# FRACGEN: LEARNING HOW OBJECTS STRETCH AND TEAR WITH PHYSICS-INFORMED VIDEO GENERATION

**Anonymous authors**
Paper under double-blind review

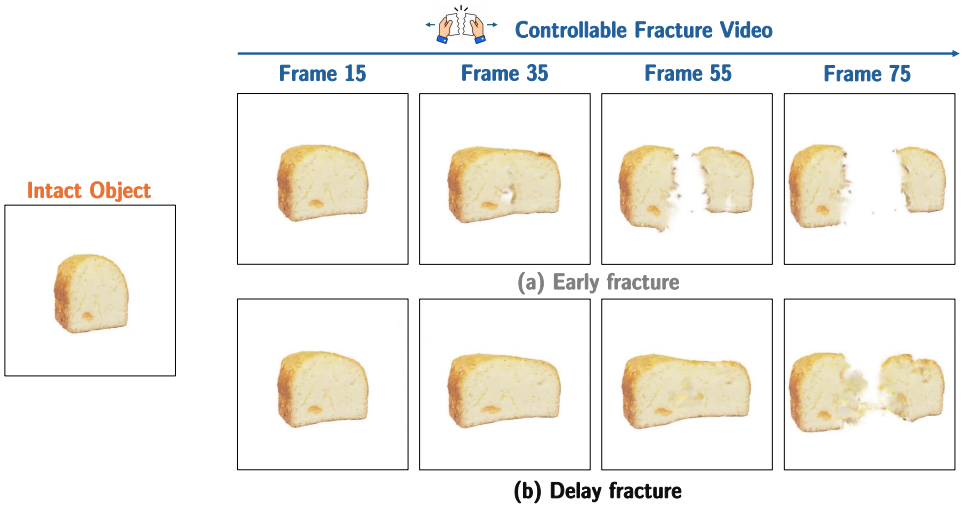



Figure 1: **Controllable stretching and tearing with FracGen**. Given a single object image and specified force, material, and fracture conditions, FracGen generates a video of the object deforming and tearing. Under the same applied force, varying the fracture properties produces earlier tearing (top) or delayed tearing after greater deformation (bottom). Results are better viewed in the video on our supplement website: https://iclr11751.github.io/.

## ABSTRACT

We introduce **FracGen**, a fracture-aware video generation model that produces plausible, controllable fracture dynamics from a single image of an intact object, conditioned on physics signals. To train FracGen, we build **FracSim**, a fracture-aware simulation framework that augments material point method (MPM) simulation with a continuum damage model, producing paired fracture videos and dense, pixel-aligned physical fields at no additional cost beyond standard rendering. FracGen leverages these maps in two ways: it is trained to jointly predict them alongside RGB video, encouraging the model to capture physical state rather than surface appearance; and it is supervised with physics-informed losses that encourage consistency among the predicted maps. As a result, FracGen captures distinct material-specific fracture behavior without expensive test-time simulation or per-scene tuning, while offering fine-grained control over where an object tears, how fast the crack propagates, and how much deformation precedes failure. We further introduce a benchmark for evaluating the physical plausibility of generated fracture video, and show through extensive experiments that FracGen outperforms existing video generation baselines in both physical and visual fidelity.

## 1 INTRODUCTION

Things break all the time. We drop, tear or snap objects and know what a fracture or damage looks like. Bread breaking like glass or a mug tearing apart like paper looks immediately wrong. Any convincing fracture-generation algorithm must match these physical intuitions — not just the final broken shape, but also how an object deforms, when it starts to fail and how the fracture develops.

A growing body of work shows that modern generative models internalize substantial knowledge of physical dynamics directly from data (Bhattad et al., 2023; Du et al., 2023; Liu et al., 2024b; Wang



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.

et al., 2025; Xing et al., 2025; Gillman et al., 2025). In this paper, we ask whether this extends to fracture: can a video generator learn how objects break from limited data and lightweight fine-tuning?

Recent work has begun to condition video generation on physical signals. ForcePrompting (Gillman et al., 2025) shows that generators can respond to push/pull forces, but does not explicitly model damage accumulation or fracture. Physics-based approaches address different parts of this problem. PhysGaussian (Xie et al., 2024) produces physically grounded deformation but lacks a damage model for fracture, while Fracture-GS (Wang et al., 2026) introduces a Collision Material Point Method (Collision-MPM) to simulate fracture under extreme mechanical collisions. These simulation pipelines require a reconstructed scene and configured physical conditions. Our goal is to instead start from a single photograph and specified material and force conditions, and answer by generating *"what happens if I pull here until it breaks?"*

To this end, we introduce **FracGen**, a video generation model that produces fracture dynamics for objects under tensile stretch. We focus on *stretch-to-tear* fracture, where substantial deformation precedes failure, making the evolution before fracture an important part of the generation task. FracGen captures distinct material-specific behaviors, from objects that fail after little stretching to those that undergo large deformation before tearing apart.

Teaching a generator these behaviors requires supervision beyond visual appearance: RGB frames alone do not explicitly describe the mechanical state that leads to fracture. We observe that simulators like PhysGaussian already track rich physical information that is not exposed in their rendered videos. Each particle carries a deformation gradient, a $3 \times 3$ matrix encoding local deformation, from which strain can be computed and stress obtained through the material's constitutive law. Particle velocities further describe the local motion. We introduce a damage field that tracks the accumulated failure state of each particle and governs fracture. Rendering these quantities as pixel-aligned maps alongside RGB frames yields dense, physically grounded supervision without additional simulation runs. We call this augmented simulation framework **FracSim**, and its rendered physical fields **FracPhys Maps**.

We use these maps in two complementary ways, forming our main technical contributions. First, we extend the video generator to jointly predict FracPhys Maps alongside the fracture RGB video, forcing the model to encode the underlying mechanical state rather than surface appearance alone. These maps also enable us to inspect the physical state underlying the generated video. Second, we introduce physics-informed losses, derived from constitutive modeling, that encourage consistency among the predicted physical channels: the predicted strain, stress, and damage representations are coupled through a latent consistency loss inspired by constitutive modeling and stress degradation. We further introduce a benchmark targeting the evaluation of physical alignment in generated fracture videos, along with extensive experiments comparing FracGen against baseline methods to quantify its effectiveness. In summary, our contributions include:

1. **FracSim**, a fracture-aware simulation framework that incorporates continuum damage mechanics into MPM, producing paired fracture videos and FracPhys Maps for training.
2. **FracGen**, a video generation model that jointly predicts fracture RGB video and physical fields, trained with physics-informed losses encouraging consistency between physics maps.
3. A benchmark with new evaluation metrics for evaluating fracture video generation across diverse materials, applied forces, and fracture behaviors.
4. Extensive experiments showing FracGen generates physically plausible, controllable fracture that generalizes across unseen objects, outperforming existing baselines both visually and physically.

## 2 Related Work

**Physics-informed 3D Simulation.** A recent line of work grounds 3D dynamics in continuum mechanics. PhysGaussian (Xie et al., 2024) couples the Material Point Method (MPM) (Hu et al., 2018) with 3D Gaussians, treating each Gaussian as both a rendering primitive and physical particle, while Gaussian Splashing (Feng et al., 2025) adopts Position-Based Dynamics (Macklin et al., 2016) for cohesive solid-fluid simulation. However, these solvers require manual scene specification and tedious parameter tuning. To tackle this, PhysDreamer (Zhang et al., 2024) optimizes MPM material parameters against generative reference videos using differentiable simulation, whereas DreamPhysics (Huang et al., 2025) optimizes directly via Score Distillation Sampling (SDS) (Poole et al., 2023). Physics3D (Liu et al., 2024a) and OmniPhysGS (Lin et al., 2025) further expand

SDS-driven optimization to richer constitutive models, incorporating viscoelastic damping or multi-material mixtures (elastic, plastic, fluid) within a single scene.

**Physics-informed Video Generation.** With advances in video generation models, recent works seek to inject faithful physical realism through two main strategies. The first conditions generation on explicit control signals: Force Prompting (Gillman et al., 2025) fine-tunes a video diffusion model on synthetic force-annotated data, PhysCtrl (Wang et al., 2025) controls video diffusion using predicted 3D point-trajectories conditioned on forces and material traits, and PhyCo (Narayanan et al., 2026) enables pixel-aligned ControlNet guidance over physical fields (friction, deformation, force) refined by VLM reward optimization. The second strategy enforces physics via high-level reasoning and reward alignment: DiffPhy (Zhang et al., 2025) uses LLM context-builder alongside a MLLM physical critic, while PhyGDPO (Cai et al., 2025) post-trains a video models using physics-aware groupwise preference optimization. However, these methods focus on rigid-body motion or simple deformation, lacking mechanisms for material failure. To the best of our knowledge, FracGen is the first model to generate complex fracture dynamics across the full deformation-to-fracture progression.

## 3 Background

**Gaussian–MPM Simulation.** 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents a scene using anisotropic Gaussian kernels that carry geometry and appearance. PhysGaussian (Xie et al., 2024) couples this representation with the Material Point Method (MPM) (Hu et al., 2018), associating particles with evolving positions $x_p^t$, velocities $v_p^t$, and deformation gradients $\boldsymbol{F}_p^t$. MPM updates these states through transfers between particles and a background grid, while the Gaussians deform with the material to render its evolving appearance. The deformation map $\phi$ transports a material point from its reference position $x_p$ to $x_p^t = \phi(x_p, t)$. Its gradient, $\boldsymbol{F}_p^t = \nabla_{x_p}\phi(x_p, t)$, encodes local stretching, compression, rotation, and shearing. The polar decomposition $\boldsymbol{F}_p = \boldsymbol{R}_p\boldsymbol{S}_p$ separates rigid rotation $\boldsymbol{R}_p$ from stretch $\boldsymbol{S}_p$, allowing us to distinguish shape change from rigid motion.

**Constitutive Models.** Strain measures local deformation relative to the reference configuration, while stress describes the internal forces per unit area transmitted through the material. A constitutive model relates deformation to stress. For hyperelastic materials, a strain energy density $\psi(\boldsymbol{F})$ determines the stress response, which can be nonlinear while remaining reversible. In isotropic linear elasticity, Hooke's law gives $\boldsymbol{\sigma} = \lambda_{\mathrm{L}}\,\mathrm{tr}(\boldsymbol{\varepsilon})\mathbf{I} + 2\mu\boldsymbol{\varepsilon}$, where $\lambda_{\mathrm{L}}$ and $\mu$ are the Lamé parameters. FracSim computes stress from the material's strain energy and introduces damage-dependent degradation, while our generator uses the linear-elastic relationship to motivate an approximate consistency loss in latent space.

**Particle State Update with MPM.** For particle $p$ at simulation time $t$, let $x_p^t$, $v_p^t$, and $\boldsymbol{F}_p^t$ denote its position, velocity, and deformation gradient, respectively. At each step, MPM transfers particle quantities to a background Eulerian grid, solves the momentum equation, and transfers the results back to update these states. Since each particle carries both appearance and physical state, a single representation supports simulation and rendering. Under a first-order approximation of $\phi$, each Gaussian's center moves from its reference position $x_p$ to $x_p^t$, and its reference covariance $A_p$ becomes $A_p^t = \boldsymbol{F}_p^t A_p (\boldsymbol{F}_p^t)^\top$. Its spherical harmonic (SH) appearance rotates with the particle rotation $\boldsymbol{R}_p$, while its opacity $o_p$ remains fixed.

**Conditional Video Generation** We build on a latent video generator (Wan et al., 2025) that encodes an RGB video $\mathbf{V}$ as $z_1 = \mathcal{E}(\mathbf{V})$ using a 3D VAE. Under flow matching (Lipman et al., 2023), a transformer $v_\theta$ learns to transport Gaussian noise $z_0 \sim \mathcal{N}(0, \mathbf{I})$ to video latents along the linear path $z_t = tz_1 + (1-t)z_0$. The training objective is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{t, z_0, z_1, c}\left[\left|v_\theta(z_t, t, c) - (z_1 - z_0)\right|^2\right], \tag{1}$$

where $t \sim \mathcal{U}(0, 1)$ and $c$ denotes conditioning inputs. We extend this formulation to jointly generate fracture videos and physical maps under specified material and force conditions.

## 4 FracSim and FracGen

We discuss in §4.1 how we design **FracSim**, which outputs dynamic fracture physics maps (FracPhys) alongside fracture videos, forming corpus for training **FracGen**, as detailed in §4.2.

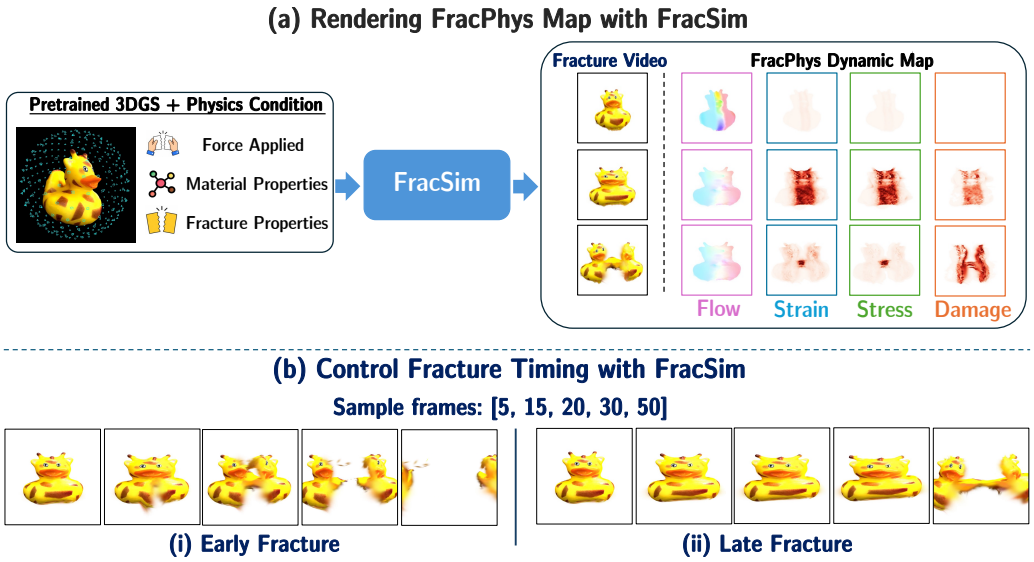



Figure 2: FracSim generates physically plausible fracture dynamics spanning deformation to structural failure. (a) Alongside RGB video, FracSim outputs FracPhys map. (b) Our damage model enables explicit control over fracture timing under fixed forces and material properties.

### 4.1 FracSim: Simulating Fracture Video and Rendering FracPhys Maps

We integrate a continuum damage model into MPM to generate dynamic physics maps alongside RGB frames. We refer to these maps as FracPhys maps and discuss how to extract them below.

**Flow.** Tensile load is prescribed via particle velocities (referred as flow), making $\boldsymbol{v}_p \in \mathbb{R}^3$ the most direct kinematic signal in the simulation. Rather than estimating this field from rendered frame via optical flow, we directly extract it from the MPM state to preserve exact direction and magnitude.

**Strain.** We require a strain measure that vanishes under rigid rotation yet remains meaningful at large deformation. The Green-Lagrange strain tensor satisfies both requirements:

$$\boldsymbol{\varepsilon}_p = \tfrac{1}{2}\left(\boldsymbol{F}_p^\top \boldsymbol{F}_p - \boldsymbol{I}\right) = \tfrac{1}{2}\left(\boldsymbol{S}_p^\top \boldsymbol{S}_p - \boldsymbol{I}\right), \tag{2}$$

where the polar decomposition $\boldsymbol{F}_p = \boldsymbol{R}_p \boldsymbol{S}_p$ isolates stretch $\boldsymbol{S}_p$ from rotation $\boldsymbol{R}_p$, which cancels exactly. The result is rotation-invariant, zero at rest, and nonzero only when genuine deformation occurs. We record its Frobenius norm $\|\boldsymbol{\varepsilon}_p\|_F$ as a scalar per particle.

**Stress.** We compute the Kirchhoff stress tensor from the strain energy density $\psi(\boldsymbol{F}_p)$:

$$\boldsymbol{\tau}_p = \frac{\partial \psi}{\partial \boldsymbol{F}_p} \boldsymbol{F}_p^\top. \tag{3}$$

Damage degrades this tensor before it is used in the MPM momentum update. We then convert the degraded tensor to Cauchy stress and extract its von Mises magnitude for rendering, as described below.

**Damage.** We augment PhysGaussian with a scalar damage variable $d_p \in [0, 1]$, where 0 denotes intact material and 1 denotes complete failure. To represent this degradation at the particle level, we use a kinematic proxy based on the maximum principal stretch $\lambda_{\max}(\boldsymbol{F}_p)$, the largest singular value of $\boldsymbol{F}_p$, following (Patnaik & Semperlotti, 2021):

$$d^t_{p,\text{raw}} = \text{clamp}\left(\frac{\lambda_{\max}(\boldsymbol{F}^t_p) - \lambda_{\text{onset}}}{\lambda_{\text{crit}} - \lambda_{\text{onset}}}, 0, 1\right), \qquad d^t_p = \max\left(d^{t-1}_p, d^t_{p,\text{raw}}\right). \tag{4}$$

We initialize $d^0_p = 0$ and require $\lambda_{\text{crit}} > \lambda_{\text{onset}} \geq 1$. Under increasing stretch, damage begins at $\lambda_{\text{onset}}$ and grows linearly to complete failure at $\lambda_{\text{crit}}$. The running maximum preserves accumulated damage during unloading, enforcing irreversibility. Adjusting these thresholds controls fracture timing under fixed loading and material parameters (Fig. 2b).

Damage feeds back into the simulation by degrading the stress tensor:

$$\tilde{\boldsymbol{\tau}}_p = (1 - d_p)\boldsymbol{\tau}_p. \tag{5}$$

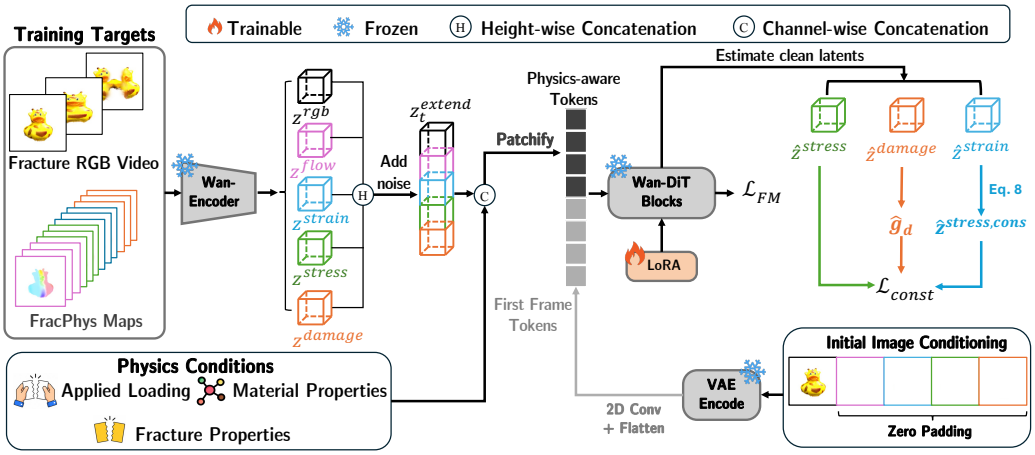



Figure 3: **FracGen overview.** RGB fracture videos and FracPhys Maps are encoded by a frozen VAE and concatenated along the spatial height dimension for joint generation. Applied loading, material properties and fracture properties provide channel-wise conditioning. The initial object image supplies reference tokens. We adapt the transformer with LoRA using flow-matching supervision and a physics-informed latent consistency regularizer coupling strain, stress, and damage representations.

This degraded tensor is used in the MPM momentum update. As $d_p$ approaches one, the particle progressively loses its ability to sustain load, allowing separation. For the stress map, we convert the degraded Kirchhoff stress to Cauchy stress, $\tilde{\boldsymbol{\sigma}}_p = \tilde{\boldsymbol{\tau}}_p / J_p$, where $J_p = \det(\boldsymbol{F}_p)$, and compute its equivalently degraded von Mises scalar $\tilde{\boldsymbol{\sigma}}_{\mathrm{vM},p}$. Damage is the sole softening mechanism; we do not model plastic flow (more discussion in §A.4).

**Rendering FracPhys Maps.** As discussed in §3, 3DGS composites pixel color using appearance term $\mathrm{SH}(l_p; \mathcal{C}_p)$. We replace this with any per-particle quantity $q_p$:

$$M_q = \sum_{p \in \mathcal{P}} \alpha_p \, q_p \prod_{j=1}^{p-1} (1 - \alpha_j). \tag{6}$$

where $q_p = \tilde{\boldsymbol{\sigma}}_{\mathrm{vM},p}$ for stress, $\|\boldsymbol{\varepsilon}_p\|_F$ for strain, $\boldsymbol{d}_p$ for damage, or $\boldsymbol{v}_p \in \mathbb{R}^3$ for velocity. For vector-valued $q_p$, compositing applies component-wise. For every RGB frame, this yields four spatially and temporally aligned **FracPhys Maps** visualizing the full evolution of fracture dynamics.

### 4.2 FracGen: Teaching Video Generator Fracture Dynamics

Given an object image along with physics controls (tensile load, material, and fracture properties), we describe how we train FracGen to jointly predict fracture RGB video alongside its corresponding dynamic physics maps $\hat{\mathbf{V}} = \{\hat{\mathbf{V}}^{\mathrm{rgb}}, \hat{\mathbf{V}}^{m}\}$. This joint prediction enhances both visual quality and physics accuracy, enabling inspection **without requiring any external physics simulators**.

**Constructing Joint Latents.** We adopt the latent flow-matching video generation model §3. To support joint prediction, we construct an extended clean latent $z^{\mathrm{extend}}$, which is formed by height concatenating encoded version of video latents and its dynamic physics map $z^m$ with $K = 4$:

$$z^{\mathrm{extend}} = \mathrm{Concat}_H\left(z^{\mathrm{rgb}}, z^{\mathrm{flow}}, z^{\mathrm{strain}}, z^{\mathrm{stress}}, z^{\mathrm{damage}}\right) \in \mathbb{R}^{(1+T') \times (K+1)H' \times W' \times C}. \tag{7}$$

Following (Chen et al., 2025), this design choice serves as an effective alternative to channel-wise concatenation without adding parameters. Preprocessed physics maps are projected into latent space via a frozen pretrained 3D VAE encoder $z^m = \mathcal{E}(\mathbf{V}^m)$, avoiding costly retraining while preserving reconstruction fidelity (see §A for more details). Finally, noise is injected into $z^{\mathrm{extend}}$ to form $z_t^{\mathrm{extend}}$.

**Conditioning Physics Signals.** FracGen is conditioned on three physics-guided signals: applied forces, material properties, and fracture properties. We detail their latent space encoding below:

1. **Applied Loading.** We use opposing velocity signals mimicking a universal testing machine (UTM) and render it as moving Gaussian blobs across frames. Blob centers denote grip locations, fixed isotropic covariances define spatial extent, and displacements reflect velocity vectors. We

use a frozen 3D VAE to encode this dynamic signal into latent $y_{\text{force}} \in \mathbb{R}^{(1+T')\times H'\times W'\times 16}$, spatially and temporally aligned with the RGB video representation.

2. **Material Properties.** Material deformation and fracture depend heavily on Young's modulus $E$ and Poisson's ratio $\nu$. Instead of using an explicit encoder, we construct $y_{\text{mat}} \in \mathbb{R}^{(1+T')\times H'\times W'\times 2}$ by broadcasting normalized parameter maps across space and time. We normalize $E \in [10^3, 10^9]$ Pa as $\hat{E} = \frac{\log(E)-6}{3} \in [-1, 1]$ and $\nu \in [0.1, 0.5]$ as $\hat{\nu} = \frac{\nu-0.3}{0.2} \in [-1, 1]$.
3. **Fracture Properties.** To control fracture behavior, we first define fracture categories using a bipolar one-hot category code $\mathbf{c} \in \{-1, +1\}^2$ (low-stretch fracture vs. high-stretch fracture). For fracture conditions, we define the fracture onset and critical stretch, at which fracture initiates and fully fails, respectively, as discussed in §4.1 via $\lambda' = [\lambda_{\text{onset}}, \lambda_{\text{crit}}]$. We normalize $\lambda \in [1, 10]^2$ to $[-1, 1]^2$ via $\frac{\lambda'-5.5}{4.5}$ and broadcast these across space and time via $y_{\text{frac}} \in \mathbb{R}^{(1+T')\times H'\times W'\times 4}$.

Following Wan 2.1-1.3B-Control (Wan et al., 2025), we allocate a 32-channel conditioning layout: 16 channels for $y_{\text{force}}$; 6 channels for $y_{\text{frac}}$ and $y_{\text{mat}}$, and 10 zero-padded channels. We concatenate $z_t^{\text{extend}}$ with these physics-condition channels along the channel axis before patchify as shown in Fig. 3.

**Conditioning Intact Object Image.** We anchor the generated video's first frame to the intact object, allowing the model to infer shape, texture, and geometry priors before force application. Following Wan-DiT's (Wan et al., 2025), we encode this single RGB image using the same frozen 3D VAE. To align with the height of the multi-modal extended latent $z_t^{\text{extend}}$, we zero-pad the reference latent along the spatial height axis to indicate the absence of initial physics-map references. The padded reference latent is then processed by a 2D convolution matching the DiT patch size and flattened into reference tokens, which are prepended to the physics-aware input tokens as shown in Fig.3.

**Training Losses.** Alongside the latent flow-matching loss (§3), we introduce a **latent constitutive regularizer** inspired by linear elasticity and damage-induced stress degradation. Our physical maps encode scalar strain magnitude, von Mises stress and damage, which are further compressed by a pretrained VAE. We therefore learn an approximate relationship among their latent representations.

During training, we estimate the clean joint latent from the noisy input and the model's predicted flow using the training scheduler. We extract the corresponding strain, stress, and damage latents, denoted by $\hat{z}^{\text{strain}}$, $\hat{z}^{\text{stress}}$, and $\hat{z}^{\text{damage}}$. From the damage latent, we construct a channel-wise gate $\hat{g}_d = \text{sigmoid}(\hat{z}^{\text{damage}})$, with the same shape as the stress latent. Motivated by the isotropic linear-elastic relationship $\boldsymbol{\sigma} = \lambda_{\text{L}}\,\text{tr}(\boldsymbol{\varepsilon})\mathbf{I} + 2\mu\boldsymbol{\varepsilon}$, we define an approximate latent stress target:

$$\hat{z}^{\text{stress,cons}} = \lambda_{\text{L}} f_\eta(\hat{z}^{\text{strain}}) + 2\mu g_\xi(\hat{z}^{\text{strain}}), \tag{8}$$

where $f_\eta$ and $g_\xi$ are zero-initialized $1\times1\times1$ convolutions that learn mappings across latent channels. The Lamé parameters are computed from each sample's Young's modulus $E$ and Poisson's ratio $\nu$:

$$\lambda_{\text{L}} = \frac{E\nu}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}. \tag{9}$$

The regularizer couples the predicted stress latent to this target through the damage gate:

$$\mathcal{L}_{\text{const}} = w(t)\left|\hat{z}^{\text{stress}} - (1-\hat{g}_d)\odot\hat{z}^{\text{stress,cons}}\right|_2^2, \tag{10}$$

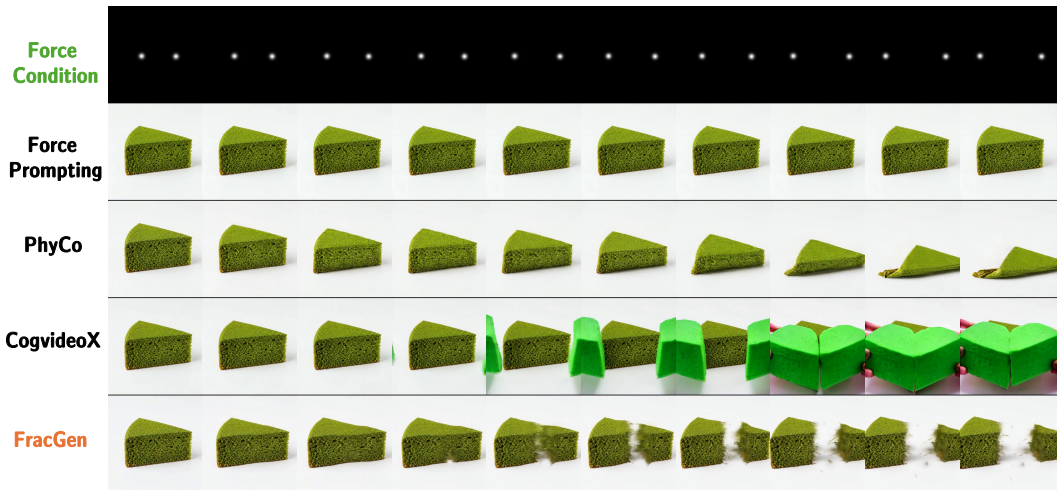
where $\odot$ denotes element-wise multiplication and $w(t)$ assigns greater weight at lower noise levels. When the gate approaches zero, the stress latent is encouraged to match the constitutive approximation; when it approaches one, the target approaches zero latent, a soft degradation prior rather than exact zero stress. This provides a latent-space analogue of stress degradation, without assuming that the VAE coordinates directly represent physical stress or damage. The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{FM}} + \lambda_{\text{const}}\cdot\mathcal{L}_{\text{const}}, \tag{11}$$

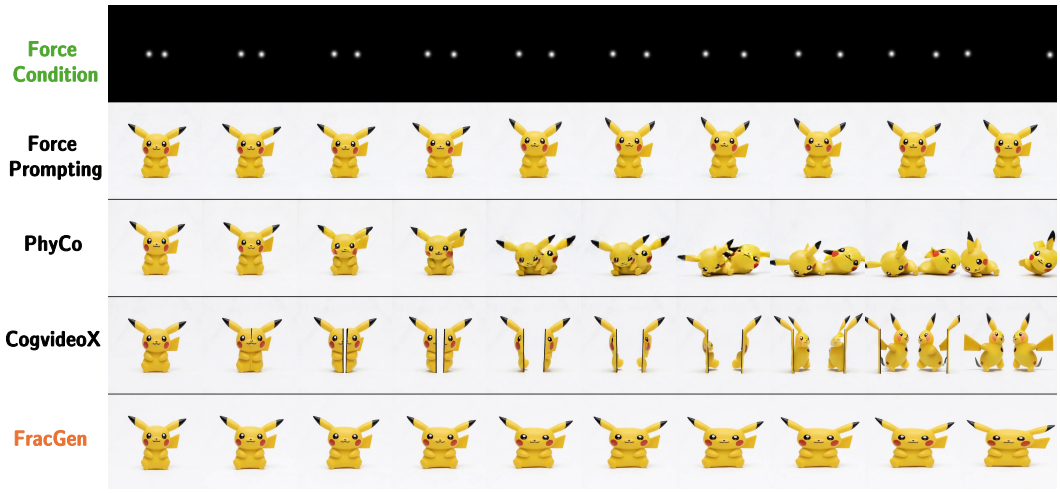
where $\lambda_{\text{const}}$ controls the regularization weight. We train FracGen in three stages. First, we train on RGB fracture videos with physics conditioning. Second, we initialize from this checkpoint and introduce joint prediction of RGB videos and FracPhys Maps. Third, we fine-tune the model with the complete objective in Eq. 11.

## 5 Experiments

**Simulation and Training Details.** For simulation, we sample 10 objects from the Objaverse dataset (Deitke et al., 2022) and SketchFab platform, covering two common material classes in stretch-to-tear




(a) Cake tearing with early break.




(b) Pikachu extreme stretching.

Figure 4: **Qualitative comparison on unseen objects.** We compare generated responses to pulling conditions for (a) a cake that tears after limited stretching and (b) a toy that undergoes sustained elongation. FracGen captures these distinct behaviors while better preserving object appearance; competing methods show limited deformation, collapse or changes in object identity. The top row of each panel shows the force conditioning, and frames progress from left to right.

fracture: 5 high-stretch objects (rubber toys) and 5 low-stretch objects (croissant, cake, muffin, bread, and bread roll). We then use FracSim to yield a paired dataset of RGB fracture videos and dynamic physics maps for generative training by varying different force condition, material properties, and fracture properties (more details can be seen in §A). For training, we leverage this simulated data to train our FracGen model. Among 10 objects, simulated results for 6 objects (3 low-stretch and 3 high-stretch objects) with varying conditions are used for training, yielding 4,212 training samples in total. More details on simulation and training setting could be found in §A).

**Benchmark.** We hold out 4 of the 10 objects for testing against groundtruth, which is mainly for quantitative evaluation against groundtruth. Note that these evaluation set includes 4 objects with varying physics conditions such as forces, material properties, and fracture properties, yielding 2,808 testing samples in total (single view images for these objects are shown in Fig.7). In addition, we collect another set of single object images (see Fig.8) with no 3DGS reconstruction, and hence no simulated video groundtruth. This is to qualitatively show real-world scenario where a user want to fracture an object from a single photo. These are used to demonstrate generalization to new objects, shown in Fig.4a, Fig.4b, Fig.5. We provide more results on our supplement website.

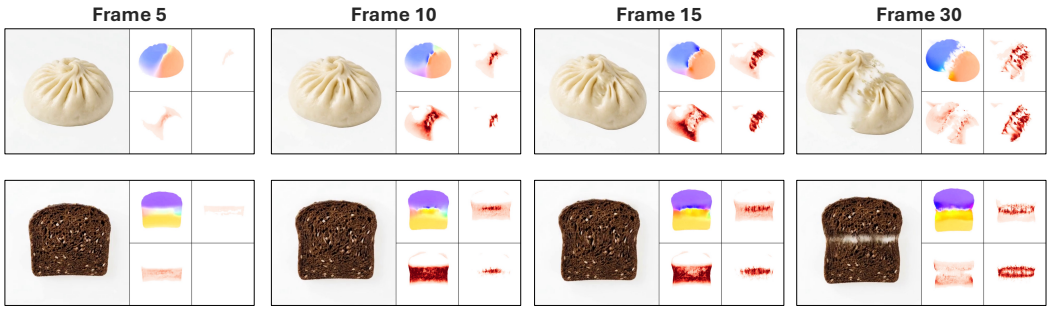



Figure 5: **Generalization to in-the-wild objects**. FracGen only needs one single object image and jointly generates the fracture video (left panel) and its FracPhys maps (right panel, 2×2 grid in top-left to bottom-right order: flow, strain, stress, damage), under varying force directions.

Table 1: Quantitative results on held-out test set. All baselines are fine-tuned on the same dataset.

| Method | Physics Map Pred | FVD ↓ | LPIPS ↓ | PSNR ↑ | SAC ↑ |
|---|---|---|---|---|---|
| ForcePrompting-FT (Gillman et al., 2025) | × | 933.21 | 0.29 | 15.93 | 0.092 |
| PhyCo-FT (Narayanan et al., 2026) | × | 832.12 | 0.33 | 14.18 | 0.434 |
| CogVideoX-5B-I2V-FT (Yang et al., 2025) | × | 1533.21 | 0.32 | 14.12 | 0.432 |
| **FracGen** | ✓ | **266.17** | **0.15** | **21.19** | **0.946** |

**Evaluation Metrics.** We evaluate visual quality using standard metrics: FVD (Unterthiner et al., 2019), LPIPS (Zhang et al., 2018), and PSNR. Since existing physics metrics such as VideoPhy (Bansal et al., 2024), VideoPhy2 (Bansal et al., 2026), and PhysicsIQ (Motamed et al., 2026) focus on general commonsense rather than fracture, we introduce several fracture-aware metrics:

1. **Stretching Axis Coherence (SAC).** SAC measures whether the object's dominant deformation direction aligns with the ground-truth stretching axis. We estimate the dominant stretching direction from point tracks using (Lai et al., 2026) and compute its alignment with the ground-truth axis; higher scores indicate better alignment (see §A for more details).
2. **Damage Progression Alignment (DPA).** Because FracGen conditions on fracture timing and predicts damage maps, we evaluate damage accumulation schedule. We primarily used this for ablation studies since baselines lack this capability, further details are in the §A.
3. **Constitutive Consistency ($R^2_{cons}$).** Unlike other metrics that evaluates physics map individually, this one evaluate the consistency among physics map. Specifically, we use the damage map to mask out already-failed regions and measure linear elasticity in the material that remains intact, precisely the regime where Hooke's law applies. The metric requires no ground truth, as it tests whether the model's own stress and strain predictions are mutually consistent. Like DPA, we use it for ablation studies rather than baseline comparisons. Further details are in the §A.

**Baselines.** We compare FracGen against recent physics-aware video generation models (e.g., ForcePrompting, PhyCo) and video generation baselines (e.g., CogVideoX). We do not compare against PhysCtrl (Wang et al., 2025), as it encodes force as a single force vector, which cannot represent our dual opposing forces. In contrast, ForcePrompting and PhyCo encode force as rendered, pixel-aligned control videos, so we extend them to our two-grip setup by rendering both grips in the control signal, and fine-tune them on the same data (more details in §A). As shown in Tab. 1 and Fig. 4. FracGen achieves superior performance across all metrics, excelling in both video quality and physical alignment (SAC). Qualitatively, baselines fail to capture realistic stretching, and fracture progressions, while FracGen faithfully captures this behavior. More results are on our project website.

**Controllable FracPhys Map and Fracture Prediction.** As shown in Fig. 5, FracGen generates physics maps that align well with fracture videos under varying force conditions (vertical and diagonal tearing directions). Furthermore, Fig. 6 demonstrates enhanced controllability by conditioning on specific material and fracture properties across different settings: low-stretch objects (featuring low stiffness and low critical damage thresholds) versus high-stretch objects (featuring high stiffness and high thresholds). Additional video results are available on the supplement website.

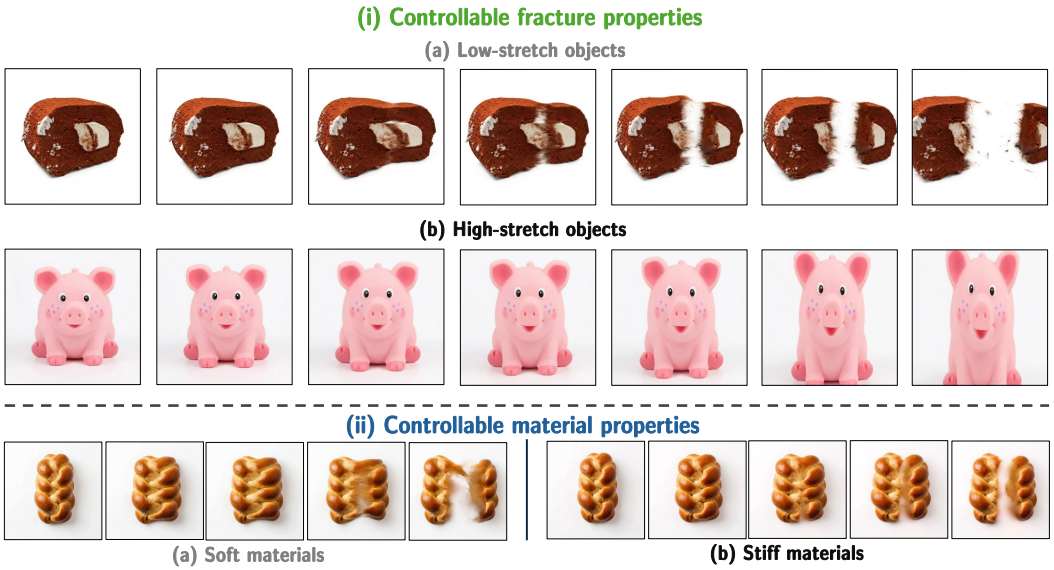



Figure 6: **Controlled deformation and tearing.** (i) Generated sequences for objects with low- and high-stretch fracture settings. (ii) Changing material stiffness for the same object produces different deformation and tearing behavior.

**Ablation: Effects of Joint Prediction.** Stage 1 (physics conditioning, RGB only) serves as our base model. Adding joint FracPhys prediction (Stage 2) improves video quality (FVD 298 → 290) and additionally yields pixel-aligned physical fields that allow the generated dynamics to be inspected directly (Tab. 2).

**Ablation: Effects of Constitutive Loss.** With constitutive loss, Stage 3 yields the largest gains across all levels. Video quality improves (FVD 290 → 266), and damage progression aligns more closely with ground truth (DPA 0.438 → 0.536). Predicted maps become more accurate: overall MSE drops from 0.0253 to 0.0175, while strain error increases slightly, which we attribute to the loss prioritizing stress–strain coupling over per-map fidelity. We also observe a rise in consistency ($R^2_{\text{cons}}$ 0.378 → 0.902). As reference, the $\mathbf{R}^2_{Cons}$ score for groundtruth physics maps is **0.948**.

Table 2: Ablation: **video quality**, **physics alignment**, and evaluation of FracPhys map prediction (MSE). Stage 1 has no physics map; hence **DPA** and FracPhysMap evaluation are not applicable.

| Metric | Stage 1 (× FracPhys) | Stage 2 (✓, w/o Eq.10) | Stage 3 (✓, w. Eq.10) |
|---|---|---|---|
| *Video Quality & Physics Alignment* | | | |
| FVD ↓ | 298.04 | 289.65 | **266.17** |
| LPIPS ↓ | 0.1555 | 0.1550 | **0.1502** |
| PSNR ↑ | 20.73 | 20.82 | **21.19** |
| SAC ↑ | 0.9460 | 0.9393 | **0.9462** |
| DPA ↑ | – | 0.438 | **0.536** |
| *FracPhys Map Evaluation* | | | |
| Flow ↓ | – | 0.03735 | **0.01147** |
| Stress ↓ | – | 0.01464 | **0.00429** |
| Strain ↓ | – | **0.02017** | 0.02681 |
| Damage ↓ | – | 0.02904 | **0.02740** |
| Overall ↓ | – | 0.02530 | **0.01749** |
| $\mathbf{R}^2_{Cons}$ ↑ | – | 0.378 | **0.902** |

## 6 Discussion

We show that pretrained video generators can be adapted to learn material-dependent stretching and tearing from simulation-derived supervision. To our knowledge, FracGen is the first video generation framework to jointly predict material-dependent tearing and the associated physical fields from a single image and specified physical conditions. FracSim provides paired videos and physical maps for joint prediction, while physics-consistency losses encourage agreement among the predicted fields. These allow us to inspect how estimated strain, stress and damage evolve. As video models improve and the materials community releases more datasets, we envision a feedback loop: experimental data improves generative models, and fast predictions help prioritize new experiments. With experimental validation, this could accelerate material-response prediction and support materials discovery.

Our work focuses on deformation and tearing under tensile loading. Brittle fracture and fragmentation under impact are outside our training and evaluation scope. Extending FracGen to these material types would require appropriate simulation data and failure models. The model also inherits the assumptions of its training simulator, so any errors in FracSim would also reflect in FracGen. Finally, multi-step diffusion sampling limits inference speed. Few-step distillation could reduce this cost, provided it preserves fracture behavior and consistency among the predicted physical fields.

### AI USE STATEMENT

We used generative AI tools to assist with language editing and translation, drafting parts of the paper text, summarizing existing literature, proposing title and keyword candidates, setting up environments for baseline methods, cleaning and reformatting datasets, and building the project website. We did not use generative AI to generate synthetic datasets, propose or refine hypotheses, design research methodology or experiments, or implement our method; theoretical and proof-related uses are not applicable to this work. All AI-assisted outputs were reviewed by the authors: drafted and translated text was edited, baseline and data scripts were verified by running them and inspecting outputs, and literature summaries were checked against the original papers. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

This work involves no human subjects, personal data, or sensitive content.

### REPRODUCIBILITY STATEMENT

FracSim's damage model and physics-map rendering are specified in §4.1, with full simulation parameters in Tab.3 and §A.4; physics-map preprocessing is described in §A. FracGen's architecture, conditioning layout, losses, and three-stage training schedule and other details are given in §4.2 and §A. Evaluation metrics are fully defined in §A.5, and baseline adaptations and training protocols are described in §A.7. We will release code, simulation configurations, and the generated dataset upon publication.

## REFERENCES

Simon Baker, Stefan Roth, Daniel Scharstein, Michael J. Black, J.P. Lewis, and Richard Szeliski. A database and evaluation methodology for optical flow. In *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, 2007. doi: 10.1109/ICCV.2007.4408903.

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, 2024. URL https://arxiv.org/abs/2406.03520.

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. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=HA8KSQW7SO.

Anand Bhattad, Daniel McKee, Derek Hoiem, and David Forsyth. Stylegan knows normal, depth, albedo, and more. *Advances in Neural Information Processing Systems*, 2023.

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.

Zhaoxi Chen, Tianqi Liu, Long Zhuo, Jiawei Ren, Zeng Tao, He Zhu, Fangzhou Hong, Liang Pan, and Ziwei Liu. 4dnex: Feed-forward 4d generative modeling made easy. *arXiv preprint arXiv:2508.13154*, 2025.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *arXiv preprint arXiv:2212.08051*, 2022.

Xiaodan Du, Nicholas Kolkin, Greg Shakhnarovich, and Anand Bhattad. Generative models: What do they know? do they know things? let's find out! *arXiv preprint arXiv:2311.17137*, 2023.

Yutao Feng, Xiang Feng, Yintong Shang, Ying Jiang, Chang Yu, Zeshun Zong, Tianjia Shao, Hongzhi Wu, Kun Zhou, Chenfanfu Jiang, and Yin Yang. Gaussian splashing: Unified particles for versatile motion synthesis and rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 518–529, June 2025.

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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=eX5aXfJQZc.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018.

Tianyu Huang, Haoze Zhang, Yihan Zeng, Zhilu Zhang, Hui Li, Wangmeng Zuo, and Rynson W. H. Lau. Dreamphysics: learning physics-based 3d dynamics with video diffusion priors. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i4.32389. URL https://doi.org/10.1609/aaai.v39i4.32389.

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015.

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

Lazar M. Kachanov. Rupture time under creep conditions. *International Journal of Fracture*, 97(1): 11–18, 1999. ISSN 1573-2673. doi: 10.1023/A:1018671022008. URL https://doi.org/10.1023/A:1018671022008.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

Zihang Lai, Eldar Insafutdinov, Edgar Sucar, and Andrea Vedaldi. Cowtracker: Tracking by warping instead of correlation. *arXiv preprint arXiv:2602.04877*, 2026.

Jean Lemaître. A continuous damage mechanics model for ductile fracture. *Journal of Engineering Materials and Technology-transactions of The Asme*, 107:83–89, 1985. URL https://api.semanticscholar.org/CorpusID:136363677.

Yuchen Lin, Chenguo Lin, Jianjin Xu, and Yadong MU. OmniphysGS: 3d constitutive gaussians for general physics-based dynamics generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9HZtP6I5lv.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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

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)*, 2024b.

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. Xpbd: position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games*, pp. 49–54, 2016.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 948–958. IEEE, 2026.

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

Sansit Patnaik and Fabio Semperlotti. Variable-order fracture mechanics and its application to dynamic fracture. *npj Computational Materials*, 7(1):27, feb 2021. ISSN 2057-3960. doi: 10.1038/s41524-021-00492-x. URL https://doi.org/10.1038/s41524-021-00492-x.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FjNys5c7VyY.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. FVD: A new metric for video generation, 2019. URL https://openreview.net/forum?id=rylgEULtdN.

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

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. In *NeurIPS*, 2025.

Xiaogang Wang, Hongyu Wu, Wenfeng Song, and Kai Xu. Fracture-GS: Dynamic fracture simulation with physics-integrated gaussian splatting. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=zcAwK50ft0.

Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. Cd-mpm: Continuum damage material point methods for dynamic fracture animation. *ACM Transactions on Graphics (TOG)*, 38(4):119, 2019.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4389–4398, June 2024.

Xiaoyan Xing, Konrad Groh, Sezer Karagolu, Theo Gevers, and Anand Bhattad. Luminet: Latent intrinsics meets diffusion models for indoor scene relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

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. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Ke Zhang, Cihan Xiao, Yiqun Mei, Jiacong Xu, and Vishal M. Patel. Think before you diffuse: Llms-guided physics-aware video generation, 2025. URL https://arxiv.org/abs/2505.21653.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

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

Zeshun Zong, Xuan Li, Minchen Li, Maurizio M Chiaramonte, Wojciech Matusik, Eitan Grinspun, Kevin Carlberg, Chenfanfu Jiang, and Peter Yichen Chen. Neural stress fields for reduced-order elastoplasticity and fracture. *arXiv preprint arXiv:2310.17790*, 2023.

# A Appendix

## A.1 Project Website

We provide link to our project website at https://iclr11751.github.io/.

## A.2 Pre-processing dynamic physics map

FracSim 4.1 produces a set of per-frame 2D physics maps: three scalar fields (stress, strain, damage) and one vector field (flow). Before VAE encoding, each raw map is normalized per video and linearly rescale (clipping) into $[0, 1]$. For the scalar fields (stress, strain, damage), the normalized scalar is replicated identically across the three RGB channels, yielding a grayscale RGB video in which pixel intensity encodes physical magnitude; background pixels normalize to exactly zero and render as black. For the flow field, which is a 2D vector rather than a scalar, we instead apply a Middlebury-style optical-flow color-wheel encoding (Baker et al., 2007). Zero flow therefore also renders as black, consistent with the convention used for the scalar fields (Note that this is what the model sees during training, for visualization as shown in Fig.5, we map it back to white background for the ease of visualization). Since the 3D VAE expects RGB input, each physics map is now a standard RGB video after this pre-processing, requiring no architectural change to accommodate a new modality. We feed each pre-processed physics-map video into the frozen 3D VAE to obtain its latent, as described in §4.2.

## A.3 Additional Background in Material Point Methods

Material Point Method (MPM) (Jiang et al., 2016) is a hybrid Eulerian–Lagrangian discretization for continuum mechanics. MPM represent a body with a set of Lagrangian material points (particles), carrying position $\mathbf{x}_p$, velocity $\mathbf{v}_p$, mass $m_p$, and deformation gradient $\mathbf{F}_p$. The deformation gradient $\mathbf{F}_p$ tracks the particle's accumulated deformation, encoding rotation and strain which mainly drives the constitutive strain-stress response. At each MPM simulation step, mass and momentum are transferred to grid nodes (P2G); grid velocities are updated under internal stress and external constraints; the updated velocities are interpolated back to the particles for advection (G2P); and each particle's deformation gradient is evolved accordingly

**Particle-to-grid (P2G) transfer.** Mass and momentum are accumulated onto the grid with the affine particle-in-cell (APIC) scheme (Jiang et al., 2015), augmenting each particle's velocity with a local affine velocity term $\mathbf{C}_p^t$:

$$m_i^t = \sum_p w_{ip}^t \, m_p, \tag{12}$$

$$(m\mathbf{v})_i^t = \sum_p w_{ip}^t \, m_p \left[\mathbf{v}_p^t + \mathbf{C}_p^t \left(\mathbf{x}_i^t - \mathbf{x}_p^t\right)\right], \tag{13}$$

where $w_{ip}^t$ is the B-spline weight coupling particle $p$ to grid node $i$.

**Grid update.** Each grid node velocity is integrated forward under the net nodal force $\mathbf{f}_i$, combining internal and external forces:

$$\mathbf{v}_i^{t+1} = \mathbf{v}_i^t + \frac{\Delta t}{m_i^t} \, \mathbf{f}_i\big(\mathbf{x}_i^t; \boldsymbol{\theta}_p\big) \, . \tag{14}$$

The force follows from a hyperelastic energy $\Psi(\mathbf{F})$, and $\theta_p$ gathers the governing material properties, Young's modulus $E$, and Poisson's ratio $\nu$.

**Grid-to-particle (G2P) transfer.** The updated nodal velocities are interpolated back to the particles, whose positions are then advanced, the local affine velocity term is also updated correspondingly:

$$\mathbf{v}_p^{t+1} = \sum_i w_{ip}^t \, \mathbf{v}_i^{t+1}, \qquad \mathbf{x}_p^{t+1} = \mathbf{x}_p^t + \Delta t \, \mathbf{v}_p^{t+1}, \qquad \mathbf{C}_p^{t+1} = \frac{4}{(\Delta x)^2} \sum_i w_{ip}^t \mathbf{v}_i^{t+1} (\mathbf{x}_i - \mathbf{x}_p^t)^T \tag{15}$$

**Deformation gradient update.** Each particle's deformation gradient is updated from the velocity gradient sampled off the grid:

$$\mathbf{F}_p^{t+1} = \left[ \mathbf{I} + \Delta t \sum_i \mathbf{v}_i^{t+1} \left( \nabla w_{ip}^t \right)^\top \right] \mathbf{F}_p^t. \tag{16}$$

### A.4 Additional Simulation and Training Details

**From Cauchy stress tensor to von Mises scalar.** We first compute the deviatoric Cauchy stress tensor $\boldsymbol{s}_p = \tilde{\boldsymbol{\sigma}}_p - \frac{1}{3}\operatorname{tr}(\tilde{\boldsymbol{\sigma}}_p)\mathbf{I}$ and collapse it into a scalar von Mises equivalent stress $\tilde{\boldsymbol{\sigma}}_{\mathrm{vM},p} = \sqrt{\frac{3}{2}\boldsymbol{s}_p : \boldsymbol{s}_p}$.

**Relation to elastoplasticity.** Our two material classes differ in their hyperelastic energy density (fixed corotated vs. neo-Hookean), elastic constants, and damage thresholds: FracSim couples hyperelasticity to continuum damage, with no yield surface and no return mapping ($F^P = I$). This is deliberate, as damage thresholds expose fracture timing as a directly controllable quantity, which is what FracGen conditions on. The cost is that we reproduce the visual distinction between materials that tear at low stretch and those that sustain large stretch, but not the permanent deformation of true ductile (plastic) fracture.

**Simulation Details.** Following PhysGaussian (Xie et al., 2024), we optimize 3DGS per object using an anisotropy regularizer and internal particle filling. Fracture is produced by a continuum damage model coupled to MPM (§4.1), and diversity is obtained by sampling loading, constitutive model, material properties constants, and fracture properties. Every simulation is fully specified by a single configuration, which makes each sample reproducible. All runs use 100 frames with 100 MPM substeps per frame ($\Delta t = 10^{-4}$ s), a $150^3$ background grid over the normalized domain $[0, 2]^3$, and no gravity, so that the observed deformation is attributable solely to the prescribed load. Across combinations of object, loading, and material/fracture parameters, FracSim yields a paired dataset of RGB fracture videos and dynamic physics maps for generative training.

i) **Uniaxial Opposing Forces:** Tensile loading is applied kinematically. We select two thin slabs of MPM particles on opposite sides of the object center (thickness 0.05 in simulation units, i.e. roughly one grid cell, and wide enough to span the object's cross-section) and prescribe their velocities to $\pm v\,\hat{\mathbf{u}}$ for the entire clip, where $\hat{\mathbf{u}}$ is the tearing axis. Because the velocity is enforced on particles rather than on grid nodes, the grips move rigidly with the material and the remaining particles respond through the constitutive law. We vary the speed $v \in [0.2, 1.5]$, the tearing axis over 12 directions in the image plane ($0^\circ$ to $165^\circ$ in $15^\circ$ steps), the grip offset from the center (0.10–0.25), and the uniaxial tensile stretch mode.

ii) **Constitutive Model:** We match the constitutive model to the deformation regime each material reaches before it fails. For low-stretch objects like bread, we use the fixed-corotated model, which is accurate and stable at small-to-moderate strain. For high-stretched object like rubber toys, it sustain stretches of $3\times$ or more before failing, which requires a genuinely finite-strain energy density; for these we use the compressible neo-Hookean model, a standard model of rubber elasticity.

iii) **Material Properties:** Young's modulus $E$ and Poisson's ratio $\nu$ are sampled per class (Tab. 3). Larger $E$ gives sharper, more localized cracks with faster snap-back; smaller $E$ gives broader necking before the tear; $\nu$ controls how much the cross-section thins under tension. Rubber objects uses much weaker velocity damping than the bread-like objects so its stored elastic energy is released visibly on fracture.

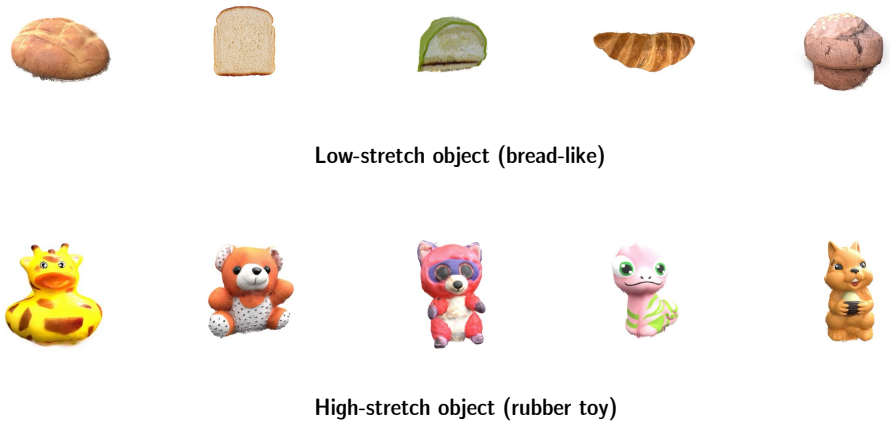



Figure 7: Ten objects used for simulation. First 3 of each category is then used for training while last two are used for testing and benchmarking.

iv) **Fracture Properties:** We augment each particle with a scalar damage $d_p \in [0, 1]$ (0 intact, 1 broken) (Lemaître, 1985). After each substep we take the principal stretches of $\mathbf{F}_p$ (its singular values) and update damage from the largest one as discussed in Eq.4 and degrade the particle's stress contribution to the grid. The onset stretch $\lambda_{\text{onset}}$ sets how far material can stretch before weakening; while the crit stretch indicates the critical stretch. Varying these thresholds independently of $E$ and $\nu$ decouples *when* an object breaks from *how* it deforms beforehand, and is the main source of qualitative diversity in fracture patterns.

Table 3: Per-class simulation parameters.

| | **Low-stretch objects** | **High-stretch objects** |
|---|---|---|
| **Constitutive model** | Fixed corotated | Neo-Hookean |
| **Young modulus** $E$ | $\{2 \times 10^3, 6 \times 10^3, 10^4\}$ | $\{2 \times 10^4, 6 \times 10^4, 10^5\}$ |
| **Poisson ratio** $\nu$ | $\{0.2, 0.25, 0.3\}$ | $\{0.45, 0.47, 0.49\}$ |
| **Onset, critical stretch** $(\lambda_{\text{on}}, \lambda_{\text{cr}})$ | $\{(1.5, 2.3), (2.0, 3.0)\}$ | $\{(2.6, 3.0), (3.0, 3.3)\}$ |
| **RPIC damping / grid velocity decay** | 0.5 / 0.9995 | 0.005 / 0.9998 |
| **Grip speed** $v$ | $\{0.5, 0.6, 0.7, 0.8\}$ | $\{0.6, 0.8, 1.2, 1.4\}$ |
| **Tearing axes** | $\{0°, 15°, \ldots, 165°\}$ | |
| **Grip offset from center** | $\{0.10, 0.15, 0.20, 0.25\}$ | |
| **Density / gravity** | 200 / 0 | |

**Training Details.** We leverage simulated data from FracSim to train our FracGen model. Among 10 objects, simulated results from 6 objects are used for training (3 low-stretch and 3 high-stretch) and 4 objects are used for testing, and benchmarking (2 low-stretch and 2 high-stretch). Our model is finetuned from a pre-trained Wan 2.1-Fun-V1.1-1.3B-Control (Wan et al., 2025) using LoRA (Hu et al., 2022) with rank 64, applied to the query, key, value, output, and feed-forward projections of the DiT backbone. We use the AdamW optimizer with a learning rate of 1e-4, weight decay of 0.01, and a cosine learning-rate schedule with a 40-step warmup. Each video clip consists of 81 frames, and the model is trained with a per-GPU batch size of 1 and gradient accumulation of 4 steps across 4 GPUs, yielding an effective batch size of 16. The total training samples are 4,212 training samples, and total testing/benchmarking samples are 2,808. Additionally, we also collect 10 random object images to further test generalization of FracGen across different new objects. We show images of these objects in Fig.8.

### A.5 Additional Details on Evaluation Metric Design

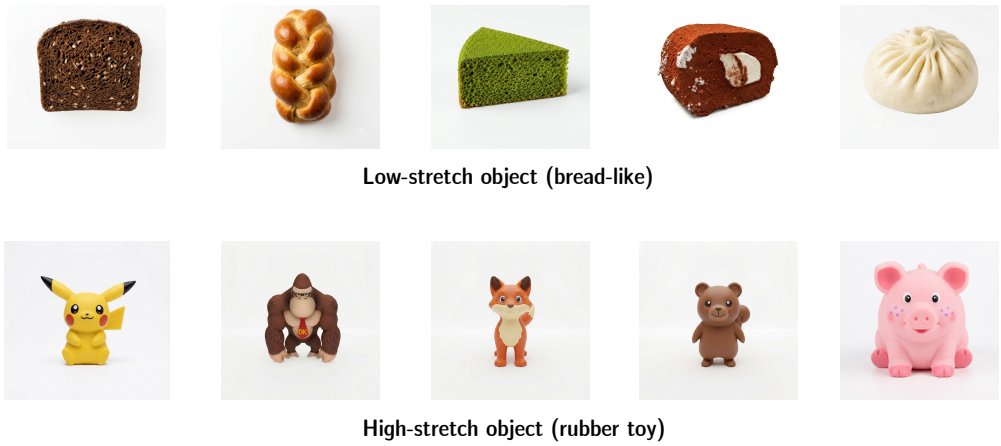



Figure 8: More ten random objects used for testing FracGen generalization. Results can be found on our website.

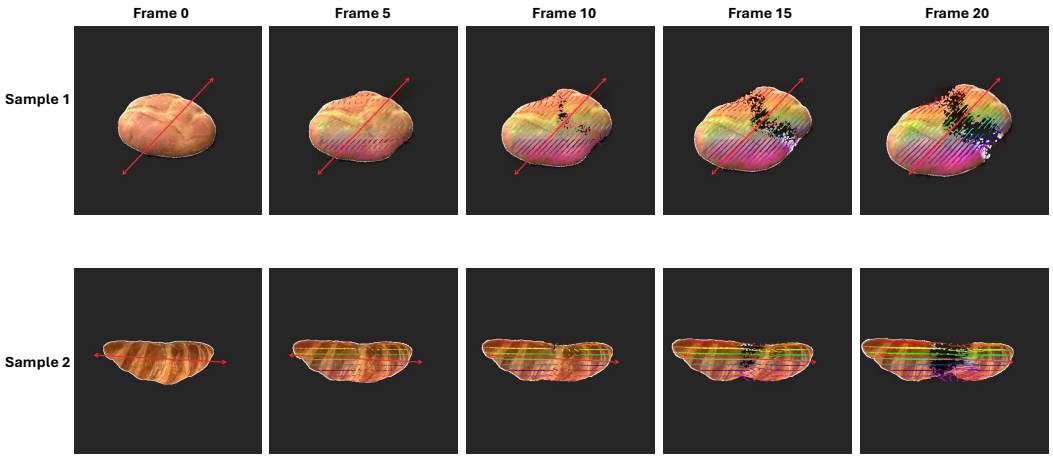



Figure 9: We show illustration of using point tracks to extract principal stretching axis.

**Stretching Axis Coherence (SAC).** Given a fracture video, we first segment out main object using SAM (Kirillov et al., 2023) and run point track algorithm (Lai et al., 2026) to extract all track information. Point track results are visualized in Fig.9. Given 2D point tracks $\{\mathbf{p}_i(t)\}_{i=1}^{N}$, we estimate the dominant deformation axis over an early window $[t_0, t_1]$ and compare it against the same quantity measured on a reference (ground-truth) clip.

***Displacement and drift removal.*** For each point $i$ visible at both $t_0, t_1$, we extract its displacement $\mathbf{u}_i = \mathbf{p}_i(t_1) - \mathbf{p}_i(t_0)$. The mean $\bar{\mathbf{u}} = \frac{1}{N}\sum_i \mathbf{u}_i$ is dominated by the object's rigid drift, since opposing deformation components cancel in the average. We subtract this quantity from each point's displacement, giving deformation-relative motion $\hat{\mathbf{u}}_i = \mathbf{u}_i - \bar{\mathbf{u}}$.

***Noise gating***. Point tracks might be unstable due to noisy prediction from tracking algorithm or other noisy motion. We use point's displacement magnitude $w_i = \|\hat{\mathbf{u}}_i\|$ as a gating function to drop these noisy points with $w_i < \tau$. If fewer than $k$ points survive, we determine the axis to be undefined (the object has not yet shown measurable deformation).

***Dominant axis extraction***. Since opposite sides of a stretching object move in opposite directions, averaging directions directly would cancel to zero. We instead form the orientation tensor over unit directions $\mathbf{e}_i = \hat{\mathbf{u}}_i / w_i$:

$$Q = \frac{\sum_i w_i \, \mathbf{e}_i \mathbf{e}_i^\top}{\sum_i w_i} \in \mathbb{R}^{2\times 2}, \tag{17}$$

Intuitively, the outer product $\mathbf{e}_i\mathbf{e}_i^\top$ does not care about direction and hence $\mathbf{e}\mathbf{e}^\top = (-\mathbf{e})(-\mathbf{e})^\top$. Weighted average of these terms give the overall voting principal axis. We then perform eigen-decompose on the symmetric matrix $Q$ to extract orthogonal eigenvectors with eigenvalues $\lambda_{\max} \geq \lambda_{\min} \geq 0$; the eigenvector of $\lambda_{\max}$ is the dominant axis $\mathbf{a}$. We show in Fig.9 the principal stretching axis (overlaying red line) computed using this approach from two samples of our fracture video. We also determine a confidence score,

$$c_{\text{pred}} = 1 - \frac{\lambda_{\min}}{\lambda_{\max}} \in [0, 1] \tag{18}$$

Intuitively, $c_{\text{pred}} \to 1$ when motion is tightly aligned along one line, $c_{\text{pred}} \to 0$ when motion is isotropically scattered.

***Computing SAC.*** Given the principal stretching axis of prediction and groundtruth fracture video $\mathbf{a}_{\text{pred}}, \mathbf{a}_{\text{gt}}$, we measure their angular deviation via

$$\theta = \arccos\big(\,|\mathbf{a}_{\text{pred}} \cdot \mathbf{a}_{\text{gt}}|\,\big) \in [0^\circ, 90^\circ], \tag{19}$$

where the absolute value dropping the axis sign ambiguity. This is squashed into a bounded similarity score via a Gaussian kernel,

$$s_{\text{raw}} = \exp\Big(-(\theta/\sigma)^2\Big) \in (0, 1], \tag{20}$$

In this way, a perfect match ($\theta = 0$) scores 1 and the score decays smoothly with angular error (we set $\sigma = 15^\circ$). The final SAC score additionally is computed as,

$$\text{SAC} = s_{\text{raw}} \cdot c_{\text{pred}}, \tag{21}$$

with $c_{\text{pred}} := 0$ whenever the predicted axis itself is undefined (scored as a hard failure, SAC $= 0$, rather than excluded).

**Damage Progression Alignment (DPA).** Let $d(\mathbf{x}, t) \in [0, 1]$ denote the damage field of a clip with $T$ frames and $H \times W$ pixels, read from its damage-map video. For each frame we record its total damage, normalized by frame size,

$$a(t) = \frac{1}{HW} \sum_{\mathbf{x}} d(\mathbf{x}, t), \tag{22}$$

together with the peak damage the clip ever attains, $a^\star = \max_t a(t)$. The idea of DPA is to assess whether the prediction and the ground truth reach the same damage checkpoints at the same moments. A checkpoint is a fraction $\beta$ of a clip's own peak damage, and the moment at which the clip reaches it is

$$f_\beta = \min\big\{t : \; a(t) \;\geq\; \beta\, a^\star\big\}, \qquad \tau_\beta = \frac{f_\beta}{T}. \tag{23}$$

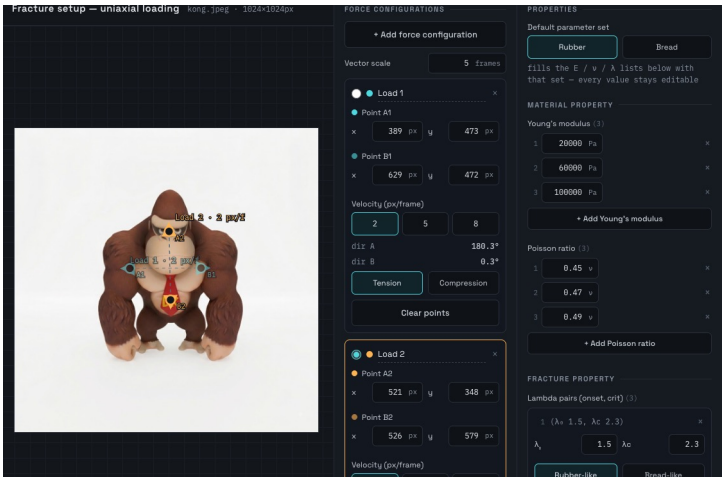



Figure 10: Screenshot of annotation tool to provide physics condition for FracGen

With $\Delta\tau_\beta = \tau_\beta^{\text{pred}} - \tau_\beta^{\text{gt}}$, DPA is the normalized agreement averaged over a set of checkpoints $B$:

$$\text{DPA} = \frac{1}{|B|}\sum_{\beta\in B}\exp\Big(-\big(\Delta\tau_\beta/\sigma\big)^2\Big) \in (0,1], \qquad \sigma = 0.1. \tag{24}$$

We sweep through a set of $B = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

**Constitutive Consistency ($R^2_{\text{cons}}$).** Let $\sigma(\mathbf{x},t)$, $\varepsilon(\mathbf{x},t)$ and $d(\mathbf{x},t)$ denote a predicted stress, strain and damage fields, read from its physics-map videos. In intact material linear elasticity requires stress to be proportional to strain; once damage accumulates the material softens, so stress falls while strain continues to grow. Therefore, we restrict attention to material that has not yet failed,

$$S = \big\{(\mathbf{x},t):\ d(\mathbf{x},t) \le d_{\max}\big\}, \qquad d_{\max} = 0.05, \tag{25}$$

On $S$ we fit a single slope through the origin — zero strain must produce zero stress — and record the coefficient of determination,

$$k = \frac{\sum_S \varepsilon\,\sigma}{\sum_S \varepsilon^2}, \qquad R^2_{\text{cons}} = 1 - \frac{\sum_S \big(\sigma - k\,\varepsilon\big)^2}{\sum_S \sigma^2}. \tag{26}$$

$R^2_{\text{cons}}$ therefore measures how tightly a single linear stress–strain law explains the maps. To validate the robustness of this metric, we evaluate the identical quantity on the ground-truth maps, obtaining $\mathbf{R}^2_{\text{cons}} = 0.948$, which serves as the upper bound. Against it, our FracGen (with constitutive loss) attains $0.902$ (95% of the ceiling) while the variant without constitutive loss reaches only $0.378$, indicating that the latter's predicted stress and strain fields are not governed by any single elastic law.

### A.6 Additional Details on Building Annotation Tool

For comprehensive annotating force applied on the single image object, we build an annotation tool that takes an input image and allow user to annotate physics condition. Specifically, users are given a UI to specify uniaxial applied load, material properties, and fracture properties. We show a screenshot of this annotation tool in Fig.10.

### A.7 Additional Details on Baseline Implementation

We provide more details on our attempt on fine-tuning baseline for fair comparison as below.

**Matched training protocol.** All fine-tuned baselines are trained on the identical FracSim corpus and train/test split used for FracGen (§5). Rather than using FracGen's hyperparameters on backbones

Table 4: Comparison of baselines with their zero-shot and fine-tuned (-FT) version.

| Method | FVD ↓ | LPIPS ↓ | PSNR ↑ | SAC ↑ |
|---|---|---|---|---|
| ForcePrompting | 1226.32 | 0.32 | 15.07 | 0.075 |
| ForcePrompting-FT | **933.21** | **0.29** | **15.93** | **0.092** |
| PhyCo | 1595.01 | 0.38 | 12.51 | 0.415 |
| PhyCo-FT | **832.12** | **0.33** | **14.18** | **0.434** |
| CogVideoX-5B-I2V | 1844.92 | 0.34 | 13.78 | **0.470** |
| CogVideoX-5B-I2V-FT | **1533.21** | **0.32** | **14.12** | 0.432 |

of differing scale, each model is trained under its default setting described in its original work. When a baseline exposes a conditioning channel for applied force we use it, adapted to our two-grip setup as described below; when it does not, force is described in the text prompt only, which cannot express pull direction or magnitude.

1. **ForcePrompting** (Gillman et al., 2025) fine-tunes a ControlNet on CogVideoX-5B-I2V to follow a rendered Gaussian-blob control video encoding a single applied force (a point poke or a global wind direction). Since tearing requires two opposing pulls rather than one, we extend its point-force conditioning with a custom *double-force* variant that renders and sums two independent Gaussian blobs at the two grip locations, matching the force representation FracGen consumes. We then finetune this ControlNet on our FracSim force-blob/RGB pairs under the matched protocol. The backbone remains frozen, following the original method. There are no conditions on other channels such as Young's modulus, Poisson's ratio, or fracture thresholds, so this baseline sees the same grips and the same pull velocities as FracGen but cannot be told what the object is made of.
2. **PhyCo** (Narayanan et al., 2026) conditions a frozen Cosmos-Predict2 Video2World backbone with a ControlNet over friction, restitution, deformability, and a single static applied force point. We adapt its data pipeline to accept two independent force points rather than one, to represent our two-grip pulling setup. We then finetune their ControlNet on FracSim data under the same matched training control.
3. **CogVideoX-5B-I2V** (Yang et al., 2025) has no mechanism for force or physics conditioning at all. Hence, we use intact-object as first frame condition and use a caption to describe tearing behavior (e.g. "Two opposing forces pull on the object, tearing it apart at the center"). We fine-tune it on our the same data provided by FracSim with LoRA of rank 64. Pull direction and magnitude cannot be expressed beyond this, so identical captions are used for opposing pull directions of the same object.

After fine-tuning, every baseline was trained on the same data distribution as similar to FracGen, making it a fair comparison. We also report results for these baselines before and after fine-tuning on the data set produced by our FracSim in Tab.4. This shows that even with training on the same dataset, our FracGen can still outperform these fine-tuned baselines.

### A.8 Additional Results

We present more comparison and controllable results on our website. Additionally, we show qualitative example for ablation studies on the effect of physics map prediction as well as physics consistency loss in Fig.11. As shown, physics map prediction helps to improve physics fidelity of tearing bread (softening first before breaking). On the other hand, physics consistency loss helps to ensure consistency and accurate physics map prediction for stress, strain, and damage.

### A.9 More Related works

**3D Gaussians Splatting.** 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents a static 3D scene as a set of anisotropic Gaussian kernels $\{G_p\}_{p=1}^{\mathcal{P}}$. Each Gaussian $G_p = \{x_p, \sigma_p, A_p, \mathcal{C}_p\}$ carries a center $x_p$, opacity $\sigma_p$, covariance $A_p$, and spherical harmonic coefficients $\mathcal{C}_p$. For rendering,

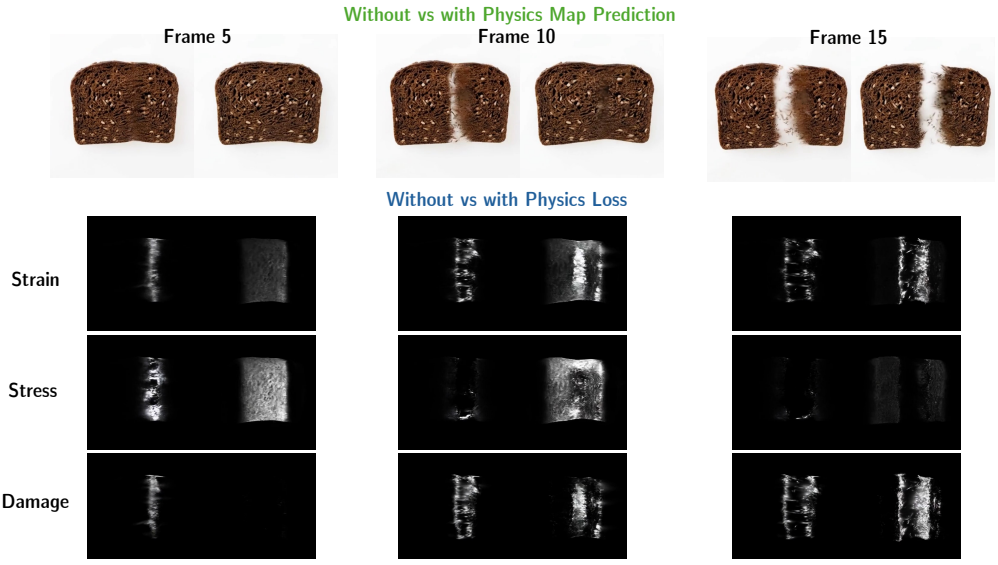



Figure 11: **Ablation Studies.** Comparison results for ablating the effect of physics map prediction and physics consistency loss.

the Gaussians are projected onto the image plane and composited front-to-back:

$$G_p = \{x_p, \sigma_p, A_p, C_p\}, \qquad C = \sum_{p \in \mathcal{P}} \alpha_p \, \mathrm{SH}(l_p; \mathcal{C}_p) \prod_{j=1}^{p-1} (1 - \alpha_j). \tag{27}$$

where $\alpha_p$ is the effective opacity at pixel location, $\mathrm{SH}(l_p; \mathcal{C}_p)$ is the view-dependent color given view $l_p$, and the product term is the accumulated transmittance. The Gaussians are optimized via photometric loss across multiple views. We adopt the original 3DGS framework to reconstruct static objects, which are then passed to FracSim (§4.1) to simulate fracture dynamics.

**Ductile Fracture Mechanics.** Dynamic fracture (Kachanov, 1999; Lemaître, 1985; Wolper et al., 2019; Patnaik & Semperlotti, 2021) is among the most challenging phenomena to model in continuum mechanics. A central concept is material *ductility*—the capacity to deform plastically before failure—which separates the two canonical fracture modes. *Ductile* fracture occurs in materials that sustain large deformation under load, typically necking before the cross-section separates, whereas *brittle* fracture occurs at very low strains, often below $5\%$, with little observable deformation prior to failure. In this work, we target the kinematic regime of ductile fracture (large deformation before failure), modeled via hyperelasticity with stretch-driven damage rather than plasticity. Specifically, we study fracture under uniaxial tensile loading, where two opposing forces act along a common axis, causing the object to stretch, deform, and eventually break. Under load, the body extends along the loading (longitudinal) direction while contracting in the transverse directions. Prior work on fracture simulation achieves high physical fidelity but requires expert knowledge to configure the governing parameters. CD-MPM (Wolper et al., 2019) models dynamic ductile fracture through continuum damage mechanics, coupling a phase-field damage evolution with the MPM to capture large-deformation crack propagation. To reduce the heavy runtime and memory cost of full-order solvers, Neural Stress Fields (NSF) (Zong et al., 2023) learns a low-dimensional manifold of the Kirchhoff stress field through an implicit neural representation, enabling reduced-order simulation of elastoplasticity and fracture. These methods, however, target physical simulation or geometric fragmentation, and expose control only through low-level physical parameters. To the best of our knowledge, our work is the first to bring stretch-to-tear fracture to a generative model—teaching a video generator the full deformation-to-fracture progression under comprehensive input conditioning, so that users can control which object to break, when it should break, how it breaks. FracSim captures the kinematic signature of this regime — large stretch, necking, progressive separation — through damage accumulation rather than plastic flow.