PHYSPLAN: GROUNDED PHYSICAL STATE REASONING AND GRAPH-GUIDED OPTIMIZATION FOR PHYSICALLY PLAUSIBLE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

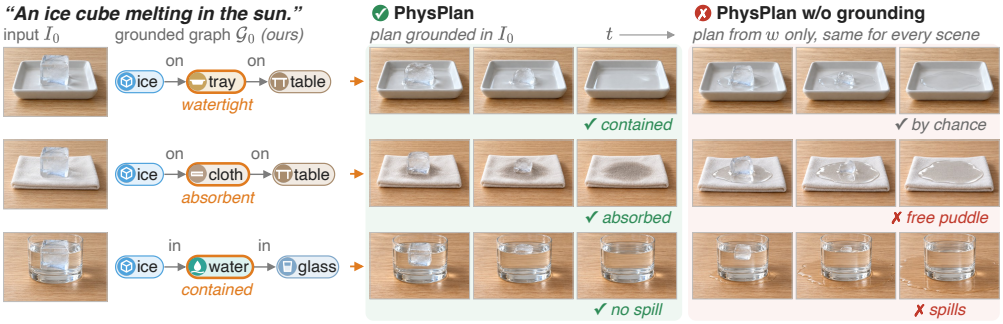


Figure 1: Same words, different scenes. "An ice cube melting in the sun" says what occurs; the scene decides where the meltwater goes: it stays in a watertight tray, soaks into an absorbent cloth, and joins the water in a glass. PhysPlan plans on a state graph grounded in the input frame. Without grounding, the plan comes from the prompt alone and yields the same free puddle in every scene.

ABSTRACT

Video diffusion models (VDMs) synthesize photorealistic content, yet they often fail to follow the course that a physical phenomenon should take within a given scene. Recent training-free methods let a vision-language model (VLM) plan the phenomenon and guide a frozen VDM toward the plan; however, such plans are derived from the prompt and consumed as whole keyframes or trajectories, which leaves unspecified where the consequences land in the observed scene and turns incidental visual details into optimization targets. We observe that a phenomenon specified in words unfolds as sparse, local changes to the physical state of the observed scene. Building on this observation, we present **PhysPlan**, a training-free image-to-video framework that represents a phenomenon as a grounded state graph and uses this graph to decide what, where, and when the guidance constrains. *Grounded Physical State Reasoning* decomposes the phenomenon into physical deltas, each stating which objects change, to what state, and by which physical rule, and translates each delta into graph edits, verified by deterministic checks, that leave all other objects unchanged. *Graph-Guided Test-Time Optimization* renders a keyframe for each state, measures the denoised estimates only along the properties selected by the edits, and concentrates the update on the edited objects. On PhyGenBench and Physics-IQ, PhysPlan raises its base model from 0.52 to 0.77 and from 27.1 to 38.2, surpassing the strongest prior I2V method (0.60 and 34.6), and lowers FVD by over 20%.

Project page: `physplan.github.io`.

1 INTRODUCTION

Video diffusion models (VDMs) (Blattmann et al., 2023; Yang et al., 2025b; HaCohen et al., 2024) synthesize photorealistic and temporally coherent videos, drawing attention to their potential as world simulators (Brooks et al., 2024). Yet VDMs mainly reproduce statistical patterns of their training data rather than the physical processes behind them (Kang et al., 2025): they often fail to follow the

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

course that a phenomenon should take, *e.g.*, an ice cube vanishes without leaving water behind, or uninvolved objects change their appearance (Motamed et al., 2026; Meng et al., 2025).

A growing line of work therefore decouples physical reasoning from visual synthesis: a language or vision-language model (VLM) plans the phenomenon as trajectories (Yang et al., 2025a), keyframes (Huang et al., 2026; Zhao et al., 2026), both (Zhuang et al., 2026), or chains of events over scene graphs (Wang et al., 2026), and the plan is injected into a frozen VDM at inference time. Two gaps remain. *First, the plan is derived from the prompt, whereas the outcome is determined by the scene.* Where the consequences of a phenomenon land depends on the physical relations among the observed objects (Figure 1). A prompt-derived plan does not know which passive objects the scene contains, and when states are described anew as complete scenes, nothing ties one state to the next, so the VLM may omit passive objects, alter uninvolved ones, or lose object identities. *Second, the plan is consumed as whole frames.* A keyframe commits to details the plan never intended, such as the exact contour of a puddle; guiding toward whole keyframes (Jang et al., 2026) turns these details into targets, and the gradient disturbs content that should remain unchanged.

We observe that a phenomenon unfolds as *sparse, local changes to the physical state of the observed scene*: a few objects and relations change at a time, while the rest persists. We therefore represent the phenomenon as changes to a state graph grounded in the observed frame, and let the same graph decide what the guidance constrains. Based on this idea, we propose **PhysPlan**, a general training-free image-to-video framework with two modules. *Grounded Physical State Reasoning* (GPSR) parses the observed frame into a state graph and decomposes the phenomenon into causally ordered *physical deltas*, each stating the new state of every changing object, the qualitative physical rule that produces it, and the share of the video the event takes. The same VLM translates each delta into typed graph edits, verified by deterministic checks, so every object outside the edits is preserved by construction. *Graph-Guided Test-Time Optimization* (GTO) renders a keyframe for each state and guides a frozen VDM by measuring its denoised estimates at the anchor frames only along the properties selected by the edits, with the update concentrated on the edited objects. Since the modules communicate only through the state graph, all models are used off the shelf and can be replaced. Our contributions are:

- PhysPlan, a general training-free framework in which a state graph grounded in the observed frame determines what, where, and when a frozen VDM is guided.
- Grounded Physical State Reasoning, which evolves the state graph only through physical deltas realized as verified edits, making every change traceable and every unchanged object explicit.
- Graph-Guided Test-Time Optimization, which selects the measured properties by the edit types, compares objects rather than pixels, and weights the update by the regions of the edited objects.
- On PhyGenBench (Meng et al., 2025) and Physics-IQ (Motamed et al., 2026), PhysPlan raises its base model from 0.52 to 0.77 and from 27.1 to 38.2 while improving FVD, VBench quality, and human preference; ablations confirm that both gaps matter.

2 Related Work

Physically plausible video generation. Large-scale VDMs (Blattmann et al., 2023; Yang et al., 2025b; HaCohen et al., 2024; Zheng et al., 2024; Wan Team et al., 2025) achieve high visual fidelity but often violate physical laws (Meng et al., 2025; Motamed et al., 2026), and scaling alone does not resolve this gap (Kang et al., 2025). Simulator-based methods (Liu et al., 2024) model exact dynamics but are limited to supported phenomena, while training-based methods (Chen et al., 2025; Cai et al., 2026) require curated data and backbone-specific retraining. Prompt-based methods (Xue et al., 2025) add physical details to prompts but remain ambiguous about scene-specific outcomes. Inference-time methods instead suppress implausible dynamics with counterfactual prompts (Hao et al., 2025) or refine samples using the generator's scores (Hassan et al., 2026). PhysPlan is training- and simulator-free, while imposing positive, object-level constraints grounded in the observed scene.

Reasoning-guided video generation. Early planners predict layouts for images (Feng et al., 2023) and videos (Lian et al., 2024; Lin et al., 2024), but not physical state changes. Recent methods plan physical phenomena as trajectories (Yang et al., 2025a), keyframes (Huang et al., 2026; Zhao et al., 2026), both (Zhuang et al., 2026), or events over evolving scene graphs (Wang et al., 2026). These plans are derived from the prompt rather than the observed scene and are typically consumed as complete scenes; even Wang et al. (2026) conditions the VDM on whole keyframes. PhysPlan instead grounds the graph in the observed frame and evolves it through verified edits, following the frame problem (McCarthy & Hayes, 1969). PhysPlan relies on VLMs for what they do well: qualitative

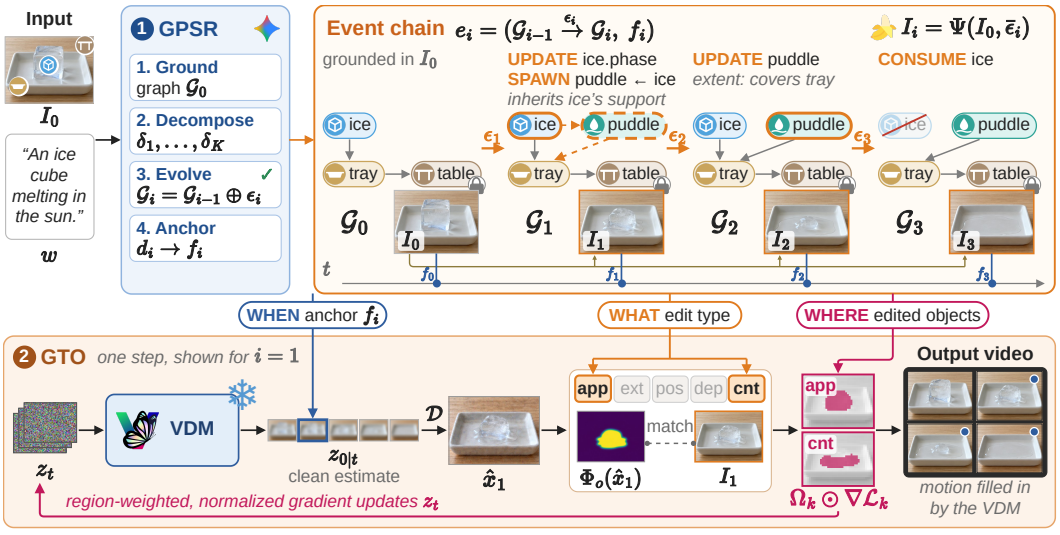


Figure 2: Overview of PhysPlan. Top: GPSR (Section 3.1) grounds I_0 and w as a state graph and evolves it through verified edits ϵ_i into an event chain, anchoring each state $\mathcal{G}_i$ to a frame f_i; unedited objects are carried over (lock). Each keyframe I_i is rendered from I_0 and the net edits $\bar{\epsilon}_i$. **Bottom:** GTO (Section 3.2), shown for $i = 1$: the anchor selects *when* the clean estimate of the frozen VDM is compared with I_i, the edit type selects *what* is measured, and the edited objects define *where* the gradient acts. The VDM fills in the motion between anchors. Icons: Gemini 3 Flash (VLM), Gemini 3 Pro Image (Ψ), CogVideoX-I2V-5B (VDM).

physical reasoning, such as judging which objects a phenomenon affects, how their states change, and in what order. Their quantitative estimates remain unreliable (Li et al., 2025), so PhysPlan never asks for them.

Training-free guidance of diffusion models. Loss-based guidance steers pretrained diffusion models by differentiating objectives on estimated clean samples (Bansal et al., 2024; Yu et al., 2023; Shen et al., 2024). Frame Guidance (Jang et al., 2026) extends this to large VDMs by decoding latent slices around guided frames, but matches each target as a whole, e.g., with an L2 keyframe loss, making unspecified details optimization targets and potentially altering unchanged content. PhysPlan instead compares anchors with keyframes at the object level and only on properties selected by graph edits, while restricting updates to the corresponding edited regions.

3 Proposed Method

Given an observed frame I_0 and a text prompt w describing a physical phenomenon, our goal is to generate a video of F frames that starts from I_0 and is physically plausible. This involves deciding what should happen in the scene and making the VDM generate it. PhysPlan addresses these with two modules (Figure 2): *Grounded Physical State Reasoning* (GPSR, Section 3.1) predicts how the scene should change, and *Graph-Guided Test-Time Optimization* (GTO, Section 3.2) guides a frozen VDM to follow this prediction. The two modules are connected by an *event chain*

$$\mathcal{G}_0 \xrightarrow{e_1} \mathcal{G}_1 \xrightarrow{e_2} \cdots \xrightarrow{e_K} \mathcal{G}_K, \qquad e_i = \big(\mathcal{G}_{i-1} \xrightarrow{\epsilon_i} \mathcal{G}_i,\ f_i\big), \tag{1}$$

where $\mathcal{G}_i$ is a state graph, ϵ_i is the graph edits of the i-th event, and f_i is the frame at which $\mathcal{G}_i$ should hold. Thus, each event specifies what to measure through its edit type, where through its edited objects, and when through its anchor; the VDM generates the remaining content and motion.

3.1 Grounded Physical State Reasoning

The effect of a phenomenon depends on the physical relations in the scene: melting ice wets the tray it rests on, but not the table beside it (Figure 1), and only a few objects change at a time. We therefore represent the scene as a state graph grounded in I_0 and the phenomenon as a sequence of small edits

Table 1: Operator set $\mathcal{O}$ and the guidance term that measures each edit (Section 3.2).

Operator	Effect on the state graph	Measured by
UPDATE(o, α, s)	set attribute α (other than extent) to s	appearance, Eq. (7a)
UPDATE(o, α, s)	set the extent of o to s	area, Eq. (7b)
LINK/UNLINK(a, r, b)	add/remove a planar relation	location, Eq. (7c)
LINK/UNLINK(a, r, b)	add/remove a depth relation	depth order, Eq. (7d)
SPAWN$(o' \leftarrow o)$, CONSUME(o)	add o' with lineage o; remove o	presence, Eq. (6)

to it (Figure 3). Since VLMs are unreliable at physical quantitative reasoning (Chow et al., 2025; Tang et al., 2026; Li et al., 2025), the edits are qualitative and verified by deterministic checks, and the VLM never regenerates a full state, so objects that should not change stay unchanged.

Grounded Phenomenon Decomposition. A VLM parses I_0 and w into a state graph $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ that records the state of each object and how objects relate. *Nodes* are all objects visible in I_0, not only those named in w: a prompt about melting ice never mentions the tray, yet the water spreads onto it. Each node has a persistent identifier, a category, and attributes under six fixed keys with open-vocabulary values: phase, integrity (*e.g.*, cracked), surface (*e.g.*, wet), color, extent (size), and configuration (pose or shape, *e.g.*, tipped over). Fixed keys make states comparable across graphs; open values describe phenomenon-specific states. *Edges* (a, r, b) take r from a fixed set, since each relation type is checked and guided differently: physical relations (support, contact, containment, attachment) carry effects between objects, and spatial relations (left of, above, in front of, near) fix the layout. Relations observable in the image plane are *planar* and guided by $\mathcal{L}_{\text{pos}}$; containment and in front of are *depth* relations, guided by $\mathcal{L}_{\text{depth}}$ (Section 3.2).

The VLM then decomposes the phenomenon into a causally ordered sequence of physical deltas $\Delta = (\delta_1, \ldots, \delta_K)$. Each delta $\delta_i = (\mathcal{S}_i, d_i)$ describes the state of the scene at the end of one event: $d_i \in (0, 1)$ is the fraction of the video that the event takes, and each entry $(o, s_o, r_o) \in \mathcal{S}_i$ gives a changing object o, its new state s_o, and the physical rule r_o that produces it. Rules are short qualitative statements without numbers: the VLM judges which objects change, how, and in what order, rather than estimating quantities (Li et al., 2025); requiring a rule for every state makes each change follow from a cause. For example, melting ice gives (ice#1, *partially melted*, *ice above its melting point turns into water*) and (tray#2, *wet*, *liquid flows down onto its support*). A delta covers all objects that change in the event, since unrelated objects can change at the same time (*e.g.*, ice and butter on separate plates). Objects are referred to by node identifiers, new objects get identifiers linked to their source, and unlisted objects stay unchanged. The whole sequence is predicted in one pass.

Delta-driven Graph Evolution. As text, the deltas cannot be checked or measured, and letting the VLM rewrite the full graph would also change objects that the delta does not mention. Instead, the same VLM translates each delta into typed edits $\epsilon_i = \text{VLM}(\mathcal{S}_i, \mathcal{G}_{i-1})$ from the operator set $\mathcal{O}$ (Table 1), and the next graph is $\mathcal{G}_i = \mathcal{G}_{i-1} \oplus \epsilon_i$, where $\oplus$ applies the edits. For example, (tray#2, *wet*) becomes UPDATE(tray#2, surface, wet), and ice turning into a puddle becomes SPAWN(puddle#3 $\leftarrow$ ice#1) and CONSUME(ice#1). The edits are verified by four deterministic checks: (i) *grounding*, edits use existing or newly spawned nodes and valid keys or relation types; (ii) *coverage*, exactly the objects in $\mathcal{S}_i$ are edited; (iii) *lineage*, nodes are added only by SPAWN and removed only by CONSUME; and (iv) *consistency*, no attribute is set twice, and support and containment stay acyclic. Violations are returned to the VLM, which retries. Unedited nodes are copied unchanged, so every change comes from a delta. Each edit type also sets how its change is measured (Table 1).

Event Timing Estimation. Physical changes are not uniform in time: an object may soften slowly and then collapse quickly. Since VLMs are unreliable at absolute durations (Li et al., 2025), the VLM only divides the video among the events through the fractions d_i, a relative judgment; a slow phenomenon may thus be compressed, as in a time-lapse. We require $\sum_{i \le K} d_i < 1$, so that the final state stays visible at the end, and rescale the fractions otherwise. The anchor of $\mathcal{G}_i$ is the end of event i; here, $\lfloor \cdot \rceil$ denotes rounding to the nearest frame. The resulting anchors complete the event chain in Eq. (1):

$$f_i = \left\lfloor \left(\sum_{j \le i} d_j \right) (F - 1) \right\rceil, \qquad f_0 = 0. \tag{2}$$

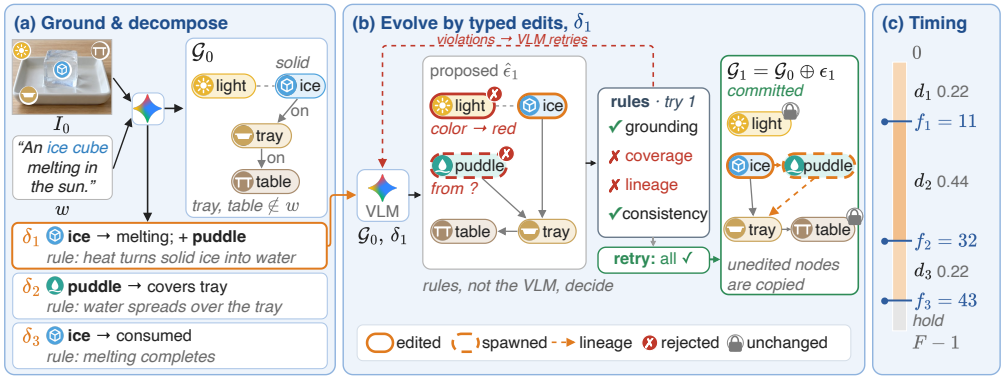


Figure 3: Grounded Physical State Reasoning. (a) The VLM grounds the scene as $\mathcal{G}_0$, including objects not mentioned in w, and decomposes the phenomenon into deltas with qualitative rules. (b) Each delta is translated into typed edits. Deterministic checks reject invalid proposals, here an edit to an object the delta does not name (coverage) and a new object without a source (lineage); the VLM retries, and only a passing proposal is committed, with all other nodes copied. (c) The share d_i of each event places its state at an anchor frame f_i.

3.2 Graph-Guided Test-Time Optimization

A VDM can be steered toward target frames at inference time by comparing its predicted frames with them (Jang et al., 2026), so a keyframe of each state is a natural target. A keyframe, however, contains more than the graph specifies, such as the exact contour of a puddle, and matching it pixel by pixel would impose these details on the video. We therefore compare the video with each keyframe only on the properties selected by the edits, and update it only in the regions of the edited objects.

Keyframe Rendering. Before denoising, we collect for each event the net edits $\bar{\epsilon}_i$: all edits up to event i, keeping the last value of each attribute and removing edits that cancel out (*e.g.*, a node spawned and later consumed), which describe how $\mathcal{G}_i$ differs from $\mathcal{G}_0$. They are written as an editing instruction, in which lineage tells where new objects appear (*e.g.*, a puddle where the ice was), and an image editor Ψ renders the keyframe I_i from I_0. Editing every keyframe from I_0 rather than from the previous one prevents errors from accumulating. Object masks $M_o(\cdot)$ and depth maps are then extracted with an open-vocabulary segmenter and a monocular depth estimator.

Preliminary. Following Frame Guidance (Jang et al., 2026), at denoising step t the clean latent is estimated from z_t; for a v-prediction VDM v_θ, Tweedie's formula (Efron, 2011) gives

$$z_{0|t} = \sqrt{\bar{\alpha}_t}\, z_t - \sqrt{1-\bar{\alpha}_t}\, v_\theta(z_t, t), \tag{3}$$

and the corresponding estimate for flow-matching models (Lipman et al., 2023) is given in the appendix. To save computation, only the slice of $z_{0|t}$ around each anchor is decoded into a preview $\hat{x}_i = \mathcal{D}\big(z_{0|t}^{(f_i)}\big)$, where $z^{(f_i)}$ denotes the latent frames that cover frame f_i under the temporal compression of the VAE. The preview and the keyframe I_i are then compared object by object.

Graph-Selected Measurement. Comparing an object requires locating it in the preview in a form that gradients can pass through. A segmenter cannot do this: its masks are not differentiable, and it is unreliable on the blurry previews of early steps. We therefore locate an object o by a soft occupancy map $\Phi_o(x) \in [0,1]^{H\times W}$, which is high wherever image x looks like the object:

$$\Phi_o(x)(u) = \sigma\Big(\big(\cos(\mathcal{F}(x)(u), \rho_o) - b\big)/\tau\Big), \tag{4}$$

$$\rho_o = \mathrm{avg}_{M_o(I_i)}\, \mathcal{F}(I_i), \tag{5}$$

where u is a grid location, $\mathcal{F}$ a frozen feature encoder, b and τ a similarity threshold and a temperature, and $\mathrm{avg}_A\, g = \frac{1}{|A|}\sum_{u\in A} g(u)$ the mean of a map g over a region A. The reference feature ρ_o is the mean feature of the object in the keyframe, or in I_0 for a consumed object. Φ_o acts as a soft mask that changes smoothly with the preview: raising it at a location makes the object appear there, and lowering it makes the object disappear, so the losses below can move, grow, add, or remove an object.

Since the VDM may duplicate, lose, or split an object, we compare instances rather than whole objects. The preview instances $\{P_j\}_{j=1}^{J}$ are the connected components of $\{u : \Phi_o(\hat{x}_i)(u) > 0.5\}$, and the keyframe instances $\{Q_l\}_{l=1}^{L}$ those of the mask. The Hungarian algorithm pairs them one-to-one into the matching π^* of $\min(J, L)$ pairs with the lowest total centroid distance $\sum_{(j,l)\in\pi} \|\bar{c}(P_j) - \bar{c}(Q_l)\|$, where $\bar{c}(\cdot)$ is the centroid of an instance, and $j \notin \pi^*$ marks an unpaired preview instance j (likewise for l). An unmatched instance signals a wrong number of copies, which SPAWN and CONSUME change, so we measure it by suppressing unmatched preview instances and filling in unmatched keyframe ones:

$$\begin{aligned}\mathcal{L}_{\text{count}}(o) = {} & \textstyle\sum_{j\notin\pi^*} \operatorname{avg}_{P_j} \Phi_o(\hat{x}_i) \\ & + \textstyle\sum_{l\notin\pi^*} \max\big(0,\, 1 - \operatorname{avg}_{Q_l} \Phi_o(\hat{x}_i) / \operatorname{avg}_{Q_l} \Phi_o(I_i)\big).\end{aligned} \tag{6}$$

For example, a consumed object has no keyframe instance, so all its preview instances are suppressed, and a spawned object missing from early previews is filled in.

Each matched pair is then compared only on the property that its edit changes. With $R_{jl} = P_j \cup Q_l$, which covers where the object is and where it should be, the terms are, summing over $(j, l) \in \pi^*$,

$$\mathcal{L}_{\text{app}} = \textstyle\sum 1 - \cos\big(\bar{\mathcal{F}}_{P_j}(\hat{x}_i),\, \bar{\mathcal{F}}_{Q_l}(I_i)\big), \tag{7a}$$

$$\mathcal{L}_{\text{ext}} = \textstyle\sum \log^2\big(|\Phi_o(\hat{x}_i) \odot R_{jl}| / |\Phi_o(I_i) \odot R_{jl}|\big), \tag{7b}$$

$$\mathcal{L}_{\text{pos}} = \textstyle\sum \big\| c_{jl}(\hat{x}_i) - c_{jl}(I_i) \big\|^2, \tag{7c}$$

$$\mathcal{L}_{\text{depth}} = \big(\Delta z_{ab}(\hat{x}_i) - \Delta z_{ab}(I_i)\big)^2, \tag{7d}$$

where $\bar{\mathcal{F}}_A(x) = \operatorname{avg}_A \mathcal{F}(x)$, $|\cdot|$ sums a map, $c_{jl}(x)$ is the centroid of $\Phi_o(x)$ within R_{jl}, and Δz_{ab} is the difference between the mean normalized depths of a and b, weighted by occupancy in the preview. The terms measure appearance, area as a scale-invariant ratio, location, and depth order. Area and location are compared on R_{jl} rather than on the preview mask, since a loss only changes the occupancy inside its region and the object must be able to grow or move toward its target. For an edit on a pair (a, b), the term is computed for both objects. Since every term compares an object-level summary, not pixels, it ignores the exact contour drawn by the editor.

The objective at event i sums the term selected by each edit and a count term for every object:

$$\mathcal{L}^{(i)} = \sum_{\epsilon\in\bar{\epsilon}_i} \mathcal{L}_{\mu(\epsilon)}(\epsilon) + \sum_{o\in\mathcal{V}_{\le i}} \mathcal{L}_{\text{count}}(o), \tag{8}$$

where $\mu(\epsilon) \in \{\text{app}, \text{ext}, \text{pos}, \text{depth}\}$ is the term selected by edit ϵ (Table 1), and $\mathcal{V}_{\le i}$ contains every object that has appeared up to event i. The first sum excludes SPAWN and CONSUME, which the second measures; the second also covers unchanged objects, so the VDM neither loses nor duplicates them. We write $\mathcal{L}_k^{(i)}$ for the k-th term; all terms have equal weight, since each gradient is normalized (Eq. (10)). Properties that the graph does not select give no gradient: a melting event is judged on the material and size of the object, not on the shape of the liquid.

Region-Weighted Update. Although each term looks only at its objects, its gradient spreads over the whole latent through the large receptive fields of the decoder and denoiser, and would change regions that should stay fixed. We therefore restrict each term to a region built from its instance masks,

$$\Omega_k = \bigcup_{(j,l)\in\pi^*} \operatorname{hull}\big(P_j \cup Q_l\big) \;\cup\; \bigcup_{j\notin\pi^*} P_j \;\cup\; \bigcup_{l\notin\pi^*} Q_l, \tag{9}$$

where $\operatorname{hull}(\cdot)$ is the convex hull, covering where a matched object is, where it should be, and the space between, and the unmatched instances are included so that they can be suppressed or filled in. Ω_k is average-pooled to the latent grid, and the update is

$$z_t \leftarrow z_t - \eta \textstyle\sum_{i=1}^{K} \sum_k \big(\Omega_k + \lambda(1 - \Omega_k)\big) \odot \frac{\nabla_{z_t} \mathcal{L}_k^{(i)}}{\|\nabla_{z_t} \mathcal{L}_k^{(i)}\|}, \tag{10}$$

where η is the step size and $\lambda \in [0, 1]$ reduces the update outside the region ($\lambda = 1$ gives unweighted guidance). As in Frame Guidance, gradients are normalized to unit norm, here per term, since the terms have different scales. The same spatial region is applied at every temporal index: the gradient at an anchor reaches all frames through the denoiser, and the weighting limits it in space but not in time. Following Frame Guidance, Eq. (10) is applied N_g times per step while the layout forms ($t > t_E$), the unweighted gradient is applied with re-noising for $t_E \ge t > t_L$, and the remaining steps are unguided. Details are in Section A.5.

Table 2: Physical plausibility on PhyGenBench (PCA score in $[0, 1]$) and Physics-IQ; higher is better, and averages are weighted by the number of samples per domain. Best in **bold**, second best underlined; subscripts give the gain of PhysPlan over its base model, CogVideoX-I2V-5B. T2V results are reported by Wang et al. (2026), except LTX-Video and CausalMotion (Zhuang et al., 2026); VLIPP is reported by Yang et al. (2025a).

	PhyGenBench					Physics-IQ					
Model	Mech.	Optics	Thermal	Material	**Avg.**	S.M.	F.D.	Optics	Magn.	Thermo.	**Avg.**
Text-to-video models											
Kling	0.45	0.58	0.50	0.40	0.49						
Wan2.2-14B	0.53	0.61	0.58	0.43	0.54						
CogVideoX-5B	0.39	0.55	0.40	0.42	0.45						
+ PhysHPO	0.55	0.68	0.50	0.65	0.61						
+ Chain-of-Events	0.70	**0.79**	**0.77**	0.64	0.73						
LTX-Video	0.35	0.45	0.36	0.38	0.39						
+ CausalMotion	0.61	0.71	0.68	0.61	0.65						
Image-to-video models											
CogVideoX-I2V-5B	0.48	0.69	0.43	0.41	0.52	30.4	29.8	16.7	13.3	8.5	27.1
SVD-XT	0.46	0.68	0.48	0.41	0.52	21.9	20.5	6.8	8.4	**17.1**	19.1
LTX-Video-I2V	0.47	0.65	0.46	0.37	0.50	30.2	29.8	15.9	13.2	8.4	26.8
Reasoning-guided and training-free guided I2V methods											
VLIPP	0.55	0.71	0.60	0.53	0.60	42.3	**34.1**	16.9	13.4	8.8	34.6
Frame Guidance	0.52	0.56	0.47	0.48	0.51	35.4	27.4	24.1	13.9	8.4	30.3
PhysPlan (Ours)	**0.81**	0.78	0.74	**0.75**	**0.77** $_{+0.25}$	**45.6**	31.8	**30.4**	**19.7**	9.5	**38.2** $_{+11.1}$

4 Experiments

4.1 Setup

Implementation. GPSR uses Gemini 3 Flash (Gemini Team et al., 2023) with JSON-constrained outputs. Keyframes are rendered with Gemini 3 Pro Image (Gemini Team et al., 2023), masks with Grounded-SAM-2 (Ravi et al., 2025), depth with Depth Anything V2 (Yang et al., 2024), and occupancy with DINOv3 features (Siméoni et al., 2025). The frozen VDM is CogVideoX-I2V-5B (Yang et al., 2025b), with guidance defaults from Frame Guidance (Jang et al., 2026) (Section A.5).

Benchmarks and baselines. PhyGenBench (Meng et al., 2025) covers 27 phenomena in four domains; since it is a text-to-video (T2V) benchmark, we generate an initial frame for each prompt with the Gemini API (Gemini Team et al., 2023), shared by all I2V methods. Physics-IQ (Motamed et al., 2026) contains 396 real-world scenarios in five domains. We compare with I2V models (CogVideoX-I2V-5B, SVD-XT (Blattmann et al., 2023), LTX-Video-I2V (HaCohen et al., 2024)), VLIPP (Yang et al., 2025a) (reported), and Frame Guidance (Jang et al., 2026), which we run with the same model, protocol, anchors, and keyframes as PhysPlan but with its original L2 loss to the whole keyframe. On PhyGenBench we also list T2V foundation models (Zheng et al., 2024; Yang et al., 2025b; Wan Team et al., 2025; Kling Team et al., 2025) and physics-aware methods (Xue et al., 2025; Chen et al., 2025; Wang et al., 2026; Cai et al., 2026; Zhuang et al., 2026) as reported in prior work (full list in Section B).

4.2 Physical Plausibility

PhyGenBench. PhysPlan reaches 0.77 on average (Table 2), against 0.52 for its base model and 0.60 for VLIPP, the strongest prior I2V method. The largest gains are in Material (+0.34), Mechanics (+0.33), and Thermal (+0.31), which are dominated by changes of object state that the graph expresses directly as edits. The gain in Optics is smaller (+0.09), since optical phenomena change how light passes through the scene rather than object states. Frame Guidance, which matches the same keyframes as whole frames, does not improve over its base model (0.51 vs. 0.52), whereas PhysPlan gains +0.26 from them. The T2V event chain of Wang et al. (2026) reaches 0.73 and is ahead in Optics and Thermal, though T2V and I2V results are not directly comparable.

Physics-IQ. PhysPlan reaches 38.2, against 27.1 for its base model, 30.3 for Frame Guidance, and 34.6 for VLIPP. The largest gains are in Solid Mechanics (+15.2), Optics (+13.7), and Magnetism (+6.4), where the outcome depends on which objects move or change while the rest of these cluttered real scenes stays fixed. Frame Guidance improves over its base model but lowers Fluid Dynamics and Thermodynamics, whereas PhysPlan improves in every domain. It remains below VLIPP in Fluid

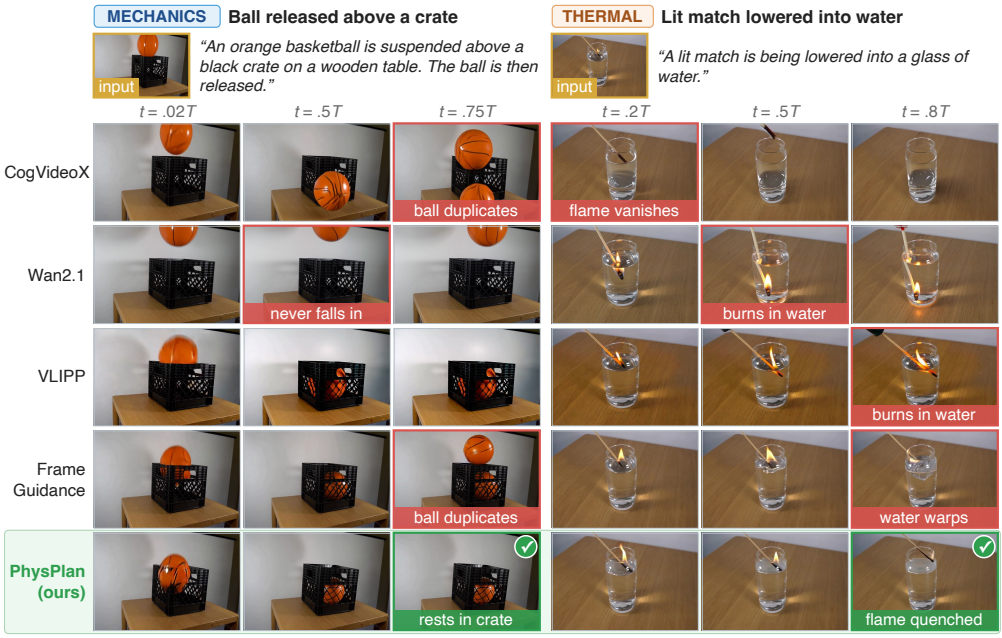


Figure 4: Qualitative comparison. Frames at matched times t/T for a mechanics (*left*) and a thermal (*right*) scene. Red marks a physical error, green the correct outcome: only PhysPlan keeps a single ball that lands in the crate, and only PhysPlan extinguishes the flame when the match enters the water.

Dynamics (31.8 vs. 34.1), since continuous flow between anchors is left to the VDM while VLIPP plans trajectories, and all guided methods stay near the base model in Thermodynamics (9 videos).

Qualitative Evaluation. In Figure 4, the baselines duplicate the ball, let it miss the crate, keep the match burning under water, or warp the water. PhysPlan follows its event chain in both scenes: the single ball comes to rest in the crate, and the flame is quenched as the match enters the water.

4.3 Visual Quality

Guidance could trade visual quality for plausibility, so we evaluate it in three ways. On *distribution metrics* (Table 3), PhysPlan improves FID (Heusel et al., 2017) over its base model and Frame Guidance, and reduces FVD (Ge et al., 2024) by about 21% and 29%, which we attribute to the region-weighted update keeping the background stable. On *per-video scores* from VBench (Huang et al., 2024) (Table 9 in the appendix), which do not penalize deviations from a reference distribution, PhysPlan raises the Quality Score from 83.05 to 84.88 and improves six of seven dimensions, most in Dynamic Degree (+6.11); it also exceeds closed-source VDMs in Quality Score and Dynamic Degree. In a two-alternative forced-choice *user study* with 60 participants against CogVideoX-I2V-5B or Frame Guidance, PhysPlan was preferred in 72%, 60%, and 73% of the comparisons for physical plausibility, frame quality, and temporal smoothness (Section E). Plausibility thus does not cost visual quality.

Table 3: Perceptual quality on PhyGenBench and Physics-IQ, measured by FID and FVD. Best results are in **bold**.

Model	PhyGenBench		Physics-IQ	
	FID↓	FVD↓	FID↓	FVD↓
CogVideoX-I2V-5B	48.2	632.8	55.4	698.5
Frame Guidance	46.4	580.4	49.2	603.4
PhysPlan (Ours)	**45.4**	**500.2**	**47.7**	**495.6**

4.4 Ablation Study

(i) Prompt-derived plan: no state graph; the VLM describes each state as a complete scene, used as the keyframe instruction, with the whole-frame L2 loss. **(ii) Full-graph regeneration**: the VLM rewrites the full graph from each delta, and the edits are taken as differences between graphs. **(iii)**

Table 4: Ablation study on Physics-IQ. Each setting changes one component of PhysPlan and keeps all others fixed. S.M. refers to Solid Mechanics and F.D. to Fluid Dynamics; the average is weighted by the number of videos per category, and subscripts give its change from the full model. Best results are in **bold**.

	Setting	S.M. (↑)	F.D. (↑)	Optics (↑)	Magn. (↑)	Thermo. (↑)	Avg. (↑)
	PhysPlan (full)	**45.6**	**31.8**	**30.4**	**19.7**	**9.5**	**38.2**
Grounded Physical State Reasoning							
(i)	Prompt-derived plan	31.1	30.2	20.4	13.8	8.6	28.1 $_{-10.1}$
(ii)	Full-graph regeneration	35.2	30.2	23.6	15.9	8.9	30.9 $_{-7.3}$
Graph-Guided Test-Time Optimization							
(iii)	Whole-frame L2 loss (= Frame Guidance)	35.4	27.4	24.1	13.9	8.4	30.3 $_{-7.9}$
(iv)	w/o region weighting ($\lambda = 1$)	42.8	30.7	28.3	17.1	9.1	36.0 $_{-2.2}$
Measurement terms							
(v)	w/o $\mathcal{L}_{\text{app}}$	42.6	29.4	25.5	18.1	9.3	35.3 $_{-2.9}$
(vi)	w/o $\mathcal{L}_{\text{ext}}$	43.1	30.6	28.7	18.8	8.8	36.2 $_{-2.0}$
(vii)	w/o $\mathcal{L}_{\text{pos}}$	40.7	31.2	29.2	17.5	9.2	35.0 $_{-3.2}$
(viii)	w/o $\mathcal{L}_{\text{depth}}$	42.9	31.2	24.2	15.4	8.9	35.6 $_{-2.6}$

Whole-frame L2 loss: the graph-selected terms are replaced by the L2 loss of Frame Guidance without region weighting (the Frame Guidance baseline). **(iv) Without region weighting**: $\lambda = 1$ in Eq. (10). **(v)–(viii)**: one term of Eq. (7) is removed, leaving its edits (Table 1) unmeasured.

Results. Every component contributes, and the two gaps of Section 1 cause the largest drops (Table 4). Removing the state graph (i) costs 10.1 points, of which grounding alone accounts for 2.2 when compared with (iii), which uses the same loss. Regenerating the graph (ii) loses most in Solid Mechanics, since unmentioned objects can then change; matching the same keyframes as whole frames (iii) loses most in Solid Mechanics and Optics, where incidental keyframe details are imposed on the video; and removing the region weighting (iv) costs 2.2 points. Each term matters most where it is designed to: $\mathcal{L}_{\text{pos}}$ for Solid Mechanics, where outcomes depend on where objects move and come to rest; $\mathcal{L}_{\text{app}}$ for Fluid Dynamics and Optics, whose outcomes are visible as changes of appearance such as wet or transparent surfaces; and $\mathcal{L}_{\text{depth}}$ for Optics and Magnetism, whose outcomes often depend on which object is in front of or inside another.

5 Conclusion

We presented PhysPlan, a training-free framework that turns physical reasoning about the observed scene into guidance for a frozen VDM. GPSR grounds the phenomenon in a state graph of the observed frame and evolves it only through verified edits, so every change is traceable and every other object stays unchanged. GTO uses the resulting event chain to decide what to measure, where to update, and when each state should hold, comparing the video with keyframes object by object and only along the properties the edits select. PhysPlan achieves the best average scores on PhyGenBench (0.77) and Physics-IQ (38.2), far above its base model (0.52 and 27.1), and with the same keyframes it outperforms Frame Guidance by 0.26 and 7.9 points, while also improving visual quality and being preferred by participants in a user study. The ablations confirm both gaps that motivate its design. Our results suggest that the physical knowledge of VLMs is most useful to video generation when expressed as explicit, verifiable changes to the observed scene, which tell the generator both what should happen and what it should leave alone.

Limitations and future work. PhysPlan represents a phenomenon through discrete objects: every node, edit, and measurement refers to an object with an instance, a mask, and a centroid, and the video is constrained only at the anchor frames. This makes it most effective when a phenomenon is carried by clearly delineated objects, where it achieves its largest gains, *e.g.*, in Solid Mechanics, Optics, and Magnetism on Physics-IQ and in Mechanics and Material on PhyGenBench. It is weaker in Fluid Dynamics, where liquids that spread, splash, or split lack clear object boundaries and their continuous motion between anchors is left to the VDM. VLIPP stays ahead there by planning trajectories, although its continuous motion is interpolated from a few coarse waypoints and may itself deviate from the true dynamics. Future work includes representations for phenomena beyond discrete objects, such as fields or particle sets for fluids, combined with motion planning between anchors, as well as evaluation on further VDMs.

AI USE STATEMENT

LLMs/VLMs were used as components of the proposed system and experimental pipeline. Specifically, Gemini 3 Flash was used for grounded physical state reasoning, including scene parsing, phenomenon decomposition, graph-edit generation, and event timing estimation; Gemini 3 Pro Image was used to render keyframes from the graph-derived editing instructions; and the Gemini API was used to generate initial frames for the PhyGenBench evaluation.

ChatGPT was used solely for grammar correction and language polishing. All technical methodology, experiments, and results were developed, conducted, and verified by the authors.

REPRODUCIBILITY STATEMENT

Implementation details, exact VLM instructions, optimization pseudocode, evaluation protocols, and additional qualitative results are provided in the supplementary material. The complete source code and generated video samples will be released upon publication.

ETHICS STATEMENT

PhysPlan can improve the realism of generated videos and may therefore be misused to create deceptive synthetic media. We advocate provenance mechanisms such as digital watermarking and restrict the intended use of this work to creative, research, and educational applications. The de-identified human evaluation used adult participants who provided informed consent; no personally identifiable information was collected.

REFERENCES

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *International Conference on Learning Representations (ICLR)*, 2024.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. OpenAI Technical Report, 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

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

Harold Haodong Chen, Haojian Huang, Qifeng Chen, Harry Yang, and Ser Nam Lim. Hierarchical fine-grained preference optimization for physically plausible video generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

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

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. LayoutGPT: Compositional visual planning and generation with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 18225–18250, 2023.

Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the content bias in Fréchet video distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7277–7288, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. LTX-Video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Yutong Hao, Chen Chen, Ajmal Saeed Mian, Chang Xu, and Daochang Liu. Enhancing physical plausibility in video generation by reasoning the implausibility. *arXiv preprint arXiv:2509.24702*, 2025.

Mariam Hassan, Kaouther Messaoud, Wuyang Li, and Alexandre Alahi. Proprio: Latent self-scoring and inference-time refinement for physically plausible video generation. *arXiv preprint arXiv:2605.28230*, 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818, 2024.

Ziqi Huang, Ning Yu, Gordon Chen, Haonan Qiu, Paul Debevec, and Ziwei Liu. VChain: Chain-of-visual-thought for reasoning in video generation. In *Findings of the Association for Computational Linguistics (ACL)*, 2026.

Sangwon Jang, Taekyung Ki, Jaehyeong Jo, Jaehong Yoon, Soo Ye Kim, Zhe Lin, and Sung Ju Hwang. Frame guidance: Training-free guidance for frame-level control in video diffusion models. In *International Conference on Learning Representations (ICLR)*, 2026.

Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model: A physical law perspective. In *International Conference on Machine Learning (ICML)*, 2025.

Kling Team, Jialu Chen, Yuanzheng Ci, Xiangyu Du, Zipeng Feng, Kun Gai, Sainan Guo, Feng Han, Jingbin He, Kang He, et al. Kling-Omni technical report. *arXiv preprint arXiv:2512.16776*, 2025.

Puyin Li, Tiange Xiang, Ella Mao, Shirley Wei, Xinye Chen, Adnan Masood, Li Fei-Fei, and Ehsan Adeli. QuantiPhy: A quantitative benchmark evaluating physical reasoning abilities of vision-language models. *arXiv preprint arXiv:2512.19526*, 2025.

Long Lian, Baifeng Shi, Adam Yala, Trevor Darrell, and Boyi Li. LLM-grounded video diffusion models. In *International Conference on Learning Representations (ICLR)*, 2024.

Han Lin, Abhay Zala, Jaemin Cho, and Mohit Bansal. VideoDirectorGPT: Consistent multi-scene video generation via LLM-guided planning. In *Conference on Language Modeling (COLM)*, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. PhysGen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision (ECCV)*, pp. 360–378, 2024.

John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence 4*, pp. 463–502. Edinburgh University Press, 1969.

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. In *International Conference on Machine Learning (ICML)*, 2025.

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

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations (ICLR)*, 2025.

Yifei Shen, Xinyang Jiang, Yifan Yang, Yezhen Wang, Dongqi Han, and Dongsheng Li. Understanding and improving training-free loss-based diffusion guidance. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 108974–109002, 2024.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Tianyi Tang, Tianyi Ma, Zhuoyi Lin, Yew-Soon Ong, Haiyan Yin, Zeyu Feng, and Ivor Tsang. Causal scaffolding for physical reasoning: A benchmark for causally-informed physical world understanding in VLMs. *arXiv preprint arXiv:2606.05966*, 2026.

Wan Team, 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.

Zixuan Wang, Yixin Hu, Wen Li, Feng Chen, Yan Liu, Duo Peng, and Yinjie Lei. Physically plausible video generation via visual-semantic chain-of-events conditioning. *arXiv preprint arXiv:2609.00656*, 2026.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18826–18836, 2025.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 21875–21911, 2024.

Xindi Yang, Baolu Li, Yiming Zhang, Zhenfei Yin, Lei Bai, Liqian Ma, Zhiyong Wang, Jianfei Cai, Tien-Tsin Wong, Huchuan Lu, et al. VLIPP: Towards physically plausible video generation with vision and language informed physical prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12360–12370, 2025a.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations (ICLR)*, 2025b.

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. FreeDoM: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23174–23184, 2023.

Yibo Zhao, Hengjia Li, Xiaofei He, and Boxi Wu. PhyRPR: Training-free physics-constrained video generation. *arXiv preprint arXiv:2601.09255*, 2026.

Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-Sora: Democratizing efficient video production for all. *arXiv preprint arXiv:2412.20404*, 2024.

Sihan Zhuang, Xinyuan Chen, Tianfan Xue, and Yaohui Wang. CausalMotion: Structured physical reasoning as keyframe and trajectory guidance for training-free video generation. *arXiv preprint arXiv:2606.14317*, 2026.

Appendix

This appendix provides the method details, additional quantitative results, failure and cost analyses, and the user study for PhysPlan.

(A) Method Details **15**
(A.1) State Graph Vocabulary 15
(A.2) Deterministic Checks 15
(A.3) VLM Prompts 16
(A.4) Keyframe Rendering 17
(A.5) Guidance Details 18
(A.6) Formulation of the Guidance Loss 19
(A.7) Example Event Chain 21

(B) Additional Quantitative Results **21**

(C) Failure Analysis **21**

(D) Cost Analysis **22**

(E) User Study **23**

A METHOD DETAILS

This section gives the details of Grounded Physical State Reasoning (Section 3.1) and Graph-Guided Test-Time Optimization (Section 3.2) that are omitted from the main paper. The notation follows the main paper.

A.1 STATE GRAPH VOCABULARY

Table 5 defines the six attribute keys of a node. Keys are fixed and values are open-vocabulary; the examples are illustrative, not exhaustive. The boundary between attributes matters for the checks, so we fix it as follows. Integrity describes damage that keeps the object as one object; once the object separates into parts, the parts are spawned as new objects. Surface describes the outer layer without a change of the object itself: a wet tray is a surface change, whereas softening butter is a change of material phase. Rust, charring, and dirt are recorded as surface changes, and color is reserved for changes of hue alone.

Table 5: Attribute keys of a node. All attributes except extent are measured by the appearance term $\mathcal{L}_{\text{app}}$; extent is measured by the area term $\mathcal{L}_{\text{ext}}$.

Key	Describes	Example values
material phase	physical phase of the material	solid, softened, partially melted, liquid, gas
integrity	structural wholeness	intact, cracked, dented, torn, crushed
surface	condition of the outer layer	dry, wet, frosted, fogged, rusted, charred
color	hue of the object	green, brown, red
extent	size of the object	small, large, spread over most of the tray
configuration	pose or shape	upright, tipped over, open, closed, bent, crumpled

Table 6 defines the relation types. Spatial relations are written in one direction only (left of, above, in front of), so that the same layout is never recorded in two ways. The graph records states, not processes: actions such as pouring, heating, or pushing are not relations, and appear only through their results, such as a spawned stream of water, a changed phase, or a new support relation. Common situations that do not match a relation name are mapped to the fixed set by the conventions in Table 7, which are included in the parsing instruction.

Table 6: Relation types. Planar relations are measured by the location term $\mathcal{L}_{\text{pos}}$, depth relations by the depth term $\mathcal{L}_{\text{depth}}$.

Relation (a, r, b)	Kind	Measured as	Meaning
support	physical	planar	a holds b up against gravity
contact	physical	planar	a and b touch
containment	physical	depth	b is inside a
attachment	physical	planar	b is fixed to a
left of	spatial	planar	a is left of b in the image
above	spatial	planar	a is above b in the image
in front of	spatial	depth	a is closer to the camera than b
near	spatial	planar	a is close to b without contact

A.2 DETERMINISTIC CHECKS

Let $\mathcal{N}(\mathcal{S}_i)$ be the objects listed in $\mathcal{S}_i$ and $\mathcal{N}^+(\epsilon_i)$ the nodes spawned by ϵ_i. An edit set ϵ_i is accepted if it passes all four checks.

1. **Grounding.** Every node referenced by an edit is in $\mathcal{V}_{i-1}$ or in $\mathcal{N}^+(\epsilon_i)$; every attribute key is one of the six keys of Table 5; every relation type is one of Table 6; every spawned identifier is new.
2. **Coverage.** Every object in $\mathcal{N}(\mathcal{S}_i)$ is the target of at least one edit. UPDATE and CONSUME target only objects in $\mathcal{N}(\mathcal{S}_i)$, and every LINK or UNLINK has at least one endpoint in $\mathcal{N}(\mathcal{S}_i) \cup \mathcal{N}^+(\epsilon_i)$. The other endpoint may be an unlisted object, since adding a relation (*e.g.*, the tray now supports the puddle) does not change that object's attributes.

Table 7: Conventions for mapping common situations to relations.

Situation	Relations
object floating on a liquid (ice in water, a boat on a lake)	support by the liquid
object submerged in a liquid (a stone in water, a teabag in a cup)	containment
object leaning against another (a ladder against a wall)	support + contact
object hanging from another (a lamp from a ceiling)	attachment
part of an object (a handle on a cup)	attachment
layer covering an object (snow on a car, a cloth over a table)	support + surface change of the covered object
right of, below, behind	inverse of left of, above, in front of

3. **Lineage.** Every SPAWN names a source node in $\mathcal{V}_{i-1}$; nodes are added only by SPAWN and removed only by CONSUME; no edit acts on a node consumed earlier in the chain.
4. **Consistency.** No attribute of a node is set twice in ϵ_i; no edge is both linked and unlinked; after applying ϵ_i, the support and containment relations form directed acyclic graphs.

If a check fails, the list of violations is returned to the VLM with p_{edit}, for at most 3 retries. If the edits are still rejected, the delta is regenerated.

A.3 VLM PROMPTS

GPSR uses a single VLM (Gemini 3 Flash) with three instructions: scene parsing (p_{parse}), phenomenon decomposition (p_{delta}), and delta translation (p_{edit}). The **colored texts** indicate dynamic inputs. All outputs are constrained to strict JSON schemas, which we parse and validate before use.

Prompt 1: Scene Parsing (p_{parse}):
This instruction produces the state graph $\mathcal{G}_0$.

You are an expert in physical scene understanding. Your task is to describe the objects in an image and their physical relations as a state graph.
Inputs:
- Initial Image: **<INITIAL_FRAME>**
- User Prompt: **<TEXT_PROMPT>**

Step 1: Objects. List every object visible in the image, including objects that the prompt does not mention. Give each object an identifier of the form `<category>#<n>`, a category, and a value for each of the attribute keys: material phase, integrity, surface, color, extent, configuration.
Step 2: Relations. List all relations (a, r, b) between objects, with r from: support, contact, containment, attachment, left of, above, in front of, near. Follow these conventions: **<RELATION_CONVENTIONS>**.
Output Format & Example:
Return a structured JSON object exactly matching this schema:

```
{
  "nodes": [
    {"id": "ice#1", "category": "ice cube",
     "attributes": {"material_phase": "solid", "integrity": "intact",
                    "surface": "dry", "color": "clear",
                    "extent": "small", "configuration": "upright"}},
    {"id": "tray#2", "category": "tray", "attributes": {...}}
  ],
  "edges": [
    {"a": "tray#2", "r": "support", "b": "ice#1"}
  ]
}
```

Prompt 2: Phenomenon Decomposition (p_{delta}):
This instruction produces the physical deltas $\Delta = (\delta_1, \ldots, \delta_K)$.

You are an expert in physical reasoning. Your task is to predict how a scene changes during a physical phenomenon, as a sequence of states.
Inputs:
- Initial Image: **<INITIAL_FRAME>**
- Scene Graph: **<GRAPH_0>**
- User Prompt: **<TEXT_PROMPT>**

Step 1: Events. Break the phenomenon into a sequence of events in causal order. Objects that change at the same time belong to the same event.
Step 2: States and rules. For each event, list every object whose state changes, using the identifiers of the scene graph. For each object, give its new state in words and the physical rule that produces it, as a short qualitative statement without numbers. Include effects on objects that the prompt does not mention. If an object comes into existence, give it a new identifier and name the object it comes from. Objects you do not list stay unchanged.
Step 3: Timing. Give the fraction of the video that each event takes. The fractions must sum to less than 1, so that the final state remains visible at the end of the video.
Output Format & Example:
Return a structured JSON object exactly matching this schema:

```
{
  "deltas": [
    {"states": [
       {"object": "ice#1", "state": "partially melted",
        "rule": "ice above its melting point turns into water"},
       {"object": "tray#2", "state": "wet beneath ice#1",
        "rule": "liquid flows down onto its support"}],
     "fraction": 0.30},
    {"states": [
       {"object": "ice#1", "state": "melted into puddle#3 on tray#2",
        "rule": "ice above its melting point turns into water",
        "new_object": {"id": "puddle#3", "source": "ice#1"}}],
     "fraction": 0.30}
  ]
}
```

Prompt 3: Delta Translation (p_{edit}):
This instruction translates one delta into typed edits ϵ_i.

You are a precise graph editor. Your task is to express a change of state as edits to a scene graph.
Inputs:
- Current Scene Graph: **<GRAPH_i-1>**
- Physical Delta: **<DELTA_i>**
- Violations of the previous attempt (on retry only): **<VIOLATIONS>**

Step 1: Edits. Express the delta using only these operators: `Update(o, key, value)`, `Link(a, r, b)`, `Unlink(a, r, b)`, `Spawn(new_id <- source_id, category, attributes)`, and `Consume(o)`.
Step 2: Scope. Edit exactly the objects listed in the delta. Use only the attribute keys and relation types of the scene graph. If violations are given, correct them.
Output Format & Example:
Return a structured JSON object exactly matching this schema:

```
{
  "edits": [
    {"op": "Spawn", "id": "puddle#3", "source": "ice#1",
     "category": "puddle", "attributes": {"material_phase": "liquid", ...}},
    {"op": "Consume", "o": "ice#1"},
    {"op": "Link", "a": "tray#2", "r": "support", "b": "puddle#3"}
  ]
}
```

A.4 KEYFRAME RENDERING

For each event i, the net edits $\bar{\epsilon}_i$ describe how $\mathcal{G}_i$ differs from $\mathcal{G}_0$. They are translated into a natural-language editing instruction c_i by the same VLM with the prompt p_{render} below. The translation

replaces node identifiers with referring expressions that the image editor can resolve, using the category and, when a category occurs more than once, a relation from $\mathcal{G}_0$ (*e.g.*, "the ice cube on the tray"); it uses lineage to place new objects (*e.g.*, a puddle where the ice was); and it describes only the edited objects, ending with an explicit request to keep everything else unchanged. Since every keyframe is edited from I_0, the instruction describes the full change from the observed frame to state i, not the change from the previous state. For the final state of Table 8, c_3 is: "Replace the ice cube with a puddle of water that spreads over most of the tray, and make the tray wet around it. Keep everything else in the image unchanged."

Prompt 4: Keyframe Instruction (p_{render}):
This instruction translates the net edits $\bar{\epsilon}_i$ into the editing instruction c_i.

You are writing an instruction for an image editor. The editor receives the initial image and your instruction, and must change the image so that it shows a later state of the scene. The editor does not know the scene graph or the identifiers of objects.
Inputs:
- Initial Image: **<INITIAL_FRAME>**
- Initial Scene Graph: **<GRAPH_0>**
- Net Edits: **<NET_EDITS_i>**

Step 1: Refer to objects. Replace every identifier with a short description that picks out the object in the image, using its category and, if the category occurs more than once, its relation to another object (*e.g.*, "the ice cube on the tray").
Step 2: Describe the changes. Write one clause per edit, describing the state the object should be in, not the process that leads to it. For a new object, say where it appears, using the object it comes from (*e.g.*, "a puddle where the ice cube was"). For a removed object, say that it is gone, unless a new object replaces it. For a new or removed relation, describe the resulting arrangement (*e.g.*, "the puddle lies on the tray").
Step 3: Keep the rest. Do not mention objects that no edit names. End with "Keep everything else in the image unchanged."
Output Format & Example:
Return a structured JSON object exactly matching this schema:

```
{
  "instruction": "Replace the ice cube with a puddle of water that spreads over most
       of the tray, and make the tray wet around it. Keep everything else in the image
        unchanged."
}
```

The instruction c_i and the observed frame I_0 are passed to the image editor Ψ, which renders the keyframe I_i. Object masks are then extracted with Grounded-SAM-2, prompted with the category of each object in $\mathcal{G}_i$; when several instances are found, all of them are kept, since they define the keyframe instances $\{Q_l\}$. Depth maps are predicted with Depth Anything V2 and normalized to $[0, 1]$ per image before computing depth differences.

A.5 Guidance Details

Hyperparameters. GPSR uses Gemini 3 Flash with temperature 0 and JSON-constrained outputs, and keyframes are rendered with Gemini 3 Pro Image (Nano Banana Pro). The VDM is CogVideoX-I2V-5B with 50 denoising steps; videos have 49 frames at 720×480 and are generated with a fixed seed on one NVIDIA H100 GPU. Guidance follows the CogVideoX defaults of Frame Guidance: $t_E = 5$, $t_L = 20$, $N_g = 10$, and $\eta = 3.0$.

Latent slicing and flow matching. CogVideoX encodes the first frame separately and compresses the remaining frames by a factor of $r = 4$ in time, so frame f corresponds to latent index $j(f) = 0$ for $f = 0$ and $j(f) = 1 + \lfloor (f-1)/r \rfloor$ otherwise; the 49 frames map to 13 latents. Following Frame Guidance, the preview $\hat{x}_i$ is decoded from a window of three latents around $j(f_i)$, and the frame at f_i is taken from the decoded clip. CogVideoX is a v-prediction model, so the clean latent is estimated with Tweedie's formula as in the main paper. For a VDM trained with rectified flow, $z_t = (1-t)\, z_0 + t\, \varepsilon$, the estimate becomes $z_{0|t} = z_t - t\, v_\theta(z_t, t)$, and re-noising follows the flow-matching variant of the time-travel step in Frame Guidance.

Occupancy, instances, and regions. The encoder $\mathcal{F}$ is DINOv3, which produces a feature map at the stride of its patch size. The preview and the keyframe are resized to the same resolution before

encoding, and masks are resized to the feature grid by average pooling. Instances in the preview are the connected components of $\Phi_o(\hat{x}_i) > 0.5$. Centroids are computed in coordinates normalized to $[0, 1]$, so the matching cost and $\mathcal{L}_{\text{pos}}$ do not depend on resolution.

The region Ω_k of each term is computed at the image resolution as a binary map and average-pooled to the latent grid, which gives soft values at its boundary. For $\mathcal{L}_{\text{depth}}$ on a pair (a, b), Ω_k is the union of the regions of a and b. For $\mathcal{L}_{\text{count}}$ of an object whose instances are all matched, the region is still built from the matched pairs, so that the term keeps the object in place. The same spatial map is applied to every temporal index of the latent.

Guidance algorithm. Algorithm 1 summarizes one sampling run. It follows the schedule of Frame Guidance: in the first n_E denoising steps, while the layout forms, the latent is updated deterministically with the region-weighted gradient; in the following steps up to n_L, the unweighted gradient is applied with time-travel, *i.e.*, the guided latent is denoised by one step and re-noised back to the current noise level, with the number of repeats decreasing linearly; the remaining steps are unguided. Keyframes, masks, and depth maps are computed once before sampling.

Algorithm 1 Graph-Guided Test-Time Optimization (one sampling run)

Require: net edits and anchors $\{(\bar{e}_i, f_i)\}_{i=1}^{K}$; keyframes $\{I_i\}$ with masks and depth maps; timesteps $t_1 > \cdots > t_T$; stage boundaries $n_E < n_L$; repeats N_{g}; step size η; region weight λ

$z_{t_1} \sim \mathcal{N}(0, \mathbf{I})$
$\mathcal{J}_i \leftarrow$ latent window covering frame f_i, for $i = 1, \ldots, K$ ▷ latent slicing
for $n = 1, \ldots, T$ **do**
  $t \leftarrow t_n$
  if $n \le n_L$ **then** ▷ guided steps
    $M \leftarrow N_{\text{g}}$ **if** $n \le n_E$ **else** $\lceil N_{\text{g}}\,(n_L - n + 1)/(n_L - n_E) \rceil$ ▷ repeats decrease linearly
    for $m = 1, \ldots, M$ **do**
      $z_{0|t} \leftarrow \sqrt{\bar{\alpha}_t}\, z_t - \sqrt{1 - \bar{\alpha}_t}\, v_\theta(z_t, t)$ ▷ Eq. (3)
      $g \leftarrow 0$
      for $i = 1, \ldots, K$ **do**
        $\hat{x}_i \leftarrow \mathcal{D}\big(z_{0|t}[\mathcal{J}_i]\big)$ ▷ preview at anchor f_i
        compute $\Phi_o(\hat{x}_i)$, instances $\{P_j\}$, and matching π^* for every object o
        for each term $\mathcal{L}_k^{(i)}$ of Eq. (8) **do**
          $\hat{g}_k \leftarrow \nabla_{z_t} \mathcal{L}_k^{(i)} \,/\, \max\big(\|\nabla_{z_t} \mathcal{L}_k^{(i)}\|, \varepsilon\big)$ ▷ normalized gradient
          if $n \le n_E$ **then**
            $g \leftarrow g + \big(\Omega_k + \lambda(1 - \Omega_k)\big) \odot \hat{g}_k$ ▷ region-weighted
          else
            $g \leftarrow g + \hat{g}_k$
          end if
        end for
      end for
      if $n \le n_E$ **then** ▷ layout stage: deterministic update
        $z_t \leftarrow z_t - \eta\, g$
      else ▷ detail stage: time-travel
        $z_{t_{n+1}} \leftarrow \mathrm{DDIM}(z_t - \eta\, g,\ t \to t_{n+1})$
        $z_t \leftarrow \sqrt{\bar{\alpha}_t / \bar{\alpha}_{t_{n+1}}}\, z_{t_{n+1}} + \sqrt{1 - \bar{\alpha}_t / \bar{\alpha}_{t_{n+1}}}\, \xi, \quad \xi \sim \mathcal{N}(0, \mathbf{I})$ ▷ re-noise
      end if
    end for
  end if
  $z_{t_{n+1}} \leftarrow \mathrm{DDIM}(z_t,\ t \to t_{n+1})$ ▷ reverse step
end for
return $\mathcal{D}(z_{t_{T+1}})$ ▷ $t_{T+1} = 0$

A.6 Formulation of the Guidance Loss

Each term of Section 3.2 compares an object-level summary of the preview $\hat{x}_i$ with the same summary of the keyframe I_i, so it is zero when the selected property matches, whatever contour or texture the

editor drew. Every term reaches the latent through the occupancy or the features of the preview, so we analyze its gradient at a grid location u; the chain rule carries it to z_t. We write $\Phi(u) = \Phi_o(\hat{x}_i)(u)$ and $\Phi^*(u) = \Phi_o(I_i)(u)$.

Soft occupancy. With $s(u) = \cos(\mathcal{F}(\hat{x}_i)(u), \rho_o)$,

$$\Phi(u) = \sigma\Big(\frac{s(u) - b}{\tau}\Big), \qquad \frac{\partial \Phi(u)}{\partial s(u)} = \frac{\Phi(u)\big(1 - \Phi(u)\big)}{\tau} \le \frac{1}{4\tau}. \tag{11}$$

The gradient peaks where $s(u) = b$, *i.e.*, at the boundary of the object, and vanishes inside it and in the background, so all terms below change the object where it can grow, shrink, or move. A small τ sharpens the mask but saturates the sigmoid.

Count term. For an unmatched preview instance P_j and an unmatched keyframe instance Q_l, with $r_l = \operatorname{avg}_{Q_l} \Phi / \operatorname{avg}_{Q_l} \Phi^*$,

$$\frac{\partial}{\partial \Phi(u)} \operatorname{avg}_{P_j} \Phi = \frac{1}{|P_j|} > 0, \qquad \frac{\partial}{\partial \Phi(u)} \max(0, 1 - r_l) = -\frac{\mathbf{1}[r_l < 1]}{|Q_l| \operatorname{avg}_{Q_l} \Phi^*} \le 0. \tag{12}$$

Descent lowers the occupancy of an extra copy uniformly until it disappears, and raises it where a missing copy should be. The fill target is the keyframe's own occupancy, which a correct object reaches, not one, and the hinge stops the term once it is reached.

Appearance term. With the mean features $p = \bar{\mathcal{F}}_{P_j}(\hat{x}_i)$ and $q = \bar{\mathcal{F}}_{Q_l}(I_i)$,

$$\mathcal{L}_{\text{app}} = 1 - \hat{p}^\top \hat{q}, \qquad \frac{\partial \mathcal{L}_{\text{app}}}{\partial \mathcal{F}(\hat{x}_i)(u)} = -\frac{\hat{q} - (\hat{p}^\top \hat{q})\,\hat{p}}{|P_j|\,\|p\|} \quad (u \in P_j). \tag{13}$$

The gradient is orthogonal to $\hat{p}$, so it rotates the appearance toward the keyframe without changing the feature norm, and it is identical at every $u \in P_j$, so it changes the object as a whole, independently of its shape.

Extent term. With the soft areas $A = \sum_{R_{jl}} \Phi$ and $A^* = \sum_{R_{jl}} \Phi^*$,

$$\begin{aligned} \mathcal{L}_{\text{ext}} &= \log^2 \frac{A}{A^*}, & \frac{\partial \mathcal{L}_{\text{ext}}}{\partial \Phi(u)} &= \frac{2}{A} \log \frac{A}{A^*}, \\ \mathcal{L}_{\text{ext}}(\kappa A^*, A^*) &= \mathcal{L}_{\text{ext}}(A^*/\kappa, A^*), & \mathcal{L}_{\text{ext}}(cA, cA^*) &= \mathcal{L}_{\text{ext}}(A, A^*). \end{aligned} \tag{14}$$

The sign says whether to grow or shrink; growth and shrinkage by the same factor cost the same, and small and large objects are treated alike. Summing over $R_{jl} = P_j \cup Q_l$ gives a gradient where the object should grow, which P_j alone would not.

Position term. With the centroid $c = \sum_{R_{jl}} u\,\Phi(u)/A$ and its keyframe counterpart c^*,

$$\frac{\partial c}{\partial \Phi(u)} = \frac{u - c}{A}, \qquad \frac{\partial \|c - c^*\|^2}{\partial \Phi(u)} = \frac{2}{A}\,(c - c^*)^\top (u - c). \tag{15}$$

The gradient is negative on the side of the object facing c^* and positive on the other side, so descent moves the object toward the target instead of deforming it. The region must therefore contain the target, which R_{jl} and the hull in Ω_k ensure. Matching the arrangement of the keyframe realizes a relation that has no differentiable predicate.

Depth term. With the occupancy-weighted depth $D_a = \sum_u \Phi_a(u)\,z(u) / \sum_u \Phi_a(u)$ and $\Delta z_{ab} = D_a - D_b$,

$$z \mapsto \alpha z + \beta \implies D_a \mapsto \alpha D_a + \beta, \qquad \Delta z_{ab} \mapsto \alpha\,\Delta z_{ab}. \tag{16}$$

The gap is free of the unknown offset of monocular depth, and per-image normalization to $[0, 1]$ fixes α, so the term compares only which object is in front and by how much.

Combination. The terms have different ranges ($\mathcal{L}_{\text{app}} \in [0, 2]$, while $\mathcal{L}_{\text{ext}}$ is unbounded). The update uses

$$\hat{g}_k = \frac{\nabla_{z_t} \mathcal{L}_k}{\|\nabla_{z_t} \mathcal{L}_k\|}, \qquad \hat{g}_k(c\,\mathcal{L}_k) = \hat{g}_k(\mathcal{L}_k) \quad \forall c > 0, \tag{17}$$

so every term contributes a unit direction regardless of its scale, and no weights are needed. A small constant is added to all denominators, and terms with empty instance sets are skipped.

A.7 Example Event Chain

Table 8 shows the complete event chain for "an ice cube melting in the sun" on a tray, for $F = 49$ frames. At the final state, the net edits are Consume(ice#1), Spawn(puddle#3 $\leftarrow$ ice#1), Update(puddle#3, extent, spread over most of the tray), Link(tray#2, support, puddle#3), and Update(tray#2, surface, wet). The table on which the tray stands (table#4) is never named by a delta and stays unchanged throughout.

Table 8: Event chain for "an ice cube melting in the sun" ($F = 49$).

i	**State** s_o	**Physical rule** r_o	**Edits** ϵ_i	d_i	f_i
1	ice#1: partially melted; tray#2: wet beneath ice#1	ice above its melting point turns into water; liquid flows down onto its support	Update(ice#1, material phase, partially melted), Update(tray#2, surface, wet)	0.30	14
2	ice#1: melted into puddle#3 on tray#2	ice above its melting point turns into water	Spawn(puddle#3 $\leftarrow$ ice#1), Consume(ice#1), Link(tray#2, support, puddle#3)	0.30	29
3	puddle#3: spread over most of tray#2	liquid spreads over a flat support	Update(puddle#3, extent, spread over most of the tray)	0.25	41

B Additional Quantitative Results

Table 9 reports the quality dimensions of VBench, including closed-source VDMs.

C Failure Analysis

We analyze where PhysPlan fails on Physics-IQ. A video counts as failed if its score is in the lowest third of all PhysPlan videos, which gives 66 failed videos. Two authors inspected each of them, together with its event chain and keyframes, and attributed it to the first stage of the pipeline that goes wrong:

- **Reasoning**: the deltas are physically wrong or incomplete, *e.g.*, a missing consequence on a passive object or a wrong order of events.
- **Edits**: the edit set is rejected by the checks after all retries, or passes them but misrepresents the delta.
- **Keyframe**: the image editor does not realize an edit, or changes objects that no edit names.
- **Localization**: masks, occupancy, or depth are wrong, *e.g.*, for transparent, reflective, thin, or fragmented objects.
- **Guidance**: the event chain and keyframes are correct, but the video does not follow them, typically for continuous motion between anchors.

Disagreements between the two authors were resolved by discussion.

Discussion. Most failures happen after the plan is made (Table 10), and the dominant cause depends on the category. Guidance dominates only Fluid Dynamics (40%): the event chain fixes the state of the scene only at the anchors, and continuous flow between them is left to the VDM, which is consistent with the gap to VLIPP in this category. Localization dominates Optics (52%) and Magnetism (35%), whose scenes often contain transparent, reflective, or small metallic objects, for which feature similarity and monocular depth are least reliable. Keyframe failures dominate Solid Mechanics and Thermodynamics (30% each): in Solid Mechanics, the editor often draws an implausible pose or arrangement after a fall or a collision, and in Thermodynamics, it tends to miss subtle changes such as a slight change of color or phase. Overall, localization and guidance each account for 30% of the failures and keyframes for 25%. Reasoning errors are less frequent (11%) and concentrate in Thermodynamics and Magnetism, whose phenomena are less common and whose effects on passive objects are harder to predict. Failures of the edits are rare (4%): as Table 11 shows, 87% of the edit sets pass the checks at the first attempt and 97% after retries. Coverage is the most frequently violated check, *i.e.*, the VLM most often edits an object that the delta does not name or forgets one that it does; this is exactly the error that the checks prevent from reaching the state graph.

Table 9: Quality dimensions of VBench (Huang et al., 2024). QS: Quality Score; SC: Subject Consistency; BC: Background Consistency; TF: Temporal Flickering; MS: Motion Smoothness; DD: Dynamic Degree; AQ: Aesthetic Quality; IQ: Imaging Quality. Best results are in **bold**.

Method	QS↑	SC↑	BC↑	TF↑	MS↑	DD↑	AQ↑	IQ↑
Closed-source VDMs								
Runway Gen-3	84.11	97.10	96.62	98.61	99.23	60.14	63.34	**66.82**
Kling	83.39	**98.33**	**97.60**	99.30	99.40	46.94	61.21	65.62
Pika	82.92	96.94	97.36	**99.74**	**99.50**	47.50	62.04	61.87
Luma	83.47	97.33	97.43	98.64	99.35	44.26	**65.51**	66.55
Open-source								
CogVideoX-I2V-5B	83.05	96.45	96.71	98.97	97.20	69.51	61.88	63.33
PhysPlan (Ours)	**84.88**	97.06	97.10	98.72	98.80	**75.62**	62.20	65.78

Table 10: Failure attribution on Physics-IQ: share of failed videos (%) per pipeline stage and category. Each row sums to 100.

Category	Reasoning	Edits	Keyframe	Localization	Guidance
Solid Mechanics	12	4	**30**	26	28
Fluid Dynamics	8	3	20	29	**40**
Optics	10	2	14	**52**	22
Magnetism	18	5	12	**35**	30
Thermodynamics	21	6	**30**	18	25
All	11	4	25	**30**	**30**

Table 11: Deterministic checks over all edit sets: share that pass at the first attempt, after retries, or not at all, and share of each check among all violations.

Statistic	Value (%)
Pass at first attempt	87
Pass after retries	97
Rejected after all retries	3
Violations: coverage	46
Violations: grounding	31
Violations: consistency	15
Violations: lineage	8

D Cost Analysis

Table 12 breaks down the average runtime of PhysPlan per video, Table 13 estimates its API cost, and Table 14 compares its total cost with the base model and Frame Guidance. All measurements use a single NVIDIA H100 (80GB) GPU, CogVideoX-I2V-5B, and 720×480 videos of 49 frames, with $K = 3$ events unless stated otherwise.

API cost. We estimate the API cost from the average number of tokens per call and the standard list prices of the Gemini API at the time of writing: \$0.50 and \$3.00 per million input and output tokens for Gemini 3 Flash, whose output tokens include its thinking tokens, and \$0.134 per output image at 1K resolution for Gemini 3 Pro Image. Delta translation includes the retries triggered by the checks. Table 13 lists the estimate for $K = 3$.

Discussion. Reasoning about the scene is cheap in time: all GPSR steps, including keyframe rendering and grounding, take 27.5 s, or 7.5% of the total (Table 12), and run once per video. Its API cost is about \$0.45 per video (Table 13), of which over 90% comes from rendering the keyframes; all VLM calls together cost about \$0.04. Evaluating a full benchmark of 200 videos thus costs about \$90 in API calls, and the Batch API of Gemini, which halves all prices, would reduce this further. The runtime is dominated by test-time optimization, in particular by backpropagating through the denoiser at every guided step, which is shared with Frame Guidance. Compared with Frame

Table 12: Runtime per stage of PhysPlan (seconds per video, $K = 3$).

Stage	Model	Time (s)
Grounded Physical State Reasoning		
Scene parsing and decomposition	Gemini 3 Flash	6.8
Delta translation and checks	Gemini 3 Flash	4.2
Keyframe rendering	Gemini 3 Pro Image	12.5
Masks and depth	Grounded-SAM-2, Depth Anything V2	4.0
Graph-Guided Test-Time Optimization		
Preview decoding	CogVideoX VAE	24.5
Measurement and backpropagation	DINOv3, CogVideoX	214.0
Sampling and final decoding	CogVideoX	100.0
Total		366.0

Table 13: Estimated API cost of PhysPlan per video ($K = 3$), at standard Gemini API list prices.

Call	Calls	Tokens (in / out)	Cost (USD)
Scene parsing (p_{parse})	1	2.5k / 2.5k	0.009
Phenomenon decomposition (p_{delta})	1	3.5k / 3.0k	0.011
Delta translation (p_{edit})	3.4	7.5k / 2.7k	0.012
Keyframe instruction (p_{render})	3	7.8k / 1.2k	0.008
Keyframe rendering (Gemini 3 Pro Image)	3	3 images	0.405
Total			0.445

Table 14: Total cost per video, and its growth with the number of events K. Frame Guidance takes keyframes as input and therefore has no API cost of its own.

Method	Time (s)	Peak memory (GB)	API cost (USD)
CogVideoX-I2V-5B	100	26	—
Frame Guidance	310	58	—
PhysPlan	366	64	0.45
PhysPlan, $K = 3$	366	64	0.45
PhysPlan, $K = 5$	471	70	0.73
PhysPlan, $K = 7$	575	76	1.01

Guidance, PhysPlan adds 56 s (18%) and 6 GB of memory (Table 14), mainly for the feature encoder, the instance matching, and the additional terms. Runtime and API cost both grow roughly linearly with the number of events, by about 52 s and $0.14 per event, since every event adds one keyframe to render and one anchor whose preview is decoded and whose terms enter the objective. Compared with training-based methods, which require fine-tuning the VDM on curated data, this cost is paid only at inference and requires no training.

E User Study

Protocol. We conduct a two-alternative forced-choice (2AFC) study with the web interface shown in Figure 5. In each trial, a participant sees two videos, labeled Video A and Video B, generated from the same prompt and the same initial frame: one by PhysPlan and one by a baseline, either CogVideoX-I2V-5B or Frame Guidance. The two videos are shown side by side and loop in sync, so that the same moment of the phenomenon is visible in both at the same time. For each of three criteria, the participant chooses the better video:

- **Physical plausibility**: which video follows the described phenomenon in a physically plausible way, including its effects on other objects in the scene?
- **Frame quality**: which video has sharper, cleaner, and more realistic frames?
- **Temporal smoothness**: which video changes more smoothly over time, without flickering, jumps, or drifting background?

A choice is required for every criterion before the participant can move to the next trial, so there are no ties. The left or right position of each method is randomized in every trial, and participants are not told which methods are compared or which video comes from which method. Before the study, participants see a short description of the three criteria with one example trial, which is not counted.

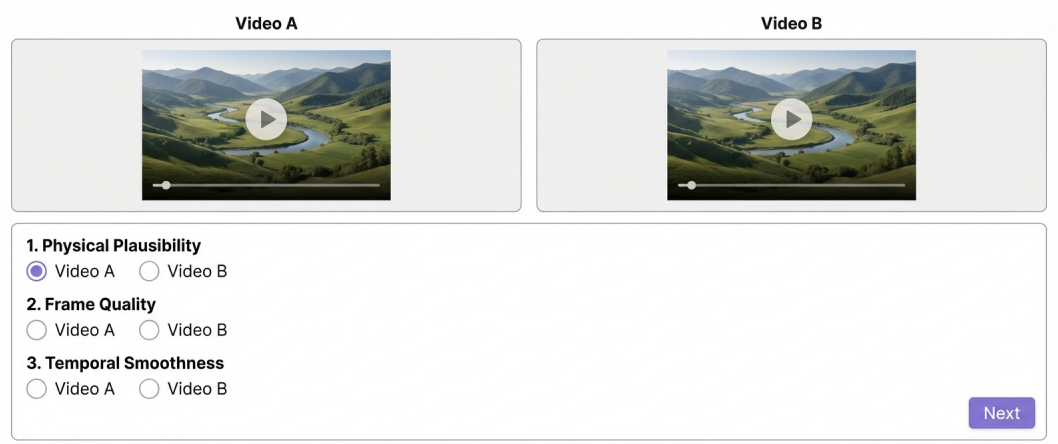


Figure 5: User study interface. Each trial shows two videos from the same prompt and initial frame, one by PhysPlan and one by a baseline, in random order. The participant chooses the better video for each of the three criteria and continues with *Next*.

Prompts and workload. We sample 40 prompts, 20 from PhyGenBench and 20 from Physics-IQ, covering all their domains, and generate one pair per prompt and baseline, which gives 80 pairs. Each participant rates 40 randomly assigned pairs, so that every pair is rated by 30 participants on average and the study collects 2,400 ratings per criterion. Table 15 summarizes the setup.

Table 15: Setup of the user study.

Item	Value
Participants	60
Prompts (PhyGenBench / Physics-IQ)	20 / 20
Baselines	CogVideoX-I2V-5B, Frame Guidance
Video pairs	80
Pairs rated per participant	40
Ratings per pair (average)	30
Ratings per criterion (total)	2,400

Preference rate. For a criterion c and a baseline B, let $V_c(B)$ be the set of all votes cast on pairs that compare PhysPlan with B. The preference rate for PhysPlan is the share of these votes that choose PhysPlan,

$$\mathrm{Pref}_c(B) = \frac{\left|\{v \in V_c(B) : v = \text{PhysPlan}\}\right|}{|V_c(B)|}. \tag{18}$$

Since every vote chooses one of the two videos, the preference rate for the baseline is $1 - \mathrm{Pref}_c(B)$, and 50% means no preference. The pooled rate reported in the main paper uses the votes against both baselines together; since both baselines receive the same number of votes, it is the mean of the two per-baseline rates. Table 16 reports the rates.

Table 16: Preference rates (%) for PhysPlan, per baseline and pooled.

Against	Physical plausibility	Frame quality	Temporal smoothness
CogVideoX-I2V-5B	76	63	77
Frame Guidance	68	57	69
Pooled	72	60	73

PhysPlan is preferred against both baselines on all criteria. The margin is smaller against Frame Guidance, which uses the same keyframes, and smallest for frame quality, where Frame Guidance is competitive, consistent with its similar FID on PhyGenBench (46.4 vs. 45.4). The margins in physical plausibility and temporal smoothness remain clear, matching the gains on the benchmarks and in FVD.

Participants. The study involved 60 adult volunteers, aged 19 to 38 (median 24), recruited from our university through mailing lists and student groups. About two thirds (38) have a background in computer science or engineering, 12 in the natural sciences, and 10 in other fields such as design and the humanities. Most participants are familiar with AI-generated video but do not work with it: 21 see generated videos regularly, 27 occasionally, and 12 rarely or never. No participant was involved in this work. Participation was voluntary and anonymous: all participants gave informed consent, no personally identifiable information was collected, and the study was granted an exemption by the institutional review board.