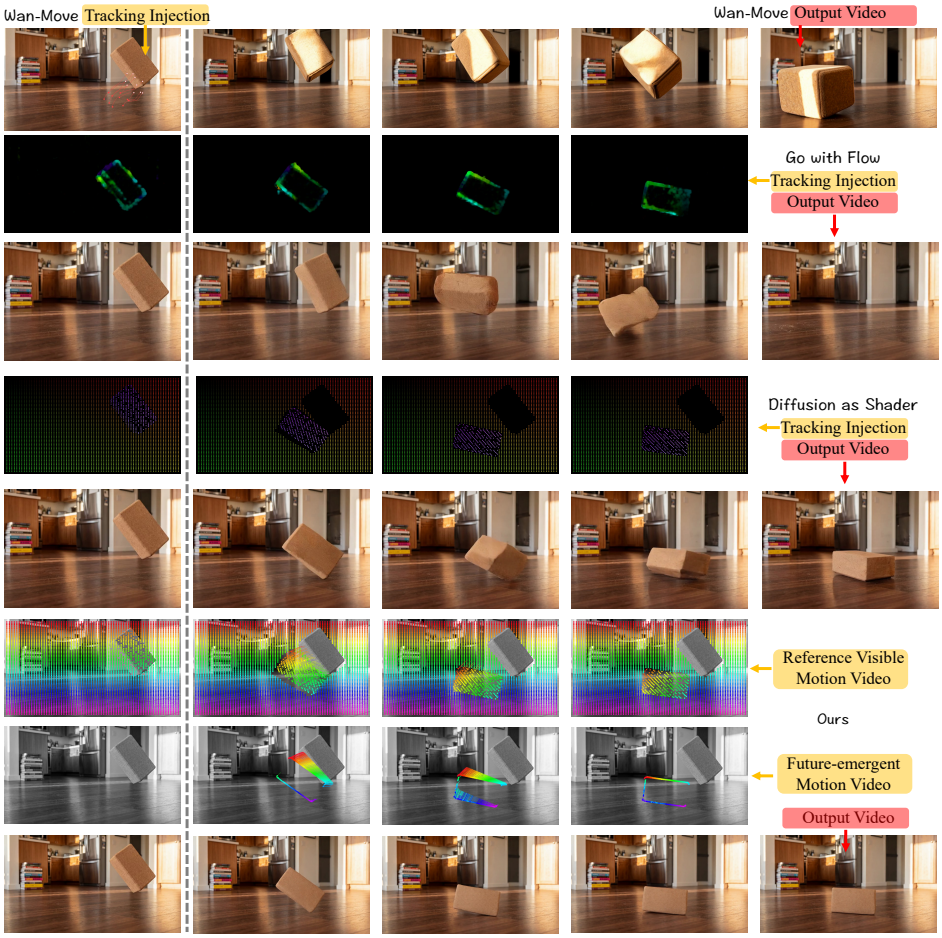

Figure 6: Comparison with different physical prior injection methods.

To further isolate the contribution of future visible motion, we conduct an ablation using the same diffusion backbone, comparing visible-only tracking videos with the joint use of reference-visible and future visible tracking videos. The visible-only variant yields lower SA and PC scores than the full model in all four scenarios, with particularly pronounced gains from dual-stream guidance on Domino and Collision. These results indicate that motion visible from the reference view alone is insufficient to fully describe occlusions and object interactions, whereas future visible trajectories supply motion information for regions that become visible later. Combining the two types of trajectories provides more complete physical guidance for video generation, improving both semantic adherence and physical plausibility.

6 Conclusion

In this work, we propose **Seeing the Unseen**, a unified framework that combines a particle-based graph dynamics simulator with visibility-aware trajectory guidance to enhance physically grounded video generation. By explicitly modeling both observable motion and future-emergent dynamics, our method provides more complete and structured physical priors for video synthesis. Extensive experiments demonstrate that our approach consistently outperforms existing methods across diverse challenging physical interaction scenarios, particularly in complex multi-object settings. Both quantitative results and human evaluations validate that our method improves physical plausibility, interaction consistency, and perceptual quality. Overall, our results highlight the importance of integrating structured physical modeling and visibility-aware motion representations. We hope this work provides a step toward bridging video generation and physical reasoning and inspires future research on physics-grounded generative models.

References

[1] Genesis Authors. Genesis: A generative and universal physics engine for robotics and beyond, December 2024. URL `https://github.com/Genesis-Embodied-AI/Genesis`.

[2] Xiangyu Bai, He Liang, Bishoy Galoaa, Utsav Nandi, Shayda Moezzi, Yuhang He, and Sarah Ostadabbas. Moregen: Multi-agent motion-reasoning engine for code-based text-to-video synthesis. *arXiv preprint arXiv:2512.04221*, 2025.

[3] 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. *arXiv preprint arXiv:2406.03520*, 2024.

[4] Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. Videophy-2: A challenging action-centric physical commonsense evaluation in video generation. *arXiv preprint arXiv:2503.06800*, 2025.

[5] Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, et al. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13–23, 2025.

[6] 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. *arXiv preprint arXiv:2512.24551*, 2025.

[7] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

[8] Boyuan Chen, Hanxiao Jiang, Shaowei Liu, Saurabh Gupta, Yunzhu Li, Hao Zhao, and Shenlong Wang. Physgen3d: Crafting a miniature interactive world from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6178–6189, 2025.

[9] Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, et al. Sam 3d: 3dfy anything in images. *arXiv preprint arXiv:2511.16624*, 2025.

[10] Yongfan Chen, Xiuwen Zhu, and Tianyu Li. A physical coherence benchmark for evaluating video generation models via optical flow-guided frame prediction. *arXiv preprint arXiv:2502.05503*, 2025.

[11] Mengyu Chu, Lingjie Liu, Quan Zheng, Aleksandra Franz, Hans-Peter Seidel, Christian Theobalt, and Rhaleb Zayer. Physics informed neural fields for smoke reconstruction with sparse data. *ACM Transactions on Graphics (ToG)*, 41(4):1–14, 2022.

[12] Ruihang Chu, Yefei He, Zhekai Chen, Shiwei Zhang, Xiaogang Xu, Bin Xia, Dingdong Wang, Hongwei Yi, Xihui Liu, Hengshuang Zhao, Yu Liu, Yingya Zhang, and Yujiu Yang. Wan-move: Motion-controllable video generation via latent trajectory guidance. *CoRR*, abs/2512.08765, 2025.

[13] Google DeepMind. Veo 3.1: Professional-grade 4k video generation with native audio, 2026. URL `https://ai.google.dev/gemini-api/docs/models/veo`.

[14] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[15] Yutao Feng, Yintong Shang, Xiang Feng, Lei Lan, Shandian Zhe, Tianjia Shao, Hongzhi Wu, Kun Zhou, Chenfanfu Jiang, and Yin Yang. Elastogen: 4d generative elastodynamics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 3966–3975, 2026.

[16] Chuang Gan, Jeremy Schwartz, Seth Alter, Damian Mrowca, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

[17] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1–12, 2025.

[18] Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force prompting: Video generation models can learn and generalize physics-based control signals. *arXiv preprint arXiv:2505.19386*, 2025.

[19] Nate Gillman, Yinghua Zhou, Zitian Tang, Evan Luo, Arjan Chakravarthy, Daksh Aggarwal, Michael Freeman, Charles Herrmann, and Chen Sun. Goal force: Teaching video models to accomplish physics-conditioned goals. *arXiv preprint arXiv:2601.05848*, 2026.

[20] Bohai Gu, Taiyi Wu, Dazhao Du, Jian Liu, Shuai Yang, Xiaotong Zhao, Alan Zhao, and Song Guo. Place-it-r1: Unlocking environment-aware reasoning potential of mllm for video object insertion. *arXiv preprint arXiv:2603.06140*, 2026.

[21] Zekai Gu, Rui Yan, Jiahao Lu, Peng Li, Zhiyang Dou, Chenyang Si, Zhen Dong, Qifeng Liu, Cheng Lin, Ziwei Liu, et al. Diffusion as shader: 3d-aware video diffusion for versatile video generation control. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–12, 2025.

[22] Xuyang Guo, Jiayan Huo, Zhenmei Shi, Zhao Song, Jiahao Zhang, and Jiale Zhao. T2vphysbench: A first-principles benchmark for physical consistency in text-to-video generation. *arXiv preprint arXiv:2505.00337*, 2025.

[23] Yidong Huang, Zun Wang, Han Lin, Dong-Ki Kim, Shayegan Omidshafiei, Jaehong Yoon, Yue Zhang, and Mohit Bansal. Planning with sketch-guided verification for physics-aware video generation. *arXiv preprint arXiv:2511.17450*, 2025.

[24] Ziqi Huang, Ning Yu, Gordon Chen, Haonan Qiu, Paul Debevec, and Ziwei Liu. Vchain: Chain-of-visual-thought for reasoning in video generation. *arXiv preprint arXiv:2510.05094*, 2025.

[25] Sihui Ji, Xi Chen, Xin Tao, Pengfei Wan, and Hengshuang Zhao. Physmaster: Mastering physical representation for video generation via reinforcement learning. *arXiv preprint arXiv:2510.13809*, 2025.

[26] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *Acm siggraph 2016 courses*, pages 1–52. ACM, 2016.

[27] Xuan Ju, Yiming Gao, Zhaoyang Zhang, Ziyang Yuan, Xintao Wang, Ailing Zeng, Yu Xiong, Qiang Xu, and Ying Shan. Miradata: A large-scale video dataset with long durations and structured captions. *Advances in Neural Information Processing Systems*, 37:48955–48970, 2024.

[28] Pika Labs. Pika 2.0: Human-centric video animation and control, 2026. URL `https://pika.art/`.

[29] Minzhang Li, Kuixiang Shao, Xuebing Li, Yuyang Jiao, Yinuo Bai, Hengan Zhou, Sixian Shen, Jiayuan Gu, and Jingyi Yu. Spread: Spatial-physical reasoning via geometry aware diffusion. *arXiv preprint arXiv:2603.27573*, 2026.

[30] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[31] Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pages 360–378. Springer, 2024.

[32] Tiantian Liu, Adam W Bargteil, James F O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM Transactions on Graphics (TOG)*, 32(6):1–7, 2013.

[33] Wei Liu, Ziyu Chen, Zizhang Li, Yue Wang, Hong-Xing Yu, and Jiajun Wu. Realwonder: Real-time physical action-conditioned video generation. *arXiv preprint arXiv:2603.05449*, 2026.

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[35] Miles Macklin and Matthias Müller. Position based fluids. *ACM Transactions on Graphics (TOG)*, 32(4):1–12, 2013.

[36] 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.

[37] Pedro Miraldo and José Pedro Iglesias. A unified model for line projections in catadioptric cameras with rotationally symmetric mirrors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15797–15806, 2022.

[38] Sriram Narayanan, Ziyu Jiang, Srinivasa Narasimhan, and Manmohan Chandraker. Phyco: Learning controllable physical priors for generative motion, 2026.

[39] OpenAI. Hello gpt-4o: Our new flagship model that can reason across audio, vision, and text in real time, 2024. URL `https://openai.com/index/hello-gpt-4o/`.

[40] Saurabh Pathak, Elahe Arani, Mykola Pechenizkiy, and Bahram Zonooz. Physvid: Physics aware local conditioning for generative video models. *arXiv preprint arXiv:2603.26285*, 2026.

[41] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[42] Andreas Peer, Christoph Gissler, Stefan Band, and Matthias Teschner. An implicit sph formulation for incompressible linearly elastic solids. In *Computer Graphics Forum*, volume 37, pages 135–148, 2018.

[43] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[44] Runway Research. Gen-3 alpha: A new frontier for high-fidelity video synthesis, 2024. URL `https://runwayml.com/research/introducing-gen-3-alpha`.

[45] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468, 2020.

[46] Xiyang Tan, Ying Jiang, Xuan Li, Zeshun Zong, Tianyi Xie, Yin Yang, and Chenfanfu Jiang. Physmotion: Physics-grounded dynamics from a single image. *arXiv preprint arXiv:2411.17189*, 2024.

[47] Kuaishou Technology. Kling ai: Commercial licensing and professional video generation, 2026. URL `https://klingai.com/`.

[48] Hsiao-Yu Tung, Mingyu Ding, Zhenfang Chen, Daniel Bear, Chuang Gan, Josh Tenenbaum, Dan Yamins, Judith Fan, and Kevin Smith. Physion++: Evaluating physical scene understanding that requires online inference of different physical properties. *Advances in Neural Information Processing Systems*, 36:67048–67068, 2023.

[49] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[50] Chen Wang, Chuhao Chen, Yiming Huang, Zhiyang Dou, Yuan Liu, Jiatao Gu, and Lingjie Liu. Physctrl: Generative physics for controllable and physics-grounded video generation. *arXiv preprint arXiv:2509.20358*, 2025.

[51] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025.

[52] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[53] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025.

[54] Xiang Wang, Hangjie Yuan, Shiwei Zhang, Dayou Chen, Jiuniu Wang, Yingya Zhang, Yujun Shen, Deli Zhao, and Jingren Zhou. Videocomposer: Compositional video synthesis with motion controllability. *Advances in Neural Information Processing Systems*, 36:7594–7611, 2023.

[55] Yiming Wang, Siyu Tang, and Mengyu Chu. Physics-informed learning of characteristic trajectories for smoke reconstruction. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.

[56] Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025.

[57] Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: Advancing 3d point tracking with explicit camera motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6726–6737, 2025.

[58] Tianyi Xie, Yiwei Zhao, Ying Jiang, and Chenfanfu Jiang. Physanimator: Physics-guided generative cartoon animation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10793–10804, 2025.

[59] Qiyao Xue, Xiangyu Yin, Boyuan Yang, and Wei Gao. Phyt2v: Llm-guided iterative self-refinement for physics-grounded text-to-video generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18826–18836, 2025.

[60] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12360–12370, 2025.

[61] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[62] Shu Yu and Chaochao Lu. Lina: Learning interventions adaptively for physical alignment and generalization in diffusion models. *arXiv preprint arXiv:2512.13290*, 2025.

[63] Yu Yuan, Xijun Wang, Tharindu Wickremasinghe, Zeeshan Nadir, Bole Ma, and Stanley H Chan. Newtongen: Physics-consistent and controllable text-to-video generation via neural newtonian dynamics. *arXiv preprint arXiv:2509.21309*, 2025.

[64] Ke Zhang, Cihan Xiao, Jiacong Xu, Yiqun Mei, and Vishal M Patel. Think before you diffuse: Infusing physical rules into video diffusion. *arXiv preprint arXiv:2505.21653*, 2025.

[65] Xiangdong Zhang, Jiaqi Liao, Shaofeng Zhang, Fanqing Meng, Xiangpeng Wan, Junchi Yan, and Yu Cheng. Videorepa: Learning physics for video generation through relational alignment with foundation models. *arXiv preprint arXiv:2505.23656*, 2025.

[66] Daoxin Zhong, Jun Li, and Meng Yee Michael Chuah. Plane detection and ranking via model information optimisation. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2416–2423, 2025.

[67] O. C. Zienkiewicz. *The Finite Element Method*. McGraw-Hill, 3rd edition, 1977.

A Ethic Statement

This work focuses on improving the physical consistency of video generation models. We do not introduce new datasets involving human subjects or sensitive personal information, and all training data are derived from publicly available or synthetic sources. Potential risks may arise from the misuse of video generation technology, such as generating misleading or fabricated content. While our method improves physical plausibility, it does not aim to create deceptive content, and we encourage responsible use aligned with existing guidelines for generative models. Overall, we believe our work contributes positively to advancing reliable and physically grounded video generation, which can benefit applications such as simulation, education, and scientific visualization.

B LLM Usage

Large language models (LLMs) were used solely for language polishing and improving the clarity of writing. All technical content, including model design, experiments, and analysis, was developed by the authors. The use of LLMs does not affect the originality or validity of the research contributions.

C Limitations

Despite its effectiveness, our method is currently evaluated on a limited set of physical scenarios, primarily focusing on structured interactions such as collisions, domino effects, and free-fall motion. While the proposed framework is general and can be extended to more complex physical processes (e.g., fluid dynamics or highly deformable objects), such extensions require additional data and simulation design. Moreover, our approach relies on an intermediate 3D scene lifting stage, which may introduce errors in geometry estimation that can propagate to downstream dynamics simulation and video generation. We leave the exploration of broader physical domains and more robust 3D perception as future work.

D Detailed Method

D.1 Particle-based Graph Dynamics Simulator

The Particle-based Graph Dynamics Simulator (PGDS) predicts particle velocities through iterative message passing over a graph constructed from the simulated scene. As illustrated in Figure 7, the model follows an Encode–Process–Decode architecture, consisting of a particle encoder, a relation encoder, a message-passing processor, and a particle predictor.

Encoding. Each particle i is associated with an input feature vector $\mathbf{z}_i = [\mathbf{c}_i; \mathbf{o}_i; \mathbf{s}_i]$, where $\mathbf{c}_i \in \mathbb{R}^C$ is a one-hot material attribute, $\mathbf{o}_i \in \mathbb{R}^6$ encodes the object-relative offset, and $\mathbf{s}_i \in \mathbb{R}^6$ denotes the (masked) velocity state. The particle encoder ϕ^v maps this feature to a latent representation:

$$\mathbf{h}_i^v = \phi^v(\mathbf{z}_i). \tag{6}$$

The offset $\mathbf{o}_i = \mathbf{x}_i - \bar{\mathbf{x}}_{g(i)}$ is defined relative to the centroid of object $g(i)$. For rigid bodies, this encodes structural constraints by preserving relative positions; for fluid particles, $\mathbf{o}_i$ is set to zero to allow free motion.

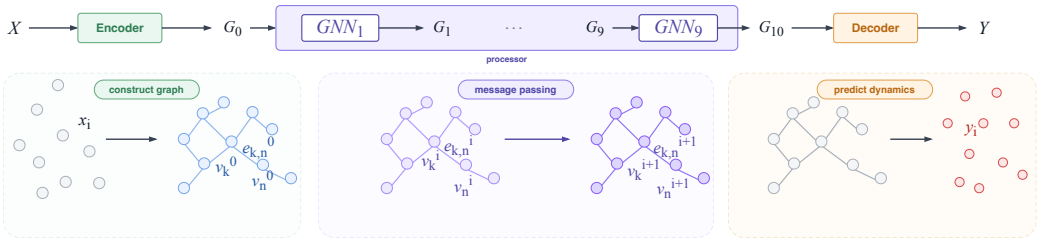

Figure 7: Architecture of the Particle-based Graph Dynamics Simulator (PGDS). The simulator follows an Encode–Process–Decode pipeline. Given particle states, a graph is first constructed based on spatial proximity. Node and edge features are encoded to capture material properties and geometric relations. A message-passing processor then iteratively propagates interaction effects across the graph. Finally, a decoder predicts particle velocities, which are used to update positions and roll out future dynamics.

For each edge e_{ij} connecting neighboring particles, we compute a relation embedding based on relative geometry:

$$\mathbf{h}_{ij}^{e} = \phi^{e}\left([\mathbf{h}_i^{v}; \mathbf{h}_j^{v}; \mathbf{r}_{ij}; \|\mathbf{r}_{ij}\|]\right), \tag{7}$$

where $\mathbf{r}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ denotes the relative displacement.

Processing. The processor performs T rounds of message passing (with $T = 10$ in our implementation) to propagate interaction effects across particles. Let $\mathbf{e}_i^{(t)}$ denote the particle effect at iteration t, initialized as $\mathbf{0}$. At each iteration, edge messages and particle updates are computed as:

$$\mathbf{m}_{ij}^{(t)} = \psi^{e}\left([\mathbf{h}_{ij}^{e}; \mathbf{e}_i^{(t)}; \mathbf{e}_j^{(t)}]\right), \tag{8}$$

$$\mathbf{e}_i^{(t+1)} = \mathbf{e}_i^{(t)} + \psi^{v}\left([\mathbf{h}_i^{v}; \sum_{j \in \mathcal{N}(i)} \mathbf{m}_{ij}^{(t)}]\right), \tag{9}$$

where $\mathcal{N}(i)$ denotes the neighbors of particle i, and the residual connection stabilizes iterative updates.

Decoding. After T iterations, the final particle effect $\mathbf{e}_i^{(T)}$ is decoded into a velocity prediction:

$$\hat{\mathbf{v}}_i = \phi^{\text{dec}}(\mathbf{e}_i^{(T)}), \tag{10}$$

where ϕ^{dec} is a multi-layer perceptron. The predicted velocities are then used to update particle positions and roll out future trajectories.

D.2 Unified Motion Encoding and Injection

We provide a detailed implementation of motion injection in Figure 8. Our model is built upon Wan2.2-TI2V-5B, a text-and-image-to-video diffusion transformer with a 3D VAE encoder and a DiT backbone.

Reference frame conditioning. We follow the standard TI2V noise-replacement scheme to incorporate appearance information. Specifically, the first frame is encoded into a clean latent $\mathbf{z}_0^{\text{ref}}$ and replaces the noisy latent at the initial temporal position:

$$\mathbf{z}_t^{(0)} \leftarrow \mathbf{z}_0^{\text{ref}}. \tag{11}$$

This mechanism injects appearance conditioning without introducing additional input channels and keeps motion guidance disentangled from visual content.

Latent construction. In the original model, the denoising network operates on a 48-channel latent produced by a 3D causal VAE with $16\times$ spatial and $4\times$ temporal compression. To incorporate motion guidance, we encode both the reference-visible tracking video $\mathcal{V}_{\text{vis}}$ and the future-emergent tracking video $\mathcal{V}_{\text{fv}}$ using the same frozen VAE, obtaining two additional 48-channel latent sequences:

$$\mathbf{z}_t = \text{Concat}\left[\mathbf{z}_t, \mathbf{z}_{\text{vis}}, \mathbf{z}_{\text{fv}}\right], \tag{12}$$

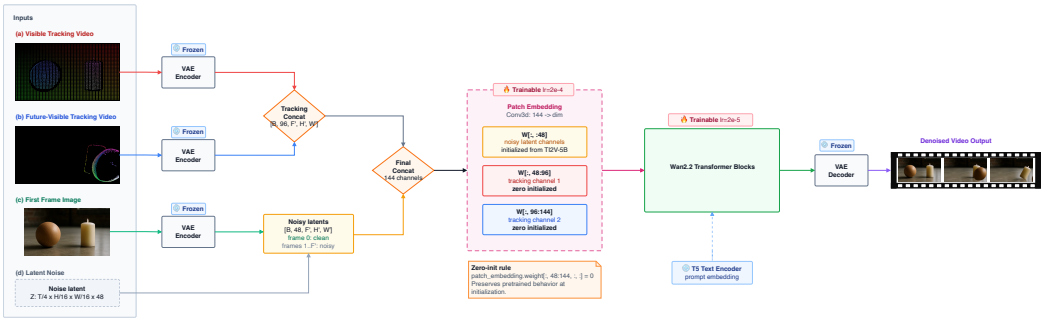

Figure 8: **Unified motion encoding and injection pipeline.** The visible tracking video $\mathcal{V}_{\text{vis}}$ and future-visible tracking video $\mathcal{V}_{\text{fv}}$ are encoded using a frozen 3D VAE to obtain two latent sequences. These are concatenated with the noisy video latent to form a multi-channel input to the diffusion transformer. To preserve pretrained behavior, the patch embedding weights corresponding to tracking channels are zero-initialized, while the original weights are copied from the pretrained model. The first frame is injected via noise replacement to provide appearance conditioning.

where $\mathbf{z}_t$ denotes the noisy video latent. This results in a 144-channel tensor that is tokenized by the original 3D patch embedding layer.

Zero-initialized motion injection. Expanding the input channels of the patch embedding introduces a distribution mismatch with the pretrained backbone. To mitigate this, we adopt a zero-initialization strategy: weights corresponding to the original 48 channels are copied from the pretrained checkpoint, while weights for the two tracking channels are initialized to zero. Formally, let $\mathbf{W} = [\mathbf{W}_{\text{orig}}, \mathbf{W}_{\text{track}}]$ denote the convolutional kernel, where

$$\mathbf{W}_{\text{track}} = \mathbf{0}. \tag{13}$$

This ensures that the injected motion signals do not affect the network at initialization, allowing the model to gradually learn to utilize motion guidance during fine-tuning.

E Extended Settings

E.1 Automatic Evaluation Protocol

We use the open-source VideoPhy2-AutoEval-7B model [4] to evaluate Semantic Adherence (SA) and Physical Commonsense (PC). Both metrics are scored from 1 to 5, with higher scores indicating better performance. We use the official VideoPhy2 evaluation prompts reproduced below, where `[V]` denotes the video being evaluated and `[T]` denotes its corresponding text description.

Semantic Adherence (SA).

> [V] Does this video match the description: '[T]'? Please rate the video on a scale from 1 to 5, where 5 indicates a perfect match and 1 indicates no relevance.

Physical Commonsense (PC).

> [V] Does this video adhere to physical laws? Rate the video on a scale from 1 to 5, where 5 means full compliance and 1 means significant violations.

E.2 End-to-End Runtime and Memory

All runtime and GPU memory measurements in this subsection are performed on a single NVIDIA H200 GPU. Table 7 reports the stage-wise runtime and peak GPU memory usage for scenes with 1, 3, and 5 foreground objects. The full pipeline consists of 3D lifting (S1), mesh-to-particle conversion and PGDS simulation (S2), dual-stream tracking rendering (S3), and video diffusion generation (S4). The end-to-end runtimes are 181.9, 189.0, and 197.6 seconds, respectively, with an overall peak GPU memory usage of 40.8 GiB. The stages preceding video diffusion (S1–S3) require approximately 30–50 additional seconds.

Table 7: End-to-end runtime and peak GPU memory usage on a single NVIDIA H200 GPU. The total covers stages S1–S4.

Stage	Component	1 Object	3 Objects	5 Objects	Peak VRAM
S1	3D lifting (segmentation + monocular depth + single-image 3D reconstruction)	10.1 s	15.8 s	24.7 s	28.7–40.8 GiB
S2	Mesh-to-particle conversion	4.0 s	4.9 s	5.1 s	< 1.2 GiB
S2	PGDS simulation	1.3 s	1.3 s	1.3 s	526 MiB
S3	VIMF dual-stream tracking rendering	15.9 s	16.0 s	15.9 s	< 1.5 GiB
S4	Video diffusion generation (Ours)	150.6 s	151.0 s	150.6 s	39.67 GiB
End-to-end total		**181.9 s**	**189.0 s**	**197.6 s**	**40.8 GiB**

To isolate the additional cost of dual-stream conditioning, we compare the video-generation stage with the base TI2V-5B backbone in Table 8. The input latent channels increase from 48 to 144, while the runtime of 35 denoising steps increases by only 0.20 seconds and peak GPU memory increases by 84 MiB. Encoding the two tracking videos requires 5.95 seconds. The total video-generation time increases from 144.20 to 150.28 seconds, an additional 6.08 seconds (4.3%). This 4.3% increase refers to the video-generation stage and excludes the additional costs of S1–S3. These measurements show that the full pipeline can run end-to-end on a single H200 GPU, with a peak memory usage of about 40 GiB.

Table 8: Additional cost of dual-stream conditioning in the video-generation stage on a single NVIDIA H200 GPU. The backbone and our model use 48 and 144 input latent channels, respectively.

Metric	Base TI2V-5B	Ours	Additional Overhead
Peak VRAM	39.59 GiB	39.67 GiB	$+84$ MiB
35 denoising steps	133.54 s	133.74 s	$+0.20$ s
Dual-stream tracking encoding	–	5.95 s	$+5.95$ s
Total generation time	144.20 s	150.28 s	$+6.08$ s ($+4.3\%$)

E.3 Input Image Lifting Construction Process.

We first apply the monocular depth estimation model MoGe-2 [53] to the input image to predict a pixel-level 3D point map, which provides a metric geometric prior for the scene. Based on this geometry, we use SAM3 [7] to segment foreground objects with text prompts and obtain an object-level binary mask for each instance. For each segmented region, SAM 3D Objects [9] performs single-image 3D reconstruction conditioned on the MoGe point map. The predicted pose directly encodes the real position and scale of the object with respect to the camera, removing the need for additional depth alignment or scale calibration. Finally, we use the estimated poses to transform all objects from their local coordinate systems into a unified scene coordinate system. The ground plane is estimated by applying RANSAC plane fitting to the MoGe point cloud from the lower half of the image. We then concatenate the 3D Gaussian parameters of all objects along the object dimension to form a complete multi-object scene representation.

E.4 Visible–Invisible Motion Field Construction.

Training. We describe the construction of the visible and future-visible motion signals used in our training data construction.

- **Visible tracking video generation.** We first apply VGGT4Track [51] to the input RGB video to estimate per-frame depth maps, camera intrinsics and extrinsics, and depth confidence. We then uniformly sample 4,900 query points on a regular grid over the first frame and track them across all subsequent frames using SpaTrack2 [57], which outputs the 2D projected position, visibility, and confidence of each point at every frame. For rendering, only visible points within the image boundary are drawn. Each point is colored by encoding its spatial position into RGB channels (R

= X, G = Y, B = 1/Z), where the color is determined at the first frame and remains fixed throughout the video.
- **Future-visible tracking video generation.** For each subsequent frame t, we uniformly sample 8,100 candidate points on a regular grid and back-project their 3D coordinates (obtained from the VGGT4Track depth and camera pose estimates) onto the first frame via geometric reprojection. A candidate point is identified as future-visible if its projection falls outside the first-frame boundary or is occluded by the first-frame depth, indicating that it corresponds to a newly revealed region not visible in the first frame. These future-visible points are then tracked using SpaTrack2 and rendered in the same manner as described above.

Inference. We describe the construction of the visible and future-visible motion signals yielded from PGDS during inference.

- **Visible Tracking Video.** We first identify visible mesh vertices in the first frame via backface culling and viewport clipping, then establish a uniform grid of sampling points on the image plane and associate each point with its nearest visible vertex using a KD-tree, establishing correspondences between 2D sampling points and 3D surface points. For each subsequent frame, we solve for the rigid-body transformation via SVD using the particle positions from the rollout, transform the first-frame vertices to the current frame, and project them onto the image plane to obtain per-frame 2D trajectories. Visibility is determined by jointly evaluating viewport bounds, backface culling, and depth-map-based inter-object occlusion; only points satisfying all visibility criteria are rendered.
- **Future Visible Tracking Video.** For mesh vertices that fail the visibility check in the first frame, we scan through subsequent simulation steps frame by frame and record the first frame at which each vertex becomes visible (the query frame). To control density, we perform grid-based downsampling at each vertex's 2D projected position at its query frame, retaining only the point with the earliest query frame per grid cell. The motion tracking and visibility computation pipeline is identical to that of the visible tracking video. Additionally, for each point, visibility is set to zero in all frames preceding its query frame, and its position is locked to the query-frame value, eliminating trajectory discontinuities before appearance.

Both types of videos are rendered in the SpaTracker format (rainbow colormap assigned by first-frame Y-coordinate on a grayscale background) and the DaS format (RGB channels encoding normalized coordinates and inverse depth on a black background), with depth-sorted drawing order to handle occlusion.

F In-depth Analyses

Analysis of Unseen/Future Dynamics. We present an ablation study on motion conditioning signals in Figure 9. The top row shows results from the full model using both reference-visible and future-emergent tracking signals, while the bottom row uses only reference-visible tracking. When relying solely on visible tracking, the model often produces artifacts such as ghosting or missing content. This is because visible trajectories only describe motion of pixels observed in the reference frame, and provide no information about newly exposed regions after object movement or occlusion. In contrast, incorporating future-emergent tracking signals provides complementary motion cues for regions that become visible over time. This allows the model to better reconstruct newly appearing content and maintain coherent motion dynamics. These results highlight the necessity of modeling future-visible trajectories, as they preserve more complete motion information and lead to more stable and physically consistent video generation.

More Visualizations. We show additional visualizations compared with baselines on different scenarios in Figures 10-14.

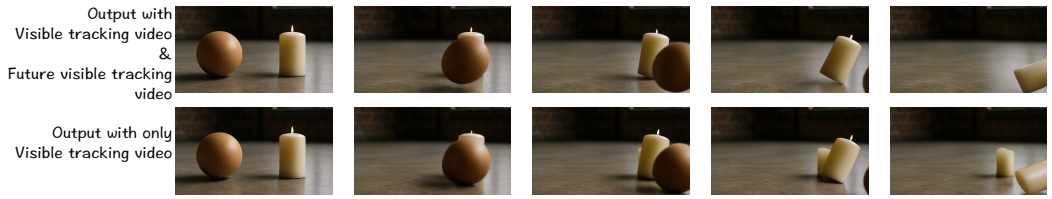

Figure 9: **Ablation on future-emergent motion signals.** Top: our full model with both reference-visible and future-emergent tracking signals. Bottom: model using only reference-visible tracking.

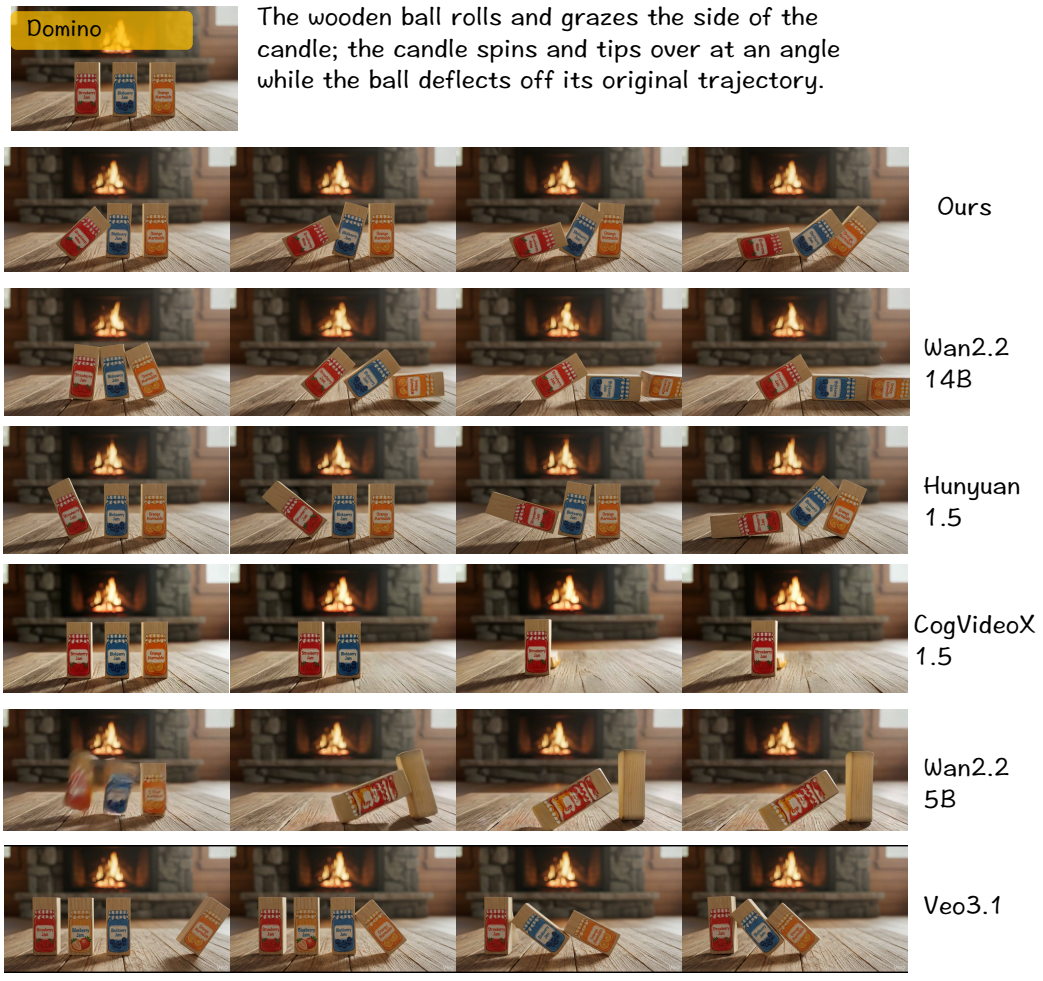

Figure 10: Qualitative comparison of domino motion.

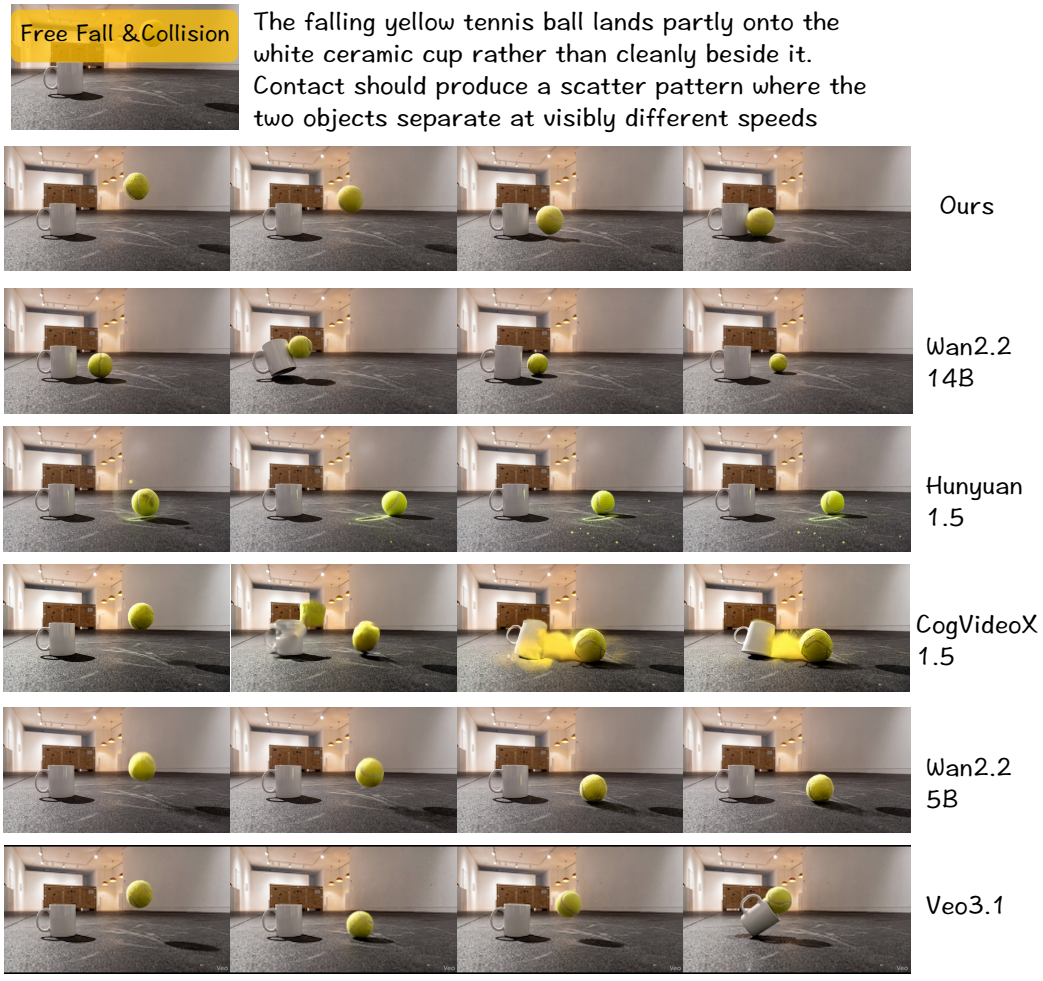

Figure 11: Qualitative comparison on free fall & collision motion.

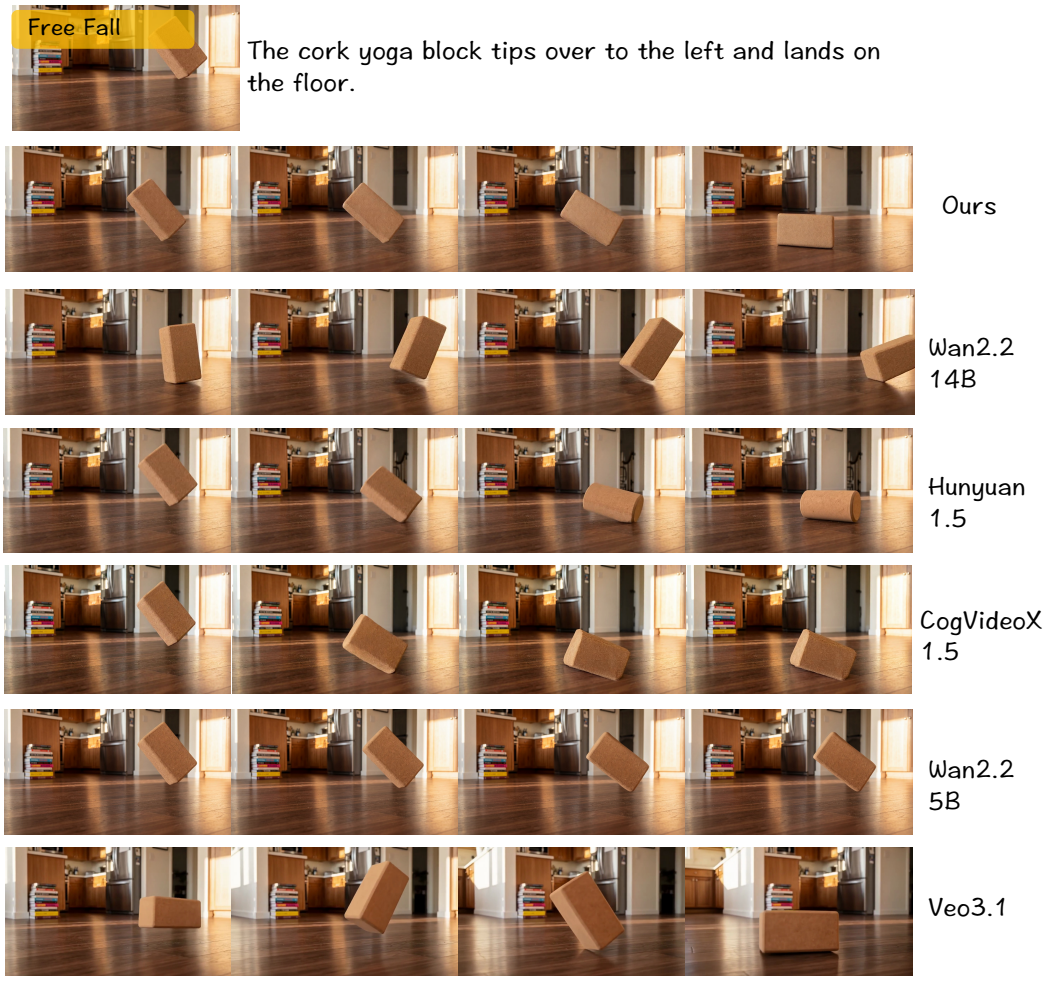

Figure 12: Qualitative comparison of free-fall motion.

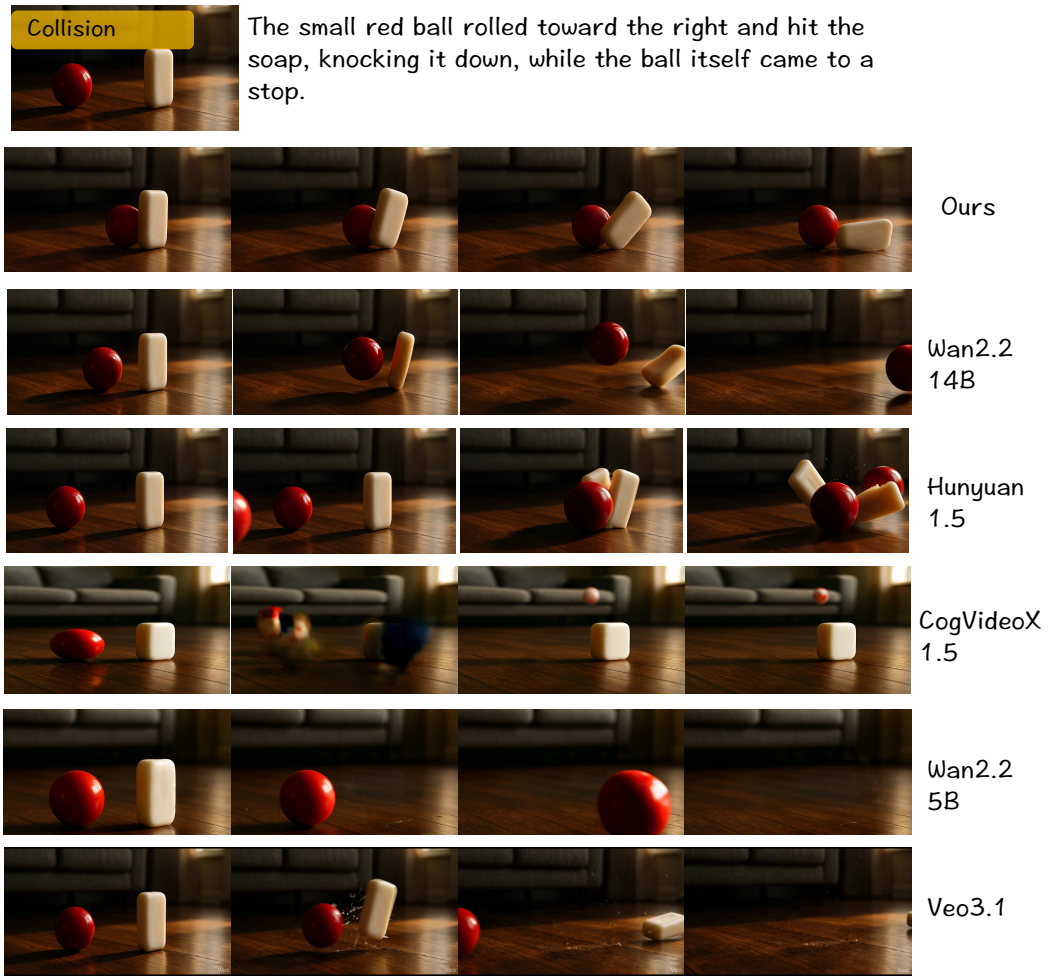

Figure 13: Qualitative comparison of collision motion.

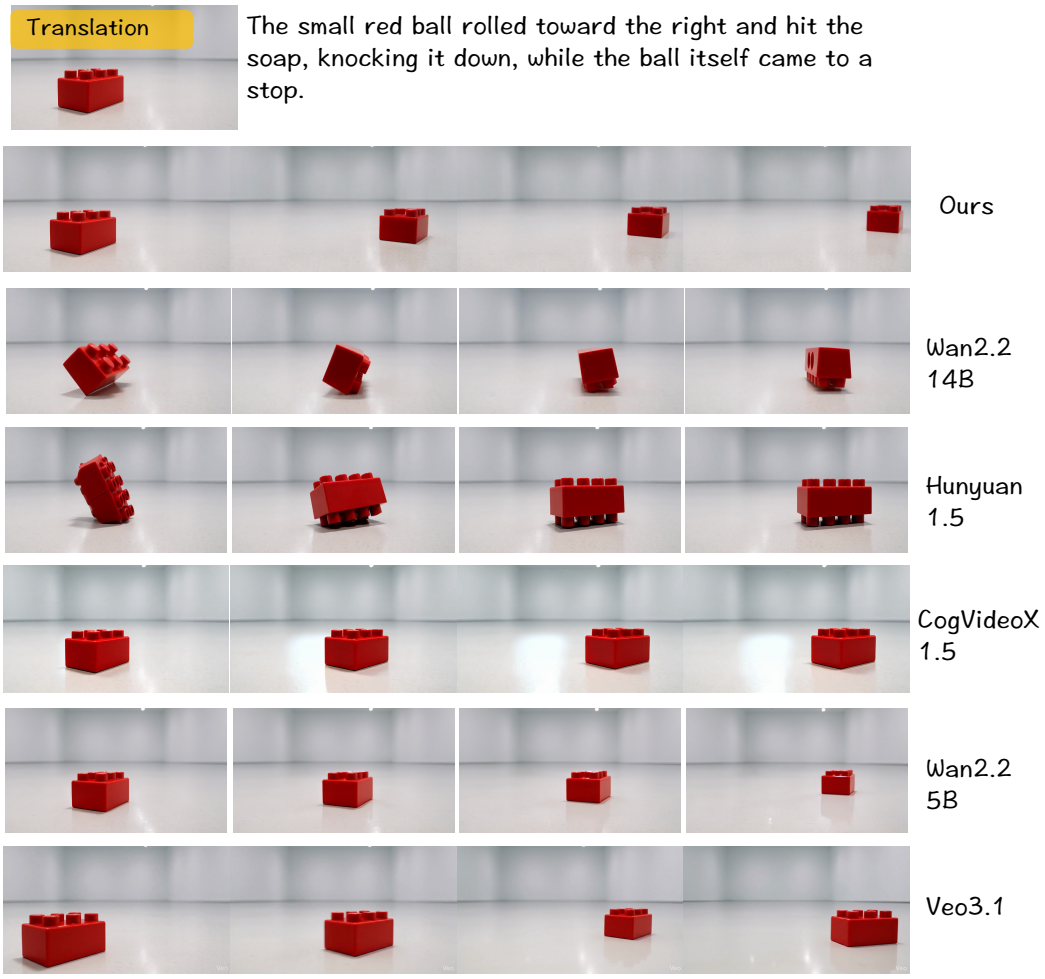

Figure 14: Qualitative comparison of translation motion.

G Robustness to Upstream Perception Errors

We assess how incomplete motion guidance and upstream perception errors affect downstream video generation by perturbing the full pipeline in two ways: randomly dropping approximately 25% of object tracks from the tracking videos, and removing 1–2 objects segmented by SAM3. We apply these perturbations separately and jointly, and compare the resulting videos with those produced by the clean full pipeline and the original Wan2.2-TI2V-5B backbone. Table 9 reports SA and PC scores across all seven video-generation scenarios, together with their averages and relative changes from the clean model.

Table 9: Robustness to incomplete motion guidance and upstream perception errors. Each entry reports SA / PC (↑). The Tracks variant drops approximately 25% of object tracks, the Objects variant removes 1–2 segmented objects, and Both applies both perturbations. Coll., Dom., and F.F. denote Collision, Domino, and Free Fall. Δ reports the relative change in average SA / PC (%) with respect to the clean full pipeline.

Scenario	**Ours (clean)**	**Ours (Tracks)**	**Ours (Objects)**	**Ours (Both)**	**Wan2.2 TI2V-5B**
Domino	3.976 / 4.013	3.777 / 3.852	3.578 / 3.732	3.459 / 3.612	2.650 / 3.033
Collision	4.160 / 4.394	3.952 / 4.218	3.744 / 4.086	3.619 / 3.955	2.917 / 3.067
Free Fall	4.347 / 4.328	4.130 / 4.155	4.086 / 4.155	3.912 / 4.025	2.183 / 3.083
Coll.+Dom.	3.854 / 4.016	3.661 / 3.855	3.469 / 3.735	3.353 / 3.614	2.200 / 3.033
F.F.+Dom.	3.985 / 4.017	3.786 / 3.856	3.587 / 3.736	3.467 / 3.615	2.467 / 3.033
F.F.+Coll.	4.327 / 4.589	4.111 / 4.405	3.894 / 4.268	3.764 / 4.130	2.183 / 3.083
Translation	4.013 / 4.578	3.812 / 4.395	3.772 / 4.395	3.612 / 4.258	3.000 / 4.100
Average	4.095 / 4.276	3.890 / 4.105	3.733 / 4.015	3.598 / 3.887	2.514 / 3.205
Δ (%)	–	-5.0 / -4.0	-8.8 / -6.1	-12.1 / -9.1	-38.6 / -25.0

Dropping tracks reduces average SA and PC by 5.0% and 4.0%, respectively, while removing segmented objects reduces them by 8.8% and 6.1%. Under both perturbations, the scores decrease by 12.1% and 9.1%, reaching 3.598 SA and 3.887 PC. These scores remain higher than those of the original backbone (2.514 SA and 3.205 PC), and the perturbed full pipeline outperforms the backbone in every scenario on both metrics. The results indicate that incomplete guidance degrades generation quality, while the framework retains useful motion guidance under the tested perturbations.

SpatialTracker2 [57] is used only to construct motion-supervision data for training. We filter out low-confidence point-cloud trajectories and retain key motion points, reducing reliance on accurate reconstruction of complete object shapes in the training data. At inference time, PGDS uses the geometry reconstructed by the 3D lifting pipeline. Remaining failure cases include inconsistent depth estimates for objects at similar depths, which ground-particle alignment can partially alleviate. Segmentation is also challenging in scenes with many small objects, particularly more than ten: some objects may be missed, or multiple objects may be merged into a single segment. Merged segments have a relatively limited impact on SAM 3D reconstruction, whereas missed objects can lead to missing particle trajectories in subsequent stages.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We provide comprehensive quantitative and qualitative evaluations in the experimental section to validate our core claims.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide limitation statement in our paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: Our contribution is centered on empirical validation rather than theoretical analysis.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide essential implementation details in the supplementary material and commit to releasing our training code and checkpoints to ensure full reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: While the code is not included with this submission, we plan to make our complete implementation, including pre-trained models and checkpoints, publicly available following the review process.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We elaborate on dataset construction, training protocols, and evaluation procedures within both the main manuscript and the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The paper reports experimental results with appropriate statistical information, including repeated runs and variance measurements (e.g., mean and standard deviation), which helps demonstrate the stability and reliability of the proposed method.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide a detailed breakdown of the computational requirements, including the number of GPUs and total training time for each experiment, in the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our paper conforms to the NeurIPS code of ethics in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed the impacts in the supplementary material.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We provide the safeguards in the supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide proper attribution by citing the original works whenever their code, datasets, or pre-trained models are utilized.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our training data are derived from existing datasets through a post-processing pipeline. We additionally construct a new evaluation benchmark, for which we provide detailed preprocessing procedures in the paper. To facilitate future research, we will release the processed data and benchmark upon publication.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We employed crowdsourcing via Amazon Mechanical Turk for human evaluation on generated videos. All participants were compensated at a rate of 20 dollars per hour, exceeding the local minimum wage.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

Justification: Our human evaluation protocol posed minimal risk to participants as it only involved viewing and scoring synthetic videos. We ensured all participants provided informed consent, and our study was conducted in accordance with the ethical guidelines of our institution. To maintain anonymity, the specific approving board is not named here but will be included in the final version.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We leverage the reasoning capabilities of Large Language Models (LLMs) to evaluate the quality of our generated videos, as detailed in the paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.