MeGAS: Thermomechanical Dynamic Gaussian Splatting for Thermophysical Scene Editing

Zesong Yang[1*], Yuanhang Lei[1*], Liyuan Cui[1], Yihang Chen[1], Jiaer Huang[1], Boming Zhao[1], Peter Yichen Chen[2], Hujun Bao[1], and Zhaopeng Cui[1†]

[1]State Key Laboratory of CAD&CG, Zhejiang University
[2]University of British Columbia

Project Page: zju3dv.github.io/MeGAS

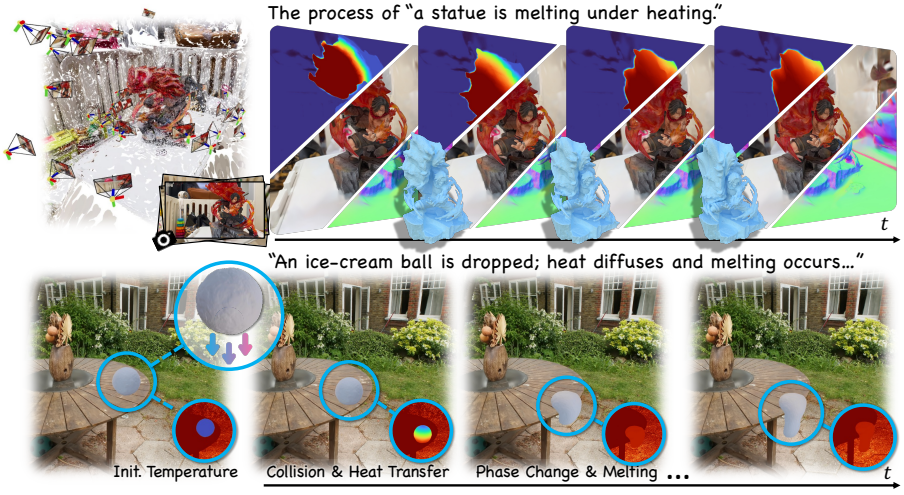

Fig. 1: Thermophysical editing and photorealistic rendering of real scenes. MeGAS integrates thermomechanical dynamics into 3D Gaussian Splatting, enabling physically grounded editing with temporally evolving temperature fields, while maintaining smooth and continuous geometry under large deformations.

Abstract. Recent advances integrate physically grounded Newtonian dynamics with neural rendering frameworks, narrowing the gap between photorealistic scene reconstruction and physics-based animation. However, existing approaches focus on mechanically driven dynamics while neglecting temperature, a fundamental yet invisible physical factor underlying phenomena such as melting, solidification, and other thermomechanical processes. In this paper, we propose **MeGAS**, a novel framework that incorporates thermomechanical phase-change dynamics into 3D Gaussian Splatting (3DGS). Specifically, we propose a new thermomechanical dynamic Gaussian Splatting representation that augments 3DGS with temperature attributes and employs a heat advection-diffusion solver with MPM dynamics incorporating phase transitions, enabling physically plausible and visually realistic synthesis of thermophysical phenomena. Furthermore, a new topology-adaptive Gaussian rendering strategy is proposed to mitigate cracking and floaters under extreme

* Equal contribution. † Corresponding Author.

deformation. Extensive experiments demonstrate that MeGAS produces physically consistent thermomechanical behavior while maintaining high-fidelity photorealistic rendering, advancing toward physics-integrated world models.

Keywords: 3D Gaussian Splatting · 3D Scene Editing · Physics-based Animation

1 Introduction

Recent advances in neural scene representations, such as Neural Radiance Fields (NeRF) [35] and 3D Gaussian Splatting (3DGS) [20], have significantly advanced photorealistic scene reconstruction and rendering from real-world imagery, substantially promoting the development toward visually faithful digital twins and world models. While these approaches achieve impressive visual fidelity, a practical open-world simulator requires not only visual realism but also the controllable synthesis of physically plausible phenomena that faithfully adhere to the physical laws. To this end, recent research has integrated physical simulators into neural rendering, enabling the simulation of diverse phenomena such as elastic deformation [12, 26, 45, 51, 53, 58], fluid animation [11], and weather effects [10, 27, 44].

However, most existing approaches primarily focus on mechanically driven dynamics, while thermomechanical dynamics remain largely unexplored. In the physical world, temperature, an invisible yet fundamental quantity, governs a broad spectrum of visually perceivable phenomena, ranging from the melting of ice cream to the solidification of lava. Although several works have incorporated thermal information, they typically treat temperature as an auxiliary sensing modality for multispectral reconstruction [29, 32, 38, 55] or estimate spatio-temporal temperature fields restricted in static scenes [9, 54]. Nevertheless, since thermomechanical dynamics are among some of the most intricate natural phenomena, integrating thermomechanical effects into neural rendering frameworks remains highly challenging. **First**, thermomechanical dynamics necessitate not only the enforcement of momentum conservation but also the coupled modeling of heat transfer and phase transitions to produce physically plausible outcomes. **Second**, thermomechanical coupling (e.g., in melting) introduces extreme deformations and topological changes, posing a significant challenge for neural rendering representations that typically assume continuous or smoothly deforming geometry. Such extreme deformations cause sharp and spiky artifacts in splatting-based rendering, and the object surface tends to crack and expose interior regions, further breaking the geometric continuity for stable rendering. Consequently, a straightforward combination of existing Gaussian pipelines with thermomechanical dynamics fails to ensure visually stable results.

To address these challenges, we propose **MeGAS**, a physics-integrated neural rendering framework that integrates thermomechanical dynamics with 3D Gaussian Splatting, enabling controllable and thermophysically grounded editing of real scenes as shown in Fig. 1. To this end, we first propose a new thermomechanical dynamic Gaussian Splatting representation that augments each

Gaussian with a per-splat temperature attribute and supports direct rendering of a consistent thermal field within the standard 3DGS formulation. Temperatures can be flexibly initialized and manipulated through user-specified heat sources, providing intuitive and physically grounded editing control. To evolve this thermal field, we adopt a heat advection-diffusion solver defined on a simulation grid. The solver propagates heat diffusion through the reconstructed scene, yielding a smooth, physical temperature distribution. We further couple this thermal field to a phase-change–aware constitutive model switching, enabling particle-level melting behavior: once a particle's temperature exceeds the melting threshold, its constitutive model transitions from an elastic solid to a viscoplastic formulation, producing physically grounded and spatially localized melting dynamics.

Since melting induces large-scale deformation and surface topology changes, directly applying such extreme motion to vanilla or PhysGaussian-style [53] 3DGS leads to severe artifacts such as cracking, floaters, and interior exposure. We therefore design a novel Topology-Adaptive Gaussian Rendering strategy to ensure stable rendering throughout the melting process. Specifically, during reconstruction, we regularize Gaussian anisotropy to suppress excessively elongated kernels, enforce volumetric consistency, and prune internal Gaussians that would otherwise become visible under deformation. During animation, we perform an online surface-aware refinement that combines adaptive densification with local Gaussian kernel refitting guided by the evolving surface, which seals emerging cracks and maintains a coherent splat distribution as the object undergoes drastic thermomechanical change.

Our main contributions are summarized as follows:

- We introduce **MeGAS**, a novel framework that integrates thermomechanical phase-change dynamics into 3D Gaussian Splatting (3DGS), enabling physically plausible and photorealistic synthesis of thermophysical phenomena from real-world imagery.
- We propose a new thermomechanical dynamic Gaussian Splatting representation that jointly models appearance and a controllable thermal field, while integrating heat advection-diffusion and temperature-controlled constitutive model switching, enabling melting dynamics controlled by phase change.
- We develop a novel topology-adaptive Gaussian Rendering strategy with anisotropy regularization, internal-free filtering, and online surface-aware refinement to effectively mitigate cracking, floaters, and interior exposure under extreme deformation.
- Extensive experiments on real and synthetic scenes demonstrate that our method achieves thermophysically consistent scene editing while preserving photorealistic rendering quality.

2 Related Work

Physics-Integrated Neural Rendering. Recent studies have integrated physical simulators into neural rendering frameworks, bridging photorealistic recon-

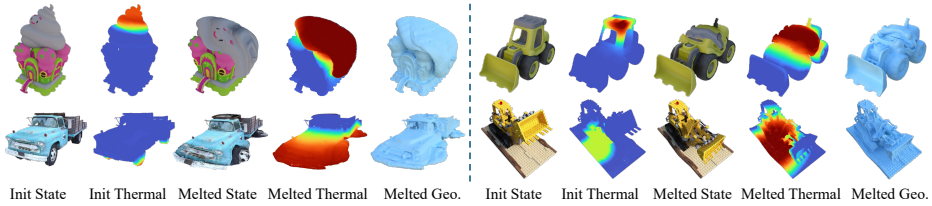

Fig. 2: Thermo-geometry evolution with MeGAS. Under user-specified heating source, MeGAS performs thermomechanically consistent phase-change simulation, steering the evolution of temperature fields and shape and delivering controllable, targeted melting without compromising geometric smoothness.

struction with physically grounded realism. Early works [12, 26] embed continuum and elastodynamic models into NeRF, enabling geometry-consistent simulation of elastic and interactive deformations. Subsequent researches [19, 25, 28, 40, 48, 51, 53, 58] adapt 3D Gaussian Splatting (3DGS) to model mechanical dynamics while preserving rendering realism. Beyond elasticity, [11] unifies position-based dynamics with 3DGS for fluid animation, and [10, 27, 44] synthesize environmental effects such as rainfall, snowfall and fire within neural rendering frameworks. Despite these advances, existing approaches primarily focus on *mechanically driven* or *environmentally induced* phenomena. In contrast, many natural phenomena are *thermally driven*, where invisible temperature variations govern visually perceivable behaviors such as softening and melting. Our work makes the first exploration of thermomechanical phase-change dynamics in neural rendering, addressing this underexplored yet fundamental dimension of physics integration.

Thermal-Integrated 3DGS. Recent works extend neural rendering to thermal modalities by incorporating infrared imagery for multimodal scene reconstruction. [29, 32, 38, 55] leverage paired RGB–thermal inputs to improve novel view synthesis via multispectral supervision, yet they rely purely on image-level supervision without explicit thermodynamic modeling. [9, 52, 54] couple 3DGS with simplified heat diffusion to estimate spatiotemporal temperature variations, achieving dynamic thermal field reconstruction but still limited to static scenes without considering more complex thermodynamic phenomena. In contrast, we integrate a physically grounded thermomechanical model into 3DGS, capturing the coupled heat transfer and deformation dynamics for realistic melting synthesis.

3 Preliminaries

3D Gaussian Splatting. Given posed RGB image sequences as input, we first reconstruct the scene with 3D Gaussian Splatting (3DGS) [20]. 3DGS represents the scene with a set of explicit 3D Gaussians, each parameterized by a center μ and a full covariance matrix Σ:

$$G(x) = e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}. \tag{1}$$

For differentiable rendering optimization, Σ is decomposed into a scaling matrix S and rotation matrix R, i.e., $\Sigma = RSS^TR^T$, where S and R are stored by a 3D scale vector s and a quaternion q respectively. Given a viewing transformation W, the 3D Gaussians are projected to the image plane, and we obtain the 2D covariance matrix Σ' and 2D center location μ' as:

$$\Sigma' = JW\Sigma W^T J^T, \quad \mu' = JW\mu, \tag{2}$$

where J is the Jacobian of the affine approximation of the projective transformation. Then we can use the point-based volume rendering to render the color C of each pixel with N ordered 3D Gaussians:

$$C = \sum_{i \in N} T_i c_i \alpha_i, \quad T_i = \prod_{j=1}^{i-1}(1-\alpha_j), \tag{3}$$

with alpha α_i defined as:

$$\alpha_i = o_i e^{-\frac{1}{2}(x-\mu_i')^\top \Sigma_i'^{-1}(x-\mu_i')}. \tag{4}$$

Here, o_i and c_i denote the learned opacity and color parameterized by spherical harmonics for each Gaussian. Besides, we define the normal n_i as the shortest axis direction.

Constitutive Governing Equations. We use the hybrid Eulerian-Lagrangian framework, Material Point Method (MPM) [16, 18], to simulate continuum dynamics. The governing equations follow the conservation of mass and momentum:

$$\frac{D\rho}{Dt} + \rho\nabla\cdot\mathbf{v} = 0, \quad \rho\frac{D\mathbf{v}}{Dt} = \nabla\cdot\boldsymbol{\sigma} + \mathbf{f}^{ext}, \tag{5}$$

where ρ, $\mathbf{v}$, and $\boldsymbol{\sigma}$ denote the density, velocity, and Cauchy stress tensor respectively, and $\mathbf{f}^{ext}$ is the external force. The stress tensor $\boldsymbol{\sigma}(\mathbf{x}, t)$ is derived from the hyperelastic constitutive model as:

$$\boldsymbol{\sigma}(\mathbf{x}, t) = \frac{1}{\det(\mathbf{F})}\frac{\partial \Psi}{\partial \mathbf{F}^E}(\mathbf{F}^E)^T, \tag{6}$$

where $\Psi : \mathbb{R}^{3\times3} \to \mathbb{R}$ is the hyperelastic energy density function, and $\mathbf{F}^E \in \mathbb{R}^{3\times3}$ represents the elastic part of the deformation gradient $\mathbf{F}$. By specifying different formulations of Ψ, various constitutive models can be incorporated to characterize distinct material behaviors.

4 Methods

As shown in Fig. 3, given posed RGB image sequences as input, we first reconstruct the scene with 3D Gaussian Splatting (3DGS) [20]. Since melting serves as a representative and visually salient stress test of thermomechanical coupling, we adopt it as the primary instantiation to present our framework. To simulate temperature-driven editing, we propose a thermomechanical dynamic Gaussian Splatting representation. Each Gaussian is augmented with a temperature attribute and we couple 3DGS to an MPM simulator with a heat advection–diffusion solver and a phase-change–aware constitutive model switching, enabling melting dynamics at the particle level. To maintain stable rendering

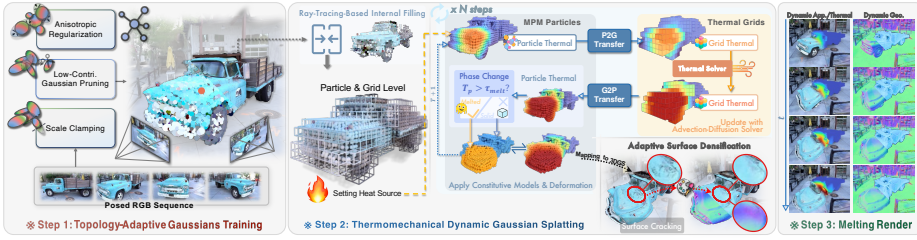

Fig. 3: System overview. Starting from posed RGB sequences, we reconstruct the scene with 3DGS. To mitigate floaters and interior artifacts under extreme deformation, we introduce an interior-free uniform-Gaussian regularization that suppresses anisotropy and prunes unsupported interior splats. We then provide volumetric support via ray-tracing–based internal filling. Next, we augment 3DGS with per-Gaussian temperature and, driven by user-specified heat sources, MeGAS updates temperatures using a grid-based heat advection-diffusion solver within an MPM simulator, and performs temperature-dependent phase-aware constitutive switching. Deformations are mapped back to 3DGS, and we apply adaptive surface densification to handle topology change and seal cracks. Finally, we synthesize physically plausible thermomechanical dynamics with photorealistic appearance and temporally smooth geometry.

under extreme deformation, we introduce a topology-adaptive Gaussian rendering strategy to obtain a uniform-distributed, volume-consistent, floater-free 3DGS, and incorporate a Ray-Tracing–based internal filling strategy to provide volumetric support for MPM. Finally, to prevent surface cracking during large deformations, we employ implicit-surface–guided densification that adaptively fills hollow regions with smoothly aligned Gaussian splats.

4.1 Thermomechanical Dynamic Gaussian Splats

We augment each Gaussian with a temperature attribute $T_i \in \mathbb{R}$, and the corresponding temperature rendering of each pixel is obtained following Eq. 3:

$$\hat{T} = \sum_{i \in N} w_i \, T_i, \tag{7}$$

where w_i are the Gaussian weights along the viewing ray.

Thermal Advection-Diffusion Solver. To drive melting, we require a physically grounded evolution of the temperature field. Instead of heuristically editing, we solve heat diffusion on a simulation grid. Specifically, we model heat transport with the advection-diffusion equation (ADE) [3]:

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \alpha \nabla^2 T + S, \tag{8}$$

where T is the temperature field, $\mathbf{u}$ is the thermal velocity field, α is the thermal diffusivity, and S represents external heat sources. The velocity $\mathbf{u}$ is modeled by the Navier-Stokes equations (NSE) [5]:

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}, \tag{9}$$

with density ρ, pressure p, dynamic viscosity μ, and external forces $\mathbf{f}$.

Algorithm 1 Thermomechanical MPM for Thermophysical Dynamics

Notation: p, i: particle & grid indices; m: mass; x: position; w_{ip}: P2G2P weight function; T: temperatures; $\mathbf{F}^E$: elastic deformation gradient; τ: Kirchhoff stress;
while time_step < total_steps **do**
  for all grid i **do** ▷ **Particle-to-Grid Transfer**
    Transfer Mass & Momentum to Grid via APIC
    $m_i T_i \leftarrow \sum_p w_{ip} m_p \big(T_p + (\mathbf{x}_i - \mathbf{x}_p) \cdot \nabla T_p\big)$ ▷ Transfer Thermal Field
  end for
  for all grid i **do** ▷ **Grid Update**
    Update Grid Velocities
    Update Thermal Boundary
    $T_i = \text{LBM\_Update}(T_i)$ ▷ **Thermal Evolution**
  end for
  for all particle p **do** ▷ **Grid-to-Particle Transfer**
    Interpolate Velocity from Grid
    Update Particle Mechanical Properties
    $T_p \leftarrow \sum_i w_{ip} T_i$
    $\nabla T_p \leftarrow \sum_i T_i (\nabla w_{ip})^T$
    if $T_p > T_{\text{melt}}$ **then** ▷ **Phase Transition Update**
      $\{\tau_p, \mathbf{F}_p^E\} \leftarrow \tau_{\text{liquid}}(\mathbf{F}_p^E)$ ▷ **Switch Material**
    else
      $\{\tau_p, \mathbf{F}_p^E\} \leftarrow \tau_{\text{solid}}(\mathbf{F}_p^E)$
    end if
  end for
end while

To advance the coupled system (8)–(9), we adopt the Lattice Boltzmann Method (LBM) that solves the flow and temperature fields and naturally captures both diffusion and advection [47]. Additionally, thermal buoyancy is incorporated via the Boussinesq approximation as a body force [14]:

$$\mathbf{F}_b \;=\; \beta\, \rho_0 \left(T - T_0\right) \mathbf{g}, \tag{10}$$

where β is the thermal expansion coefficient, ρ_0 the reference density at temperature T_0, and $\mathbf{g}$ gravity. We add $\mathbf{F}_b$ to $\mathbf{f}$ in (9) for correction, avoiding explicit density updates while enabling accurate temperature-driven convection. See **supplementary material** for more details.

Thermal-Driven Material Point Method. While we have defined how temperature is represented and evolved, building upon the standard MPM framework [16, 18, 59], we extend the particle and grid states with thermal attributes, enabling temperature-dependent material behavior. During each time step, in addition to the standard MPM updates of mechanical properties, particle temperatures are transferred to the grid (P2G), followed by thermal boundary updates and temperature diffusion on the grid solved via a Lattice Boltzmann Method (LBM). The updated grid temperature is then interpolated back to the particles

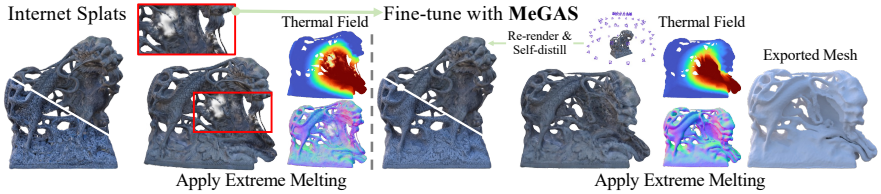

Fig. 4: Naïve combination fails under extreme topology changes. Our thermomechanical augmentation can directly apply to Internet-pretrained splats, nevertheless, naïve 3DGS cracks under large deformations. With low-cost self fine-tuning (1000 steps $\approx$ 10s), **MeGAS** makes pretrained splats robust and yields smoother geometry.

(G2P). If the particle temperature exceeds a predefined melting threshold T_{melt}, we switch its material phase and update the constitutive model accordingly. For the solid state, we adopt a StVK elasticity model [21], while the melted state can be treated as a viscoplastic material and we model it using a Herschel–Bulkley plasticity formulation [57]. The thermal boundaries are user-defined heat source regions that directly control the spatial extent and intensity of melting, as shown in Fig. 2. The overall thermomechanical pipeline is summarized in Algorithm 1 and an expanded version with details is provided in **supplementary material**.

Mapping Deformation to 3D Gaussians. After each simulation step t, following [53], the elastic deformation gradient $\mathbf{F}_p^{E,t}$ for splat G_p updates the covariance via an affine push-forward:

$$\boldsymbol{\Sigma}_p^t = \mathbf{F}_p^{E,t}\, \boldsymbol{\Sigma}_p^0\, (\mathbf{F}_p^{E,t})^\top. \tag{11}$$

Then we decompose $\boldsymbol{\Sigma}_p^t = \mathbf{R}_p^t(\mathbf{S}_p^t\mathbf{S}_p^{t\,T})\mathbf{R}_p^{t\,T}$ to obtain the updated rotation matrix $\mathbf{R}_p^t$ and scaling matrix $\mathbf{S}_p^t$, and the orientation-dependent appearance is updated by the relative rotation $\Delta\mathbf{R}_p^t = \mathbf{R}_p^t\mathbf{R}_p^{0T}$: we rotate the view direction d for SH color as: $c_p^t(\mathbf{d}) = c_p^0\big((\Delta\mathbf{R}_p^t)^\top\mathbf{d}\big)$.

4.2 Topology-Adaptive Gaussian Rendering

After obtaining particle-level melting animation through our thermomechanical MPM-LBM simulation, we achieve a unified framework for elastoplastic simulation and photorealistic rendering. However, since melting entails extreme non-rigid deformation and topology change (e.g., surface cracking, internal exposure), naïvely applying PhysGaussian under such conditions leads to severe visual artifacts, e.g., hollow surfaces and needle-like internal floaters, as shown in Fig. 4. To address these issues, we propose a topology-adaptive 3D Gaussian Splatting (3DGS) framework specifically designed for stable rendering under large-scale melting deformation with the following innovations.

Internal-Free Uniform Gaussian Regularization. The original 3D Gaussian Splatting (3DGS) reconstructs object details through outside-in multi-view photometric supervision, which leads to undesirable internal distribution of redundant Gaussians, and consequently degrades rendering quality when hidden Gaussians are exposed under large deformation. Additionally, the reconstructed surface Gaussians typically exhibit highly irregular anisotropy and non-uniform spatial distribution, resulting in needle-like visual artifacts during melting.

We build upon the planar-based Gaussian Splatting [17] and further enhance the training stage. Previous work [4] observes that enforcing a delta-like per-pixel alpha weight distribution along each ray effectively suppresses floating artifacts in volumetric rendering. Inspired by this observation, since 3DGS employs an explicit point-based representation, we propose a direct yet effective strategy—pruning low-contribution Gaussians during training. Specifically, we traverse all training views and record the maximum weight $w_i^{\max}$ of each Gaussian G_i across all pixels. Then we prune Gaussians whose $w_i^{\max}$ falls below a predefined threshold τ_{prune} which contribute negligibly to the rendered appearance: $\mathcal{G}' = \{ G_i \in \mathcal{G} \mid w_i^{\max} > \tau_{\text{prune}} \}$. The remaining subset $\mathcal{G}'$ is subsequently fine-tuned until convergence, eliminating redundant internal Gaussians and continuing the original 3DGS training pipeline.

Irregular anisotropic Gaussians often lead to needle-like kernels or large volumetric discrepancies, which deteriorate rendering stability under large deformation. Following [11, 53], we adopt an anisotropy loss that penalizes excessive scale ratios and encourages the kernel to approach a disk-like shape:

$$\mathcal{L}_{\text{aniso}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \max \left(\frac{S_p^1}{S_p^2} - a,\, 0 \right), \tag{12}$$

where a is a ratio threshold, and $\mathbf{S}_p = \{S_p^1, S_p^2, S_p^3\}$ denotes the principal scalings of Gaussian G_p, with S_p^1 being the largest and S_p^3 the smallest scaling factor.

We further apply scale clamping with a threshold τ_{scale} to prevent large Gaussian volumetric discrepancies: $S_p \leftarrow \min(S_p, \tau_{\text{scale}})$. Densification in 3DGS naturally compensates for potential sparse regions caused by the clamping, yielding a uniform, volume-consistent, and surface-aligned Gaussian distribution. As illustrated in Fig. 8, our regularization produces a uniform Gaussian distribution and reduces internal floaters compared to the regularization used in PhysGaussian [53] and VR-DoH [33] without compromising rendering fidelity.

Ray-Tracing-Based Internal Filling. To enable a physically consistent melting simulation, the object interior needs to be properly filled for volumetric support. Since our reconstruction is internal-floater-free, we adopt a simple yet robust Gaussian-ray-tracing–based strategy [36] to detect interior regions. For each sampled ray, if the accumulated alpha mask indicates full opacity and the blended normal aligns with the ray direction, the ray is classified as interior-originated. As shown in Fig. 5, we divide the object's bounding box into uniform grids, each grid casts rays along principal directions. Grids with a majority of interior-originated rays are supplemented into internal particles.

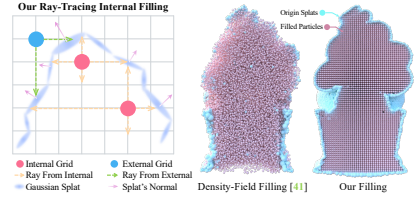

Fig. 5: Our ray-tracing internal filling. Starting from an interior-free, uniformly distributed GS model, we cast rays from uniform grids and detect interior samples by enforcing directional consistency with Gaussian ray-traced normals. This produces uniform, well-conditioned volumetric filling for physically plausible melting.

Adaptive Surface Cracking Densification. Melting induces extreme non-

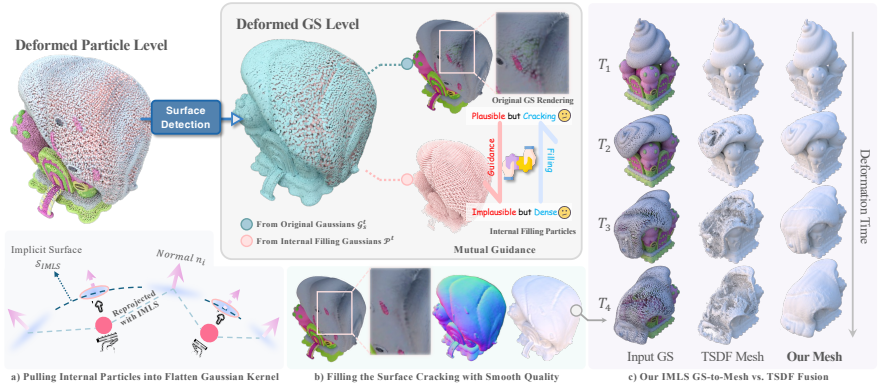

Fig. 6: Implicit-surface–guided adaptive densification. Internal particles migrating to cracks are detected and projected onto an IMLS-fitted implicit surface to reorient Gaussian kernels with surface-aligned normals. Locality-aware densification seals emerging cracks and maintains coherent splat distributions under severe thermomechanical topology change. Integrating Gaussian Splatting with IMLS exports temporally more coherent and smoother surfaces while preserving fine geometric details.

rigid deformation and frequent topology changes, which often cause surface cracking and hollow artifacts, as shown in Fig. 6. We observe that the internal particles filled naturally migrate toward surface voids during the MPM evolution, partially compensating for the missing geometry. However, these internal particles lack valid Gaussian kernel parameters (covariance/appearance), thus cannot be rendered coherently. Let the initial surface Gaussians be denoted as $\mathcal{G}_s = \{G_i^s = (\mathbf{x}_i^s, \mathbf{R}_i^s, \mathbf{S}_i^s, \mathbf{o}_i^s)\}$. After deformation step t, the internal filled particles $\mathcal{P}_t = \{\mathbf{p}_j^t\}$ geometrically occupy the cracked regions of $\mathcal{G}_s$ but without valid kernel parameters. To reconstruct a coherent surface representation, we leverage the deformed surface Gaussians $\mathcal{G}_s^t = \Phi_t(\mathcal{G}_s)$ to guide the local Gaussian kernel fitting of nearby internal particles.

Surface Detection. We first identify which internal particles are suitable for use in cracking filling with surface detection. Specifically, we place virtual cameras uniformly distributed over 360°. Internal particles $\mathbf{p}_j^t$ are initialized as isotropic Gaussian splats G_j^p with opacity $\mathbf{o}_j = 1$ and radius r_j:

$$r_j = \kappa \, \min\left\{ \min_i \|\mathbf{p}_j^t - \mathbf{x}_i^s\|, \; \min_{k \neq j} \|\mathbf{p}_j^t - \mathbf{p}_k^t\| \right\}, \tag{13}$$

where $\kappa \in (0, 1)$ is a shrinkage ratio. For each pixel, we record the Gaussian splat G_i with maximal alpha contribution along the ray, and yields the surface subset $\{\widetilde{\mathcal{G}}_s^t, \; \widetilde{\mathcal{P}}^t\}$.

Implicit-Surface Guidance. As shown in Fig. 6, to restore plausible Gaussian kernel for $\widetilde{\mathcal{P}}^t$, we fit an implicit surface $\mathcal{S}_{\text{IMLS}}$ from $\widetilde{\mathcal{G}}_s^t$ via Implicit Moving Least Squares (IMLS) [2, 23, 39]:

$$f(\mathbf{x}) = \frac{\sum_i \phi(\|\mathbf{x} - \widetilde{\mathbf{x}}_i^s\|) \, (\widetilde{\mathbf{n}}_i^s)^\top (\mathbf{x} - \widetilde{\mathbf{x}}_i^s)}{\sum_i \phi(\|\mathbf{x} - \widetilde{\mathbf{x}}_i^s\|)}, \tag{14}$$

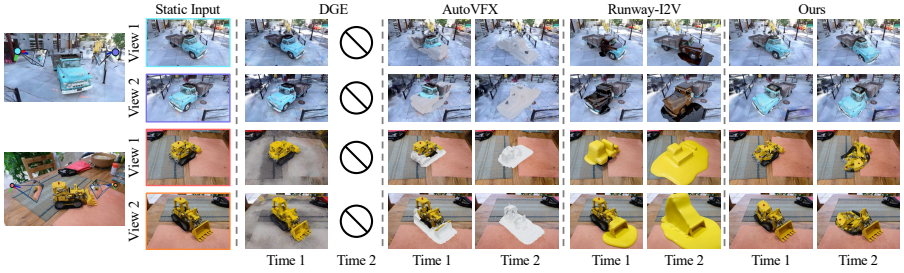

Fig. 7: Melting-style scene editing comparisons. Qualitative results with multi-view, temporally evolving melting edits across DGE, AutoVFX, Runway Gen-4.5, and our MeGAS show that prior methods lack physical realism, photorealism, or multi-view consistency, whereas MeGAS preserves photorealistic appearance and delivers physically plausible, temporally consistent melting. **Please refer to our supp. video for more vivid dynamic comparisons.**

where $\phi(\cdot)$ is a Gaussian weighting kernel and $\widetilde{\mathbf{n}}_i^s$ is the normal derived from $\widetilde{G}_i^s \subset \widetilde{\mathcal{G}}_s^t$. Each internal particle is then projected onto $\mathcal{S}_{\text{IMLS}}$ to obtain its surface normal $\mathbf{n}_j$ and rotation matrix $\mathbf{R}_j$, and we further assign color $\mathbf{c}_j$ as the average of K nearest Gaussians in $\widetilde{\mathcal{G}}_s^t$, yielding a surface-aligned kernel:

$$G_j^p = \left(\mathbf{p}_j^t, \mathbf{R}_j, \mathbf{S}_j, \mathbf{o}_j, \mathbf{c}_j\right), \text{ where } \mathbf{S}_j = r_j \mathbf{I}. \tag{15}$$

This adaptive densification seamlessly re-covers cracked regions with correctly oriented internal Gaussians and yields smooth, stable rendering under extreme melting deformation. Meanwhile, the fitted implicit surface $\mathcal{S}_{\text{IMLS}}$ also enables direct extraction of high-quality meshes. See more details in **supplementary**.

5 Experiments

We evaluate our MeGAS on diverse synthetic and real-world scenes to demonstrate our capability to produce physically plausible, photorealistic thermophysical editing and robustness under extreme deformations.

5.1 Evaluation on Thermophysical Scene Editing

Datasets and Comparison. To evaluate MeGAS as a practical editing tool, we perform melting-style edits on in-the-wild scenes from MipNeRF360 [4] and Tanks and Temples [22] datasets. For each scene, we first obtain a high-quality static 3DGS reconstruction, and then compare our melting edits against three representative state-of-the-art approaches: 1) DGE [8], a text-driven diffusion-based editing method for 3DGS; 2) AutoVFX [15], which leverages an LLM [1] to generate scripts for external physics engines to perform mesh-based melting; 3) Runway Gen-4.5 [42], a leading commercial image-to-video generation model that augments a single input image with text-guided dynamic effects. **Notably**, existing physics-integrated neural rendering methods do not support thermophysical animation or phase-change-aware dynamics, therefore, we compare against these general editing baselines (including a commercial open-domain video generator) to contextualize our thermophysical editing. All methods are

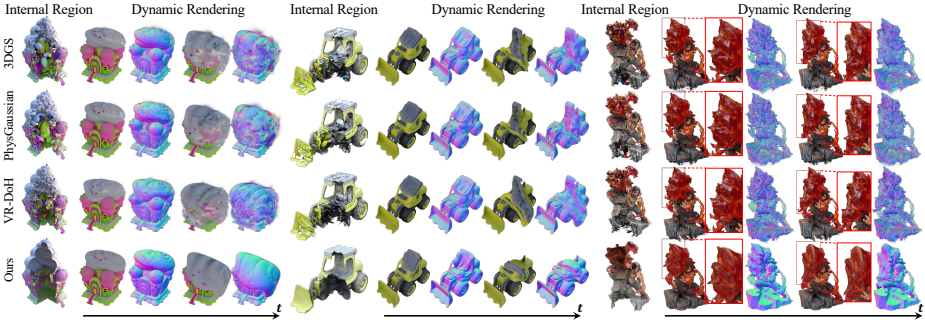

Fig. 8: Extreme melting deformation comparisons. Internal cross-sections and dynamic frames show that Original 3DGS, PhysGaussian, and VR-DoH accumulate internal floaters, needle-like artifacts, and surface cracking under large deformations, whereas MeGAS yields volume-consistent, crack-free geometry and stable rendering throughout melting. **Please refer to supp. video for vivid dynamic comparison.**

provided with comparable editing instructions, and detailed prompts and parameter settings are reported in **supplementary material**.

Results. Qualitative comparisons are shown in Fig. 7. DGE fails to induce coherent dynamic melting behavior and exhibits limited physical realism. AutoVFX generates plausible mesh-based melting, but decoupled geometry/shading from radiance causes non-photorealistic compositions and inconsistencies with the original scene. Runway Gen-4.5 produces visually vivid effects, yet the motion is often physically implausible, object identities drift over time, and it lacks multi-view consistency. These limitations stem from the absence of explicit physical modeling, which makes it difficult to capture complex physical phenomena. In contrast, by integrating thermomechanical simulation with neural scene representations, MeGAS preserves the original photorealistic appearance and produces multi-view consistent, physically plausible thermophysical dynamics. **Please refer to supp. for more edited results on real-world data.**

User Study. We further conduct a user study as shown in Tab. 1 across physical plausibility, photorealism, and multi-view/temporal consistency, supporting our thermomechanical dynamic Gaussian Splatting modeling yields more convincing and reliable melting edits (see **supp.** for full config.).

Table 1: User Study. The rankings assigned by participants to each method are converted to integer scores (best=4, worst=1) and averaged over all videos and raters.

Methods	DGE [8]	AutoVFX [15]	Gen-4.5 [42]	Ours
Physical-Realism ↑	1.538	2.692	2.038	**3.731**
Photo-Realism ↑	2.038	2.154	2.154	**3.654**
Multi-View Consistency ↑	2.154	2.385	1.885	**3.577**

5.2 Evaluation on Extreme Melting Deformation

Datasets and Comparison. We evaluate rendering quality under extreme thermomechanical deformations with a set of Blender-synthetic objects and real-world scenes captured with an iPhone. We compare the following 3DGS-based deformation pipelines: 1) Original 3DGS [20]; 2) PhysGaussian [53], a physics-

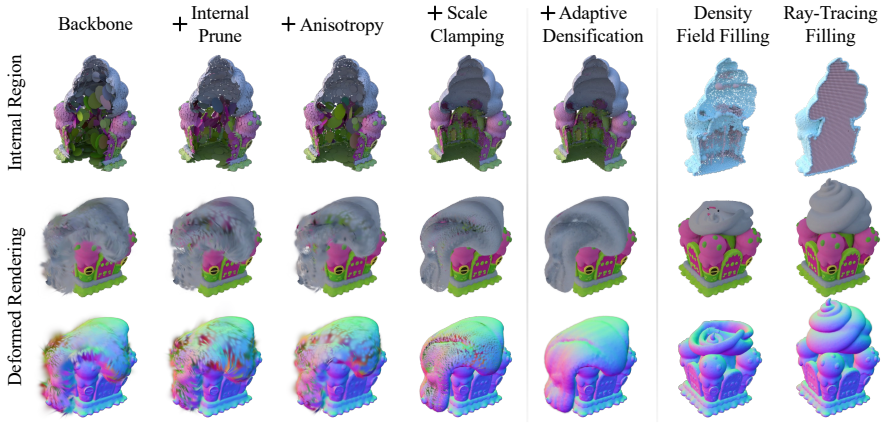

Fig. 9: Ablations on Topology-Adaptive strategy. Our topology-adaptive strategy yields an interior-free, uniformly distributed Gaussian representation with consistent volume, mitigating needle-like artifacts and internal floaters. The adaptive densification fills surface cracking during animation, and our ray-tracing–based internal filling uniformly populates the interior, providing robust volumetric support.

integrated 3DGS with isotropic constrained kernels; 3) VR-DoH [33], which enforces volume-consistency constraints. All baselines are coupled with the same MPM-LBM simulator, and differ only in how Gaussians are regularized, internally filled, or adapted to deformation.

Implementation Details. For each scene, we first reconstruct 3DGS with the respective methods, adopting the internal filling strategies proposed for each baseline (e.g., density-field–based schemes [53]) and employing our ray-tracing–based filling strategy for MeGAS. We then manually place identical heat sources across methods, and evolve the thermal field using the same LBM-based advection–diffusion solver with consistent thermal diffusivity, boundary conditions, and phase-transition parameters. The resulting thermal field drives MPM particles according to phase-aware constitutive models, and Gaussians are updated at each step via deformation-gradient–based transformations. All methods are simulated for the same duration with identical material and solver settings (**see supp. for full configurations and computational cost**).

Results. Fig. 8 visualizes internal cross-sections and dynamic frames under large-scale melting. Baseline methods accumulate internal floaters, produce severe needle-like artifacts, and suffer from surface cracking that exposes underspecified interiors. In contrast, our MeGAS yields an internal-floater-free and volume-consistent Gaussian distribution, with stable, crack-free surfaces refined by our adaptive surface densification. The resulting geometry remains smooth and coherent throughout the melting process, allowing direct extraction of high-quality meshes from the evolving Gaussian representation, as illustrated in Fig. 2.

5.3 Ablation Studies

Topology-Adaptive Strategy. As shown in Fig. 9, we progressively incorporate our topology-adaptive strategy into the backbone model [7]. Internal-Prune

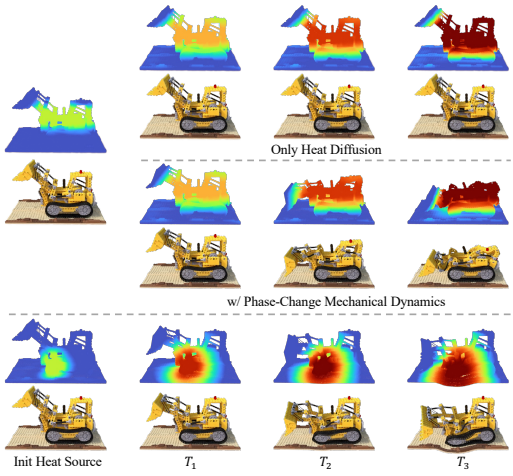

Fig. 10: Heat diffusion with constitutive model switching. Phase-change-driven constitutive switching couples melting to temperature evolution, while different heat-source configurations yield distinct melting patterns.

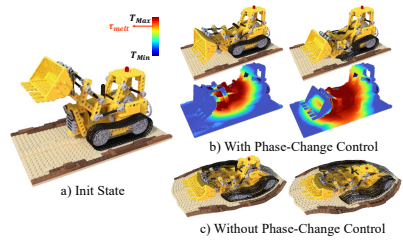

Fig. 11: Ablations on phase-change switching. Without phase change, melting is triggered globally and collapses prematurely.

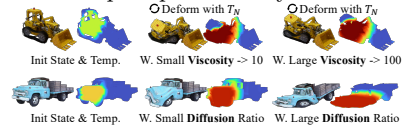

Fig. 12: Controllable melting via parameter adjustment. By modifying thermophysical parameters, users controllably achieve different melting effects.

effectively suppresses internal floaters, while anisotropy loss and scale clamping yield a uniform, volume-consistent Gaussian distribution that avoids needle-like artifacts under large deformation. During animation, adaptive densification compensates for surface cracking and maintains smooth rendering. Since our representation is interior-free, directly applying density-field–based filling [53] leaves the interior hollow and prone to collapse, whereas our ray-tracing filling uniformly populates the volume and provides robust volumetric support.

Phase-Change Switching. As shown in Fig. 11, we demonstrate the necessity of phase-change–driven constitutive switching by comparing an init-melted strategy, where all Gaussians are initialized in the melted state and follow the viscoplastic constitutive model, while we incrementally change each Gaussian's phase state based on the evolving temperature field. Init-melted strategy exhibits premature global slumping, while our temperature-controlled transition yields a more physically plausible melting progression. Moreover, we present visualizations of pure heat diffusion in Fig. 10, where the temperature field progressively propagates from high- to low-temperature regions over time. Building on this diffusion process, our phase-change-aware constitutive switching produces melting dynamics explicitly coupled to temperature evolution. We additionally show that interactive heat-source specification (Fig. 10) and thermophysical parameter tuning (Fig. 12) enable controllable melting with distinct patterns and effects.

5.4 Scalability to Complex Thermomechanical Interactions.

Built upon an MPM-based simulator, our framework supports broader thermomechanical interactions beyond single-object melting. In addition, the cooling-

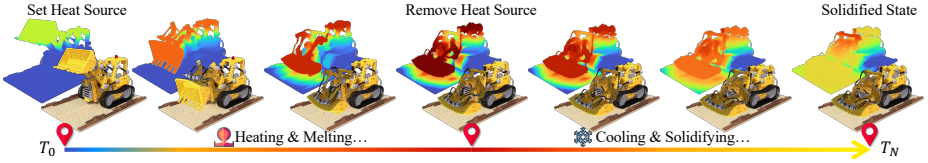

Fig. 13: Bidirectional phase-change editing. Controlled heating induces melting, while subsequent cooling drives solidification, demonstrating consistent thermomechanical evolution across phase transitions and enabling general thermophysical editing.

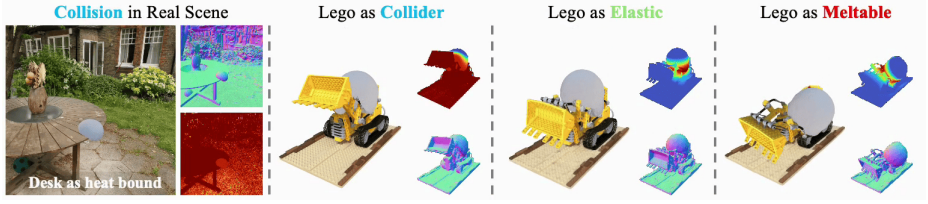

Fig. 14: Scalability to complex scenes. Our method naturally supports multi-object interactions with multi-material and collisions (e.g., an ice cream ball). **Please view the dynamic videos in Adobe Acrobat Reader!**

driven solidification shown in Fig. 13 demonstrates that our framework naturally supports bidirectional phase transitions, extending thermomechanical editing beyond melting. As illustrated in Fig. 14, we further showcase collisions with real-scene environments, interactions with complex geometric colliders and elastic bodies, and heat-transfer interactions among multiple melting objects. Our method preserves the inherent extensibility of MPM while maintaining stable rendering under severe deformations, thereby broadening dynamic neural rendering toward richer thermomechanical dynamics modeling.

6 Conclusion

We have proposed MeGAS, a novel framework that integrates thermomechanical dynamics into 3D Gaussian Splatting. MeGAS jointly models appearance and a controllable thermal field, coupling a heat advection-diffusion solver with temperature-conditioned constitutive switching to realize phase-change–driven thermophysical editing. To handle extreme deformations that induce topology changes and surface cracking, we combine training-time anisotropy regularization and interior-free filtering with an online deformation-stage refinement that we fit local implicit surfaces to seal cracks and yield smooth renderings. To our knowledge, MeGAS provides the first exploration of thermomechanical phase-change dynamics within a neural rendering pipeline, addressing an underexplored yet fundamental dimension of physics integration.

Limitation. We currently do not model material-dependent appearance changes induced by phase transitions (e.g., PBR property variations during melting), as jointly coupling thermomechanics with dynamic material appearance remains highly challenging. A promising direction is to integrate our framework with video diffusion models [6, 30, 50] to enhance the final appearance rendering.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (NSFC) under Grants 62572425 and 624B2132. We thank the anonymous reviewers for their valuable comments and suggestions.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C.T.: Point set surfaces. In: Proceedings Visualization, 2001. VIS'01. pp. 21–29. IEEE (2001)
3. Ashgriz, N., Mostaghimi, J.: An introduction to computational fluid dynamics. Fluid Flow Handbook **1**, 1–49 (2002)
4. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022)
5. Batchelor, G.K.: An Introduction to Fluid Dynamics. Cambridge University Press (2000)
6. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)
7. Chen, D., Li, H., Ye, W., Wang, Y., Xie, W., Zhai, S., Wang, N., Liu, H., Bao, H., Zhang, G.: Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. IEEE Transactions on Visualization and Computer Graphics (2024)
8. Chen, M., Laina, I., Vedaldi, A.: Dge: Direct gaussian 3d editing by consistent multi-view editing. In: European Conference on Computer Vision. pp. 74–92. Springer (2024)
9. Chen, Q., Shu, S., Bai, X.: Thermal3d-gs: Physics-induced 3d gaussians for thermal infrared novel-view synthesis. In: European Conference on Computer Vision. pp. 253–269. Springer (2024)
10. Dai, Q., Ni, X., Shen, Q., Chen, W., Chen, B., Chu, M.: Rainygs: Efficient rain synthesis with physically-based gaussian splatting. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16153–16162 (2025)
11. Feng, Y., Feng, X., Shang, Y., Jiang, Y., Yu, C., Zong, Z., Shao, T., Wu, H., Zhou, K., Jiang, C., et al.: Gaussian splashing: Unified particles for versatile motion synthesis and rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 518–529 (2025)
12. Feng, Y., Shang, Y., Li, X., Shao, T., Jiang, C., Yang, Y.: Pie-nerf: Physics-based interactive elastodynamics with nerf. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4450–4461 (2024)
13. Google DeepMind: Veo 3. https://deepmind.google/models/veo/ (2025), official model page, accessed March 8, 2026
14. Guo, Z., Zheng, C., Shi, B.: Discrete lattice effects on the forcing term in the lattice boltzmann method. Physical Review E **65**(4), 046308 (2002)

15. Hsu, H.Y., Lin, C.H., Zhai, A.J., Xia, H., Wang, S.: Autovfx: Physically realistic video editing from natural language instructions. In: 2025 International Conference on 3D Vision (3DV). pp. 769–780. IEEE (2025)
16. Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., Jiang, C.: A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. ACM Transactions on Graphics **37**(4), 1–14 (2018)
17. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2d gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–11 (2024)
18. Jiang, C., Schroeder, C., Selle, A., Teran, J., Stomakhin, A.: The affine particle-in-cell method. ACM Transactions on Graphics **34**(4), 1–10 (2015)
19. Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., Li, M., Lau, H., Gao, F., Yang, Y., et al.: Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–1 (2024)
20. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
21. Klár, G., Gast, T., Pradhana, A., Fu, C., Schroeder, C., Jiang, C., Teran, J.: Drucker-prager elastoplasticity for sand animation. ACM Transactions on Graphics **35**(4), 1–12 (2016)
22. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4), 1–13 (2017)
23. Kolluri, R.: Provably good moving least squares. ACM Transactions on Algorithms (TALG) **4**(2), 1–25 (2008)
24. Kuaishou Technology: Kling video 2.6 model with simultaneous audio-visual generation. https://ir.kuaishou.com/news-releases/news-release-details/kling-ai-launches-video-26-model-simultaneous-audio-visual/ (2025), official press release, accessed March 8, 2026
25. Lei, Y., Zhao, B., Yang, Z., Li, X., Cheng, T., Peng, H., Zhang, R., Yang, Y., Huang, S., Shen, Y., et al.: Diffwind: Physics-informed differentiable modeling of wind-driven object dynamics. In: International Conference on Learning Representations
26. Li, X., Qiao, Y.L., Chen, P.Y., Jatavallabhula, K.M., Lin, M., Jiang, C., Gan, C.: Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. arXiv preprint arXiv:2303.05512 (2023)
27. Li, Y., Lin, Z.H., Forsyth, D., Huang, J.B., Wang, S.: Climatenerf: Extreme weather synthesis in neural radiance field. In: IEEE/CVF International Conference on Computer Vision. pp. 3227–3238 (2023)
28. Lin, Y., Lin, C., Xu, J., MU, Y.: Omniphysgs: 3d constitutive gaussians for general physics-based dynamics generation. In: International Conference on Learning Representations
29. Lin, Y.Y., Pan, X.Y., Fridovich-Keil, S., Wetzstein, G.: Thermalnerf: Thermal radiance fields. In: 2024 IEEE International Conference on Computational Photography (ICCP). pp. 1–12. IEEE (2024)
30. Liu, S., Ren, Z., Gupta, S., Wang, S.: Physgen: Rigid-body physics-grounded image-to-video generation. In: European Conference on Computer Vision. pp. 360–378. Springer (2024)
31. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: Seminal graphics: pioneering efforts that shaped the field, pp. 347–353 (1998)

32. Lu, R., Chen, H., Zhu, Z., Qin, Y., Lu, M., Zhang, L., Yan, C., Xue, A.: Thermal-gaussian: Thermal 3d gaussian splatting. arXiv preprint arXiv:2409.07200 (2024)
33. Luo, Z., Cui, Z., Luo, S., Chu, M., Li, M.: Vr-doh: Hands-on 3d modeling in virtual reality. ACM Transactions on Graphics **44**(4) (Jul 2025). https://doi.org/10.1145/3731154
34. Macklin, M.: Warp: A high-performance python framework for gpu simulation and graphics (Mar 2022), https://github.com/NVIDIA/warp, NVIDIA GPU Technology Conference (GTC)
35. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
36. Moenne-Loccoz, N., Mirzaei, A., Perel, O., de Lutio, R., Martinez Esturo, J., State, G., Fidler, S., Sharp, N., Gojcic, Z.: 3d gaussian ray tracing: Fast tracing of particle scenes. ACM Transactions on Graphics (2024)
37. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: 2011 10th IEEE international symposium on mixed and augmented reality. pp. 127–136. Ieee (2011)
38. Özer, M., Weiherer, M., Hundhausen, M., Egger, B.: Exploring multi-modal neural scene representations with applications on thermal imaging. In: European Conference on Computer Vision. pp. 82–98. Springer (2024)
39. Oztireli, C., Guennebaud, G., Gross, M.: Feature preserving point set surfaces based on non-linear kernel regression. In: Computer Graphics Forum. vol. 28, pp. 493–501. Wiley (2009)
40. Qiu, R.Z., Yang, G., Zeng, W., Wang, X.: Language-driven physics-based scene synthesis and editing via feature splatting. In: European Conference on Computer Vision (2024)
41. Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F., Zeng, Z., Zhang, H., Li, F., Yang, J., Li, H., Jiang, Q., Zhang, L.: Grounded sam: Assembling open-world models for diverse visual tasks (2024)
42. Runway AI, Inc.: Introducing runway gen–4. https://runwayml.com/research/introducing-runway-gen-4.5 (2025), accessed: 2025-12-1
43. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016)
44. Shen, Q., Tao, N., Dai, Q., Chen, T., Qin, M., Zhang, Y., Chu, M., Chen, W., Chen, B.: Fierygs: In-the-wild fire synthesis with physics-integrated gaussian splatting. In: International Conference on Learning Representations (ICLR) (2026)
45. Shen, S., Shao, T., Zhou, K., Jiang, C., Yang, Y.: Enliveninggs: Active locomotion of 3dgs. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 896–905 (2025)
46. Side Effects Software Inc.: Houdini. https://www.sidefx.com/products/houdini/ (2026), version 21.0, accessed June 22, 2026
47. Succi, S.: The Lattice Boltzmann Equation: For Fluid Dynamics and Beyond. Oxford University Press (2001)
48. Vasile, F., Qiu, R.Z., Natale, L., Wang, X.: Gaussian-augmented physics simulation and identification with complex colliders. In: Advances in Neural Information Processing Systems (2025)
49. Wan AI: Wan 2.6. https://wan.video/introduction/wan2.6 (2026), official model introduction page, accessed March 8, 2026

50. Wan, Team, Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., et al.: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 (2025)
51. Wang, M., Zhang, Y., Xu, W., Ma, R., Zou, C., Morris, D.: Decoupledgaussian: Object-scene decoupling for physics-based interaction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11361–11372 (2025)
52. Wang, Z., Ling, H., Zhang, W., Sun, Y., Sun, Q.: ETGS: Explicit thermodynamics gaussian splatting for dynamic thermal reconstruction. In: International Conference on Learning Representations (2026), https://openreview.net/forum?id=P2Nw2LMkjH
53. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4389–4398 (2024)
54. Yang, K., Liu, Y., Cui, Z., Liu, Y., Zhang, M., Yan, S., Wang, Q.: Ntr-gaussian: Nighttime dynamic thermal reconstruction with 4d gaussian splatting based on thermodynamics. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 691–700 (2025)
55. Ye, T., Wu, Q., Deng, J., Liu, G., Liu, L., Xia, S., Pang, L., Yu, W., Pei, L.: Thermal-nerf: Neural radiance fields from an infrared camera. In: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1046–1053. IEEE (2024)
56. Ying, H., Yin, Y., Zhang, J., Wang, F., Yu, T., Huang, R., Fang, L.: Omniseg3d: Omniversal 3d segmentation via hierarchical contrastive learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20612–20622 (2024)
57. Yue, Y., Smith, B., Batty, C., Zheng, C., Grinspun, E.: Continuum foam: A material point method for shear-dependent flows. ACM Transactions on Graphics **34**(5), 1–20 (2015)
58. Zhong, L., Yu, H.X., Wu, J., Li, Y.: Reconstruction and simulation of elastic objects with spring-mass 3d gaussians. In: European Conference on Computer Vision. pp. 407–423. Springer (2024)
59. Zong, Z., Li, X., Li, M., Chiaramonte, M.M., Matusik, W., Grinspun, E., Carlberg, K., Jiang, C., Chen, P.Y.: Neural stress fields for reduced-order elastoplasticity and fracture. In: SIGGRAPH Asia 2023 Conference Papers. pp. 1–11 (2023)

MeGAS: Thermomechanical Dynamic Gaussian Splatting for Thermophysical Scene Editing - Supplementary Material -

This supplementary material provides detailed descriptions of the training procedure for topology-adaptive Gaussian Splatting (Sec. A), the full formulations and implementation details of the MPM-LBM simulator (Sec. B), the derivation and implementation of the implicit-surface-guided densification module (Sec. C), the computational cost analysis of each stage (Sec. D), the generalization behavior of topology-adaptive rendering (Sec. E), the experimental setup for thermophysical scene editing including baseline prompts, parameter settings, and user-study details (Sec. F), and additional qualitative results on multi-material editing, multi-view consistency, and real-world scenes (Secs. H-J).

A Details for Training Topology-Adaptive GS

We first run COLMAP [43] with the input image sequence to estimate the camera poses and sparse point cloud for reconstruction. Building on the vanilla 3D Gaussian Splatting pipeline [20], we incorporate the planar regularization from [7, 17] to improve geometric fidelity. Specifically, each Gaussian is encouraged to approximate a local plane by penalizing the smallest principal scale $||\min(s_1, s_2, s_3)||_1$, and we adopt the unbiased plane-depth regularization to enforce consistency between the rendered depth and the corresponding surface normals to further align with the underlying surface. On top of the planar regularization, we apply our internal-free uniform Gaussian regularization, introduced in Sec. 4.2.1 of the main paper, to enhance robustness under large deformations. The topology-adaptive 3DGS is optimized for 30k iterations with a joint loss:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{geo}} + \lambda_{\text{aniso}}\mathcal{L}_{\text{aniso}}, \tag{1}$$

where $\mathcal{L}_{\text{rgb}}$ is the photometric loss from 3DGS [20], $\mathcal{L}_{\text{geo}}$ corresponds to the planar constraints from PGSR [7], and $\mathcal{L}_{\text{aniso}}$ is our anisotropy loss. The weight for anisotropy regularization λ_{aniso} is set to 0.1. During training, we perform low-contribution Gaussian pruning every 2,500 iterations with the pruning threshold of τ_{prune}=0.2, and apply scale clamping every 1,000 iterations by truncating the principal scales with an upper bound τ_{scale}=1/128.

Moreover, since we define each Gaussian normal as the direction of its shortest principal axis, the normal is ambiguous up to a sign in the absence of camera. To resolve this ambiguity, we introduce a learnable pseudo normal for each Gaussian and perform a short self-distillation stage after training. Each pseudo normal $\mathbf{n}'_i$ is randomly initialized and optimized using the rasterized Gaussian normal $\mathbf{n}_i$ as a pseudo label, with the loss:

$$\mathcal{L}_{\text{normal}} = \|\mathbf{N}' - \text{stopgrad}(\mathbf{N})\|_1, \tag{2}$$

where $\mathbf{N}'=\sum_i T_i\alpha_i\mathbf{n}'_i$ and $\mathbf{N}=\sum_i T_i\alpha_i\mathbf{n}_i$. We finetune it for 500 iterations and correct the orientation of Gaussian normals by enforcing alignment with the directions of the pseudo normals.

We additionally validate our internal-free uniform Gaussian regularization on the NeRF-Synthetic dataset [35] in terms of rendering quality. As shown in Tab. 1, the regularization substantially reduces internal floaters and yields a more uniform Gaussian distribution, while maintaining comparable PSNR to the vanilla PGSR [7] baseline.

Table 1: Ablation of rendering quality. We progressively add each module on top of the backbone. Our module substantially reduces internal floaters and yields a more uniform Gaussian distribution, while maintaining comparable PSNR. Although the anisotropy regularization slightly degrades PSNR, the subsequent scale clamping recovers the rendering quality.

Module (Progressively Add)	PSNR↑	SSIM↑	LPIPS↓
Backbone [7]	37.33	0.9813	0.031
w/ Internal Pruning	36.76	0.9801	0.035
w/ Anisotropy Reg.	35.81	0.9767	0.045
w/ Scale Clamping	37.24	0.9805	0.031

For outdoor scenes, we first perform scene reconstruction with PGSR and then extract the foreground. Specifically, we employ GroundedSAM [41] with foreground prompts (e.g., "*Truck*", "*Lego Dozer*") to obtain multi-view 2D masks, and subsequently perform 3D foreground segmentation via contrastive learning [56]. The resulting foreground Gaussians are then refined using our internal-free uniform Gaussian regularization. All experiments are conducted on an NVIDIA RTX 4090 24GB GPU.

Fig. 1: We also add real-world finetuning comparisons, where our method adapts GS topology to melting-induced extreme deformations, while maintaining comparable rendering quality.

B Details for MPM-LBM Simulator

Table 2: Material and discretization parameters. We use identical physical parameters across all experiments.

Parameter	Value	Description
E	2.8×10^4	Young's modulus
ν	0.4	Poisson's ratio
g	1.0	Gravity acceleration
ρ	100.0	Density of the solid
η	100.0	Viscosity coefficient
h_{shear}	1.0	Shear ratio for plastic flow
σ_Y	1.0	Yield threshold
T_{melt}	0.7	Thermal phase threshold
Δt	1.0×10^{-4}	Time step size
N_{sub}	100	MPM sub-steps per frame
N_{grid}	100^3	Grid resolution

We employ a hybrid particle-grid discretization for the MPM solver and the Lattice Boltzmann solver for the thermal advection-diffusion field. The MPM background grid and the LBM thermal lattice share the same Cartesian layout and spacing Δx, which allows all operations to be carried out on a common grid without any additional interpolation.

At the particle level (index p), we store the position $\mathbf{x}_p$, velocity $\mathbf{v}_p$, mass m_p, volume V_p^0, elastic deformation gradient $\mathbf{F}_p^E$, affine momentum matrix $\mathbf{C}_p$, temperature T_p and its gradient ∇T_p, and a phase flag indicating whether the material is solid or melted. Grid / lattice nodes (index i) store mass m_i, velocity $\mathbf{v}_i$, and temperature T_i.

For the LBM solver, each lattice node additionally stores the scalar distribution functions $g_{i,k}$ associated with discrete velocities $\mathbf{c}_k$ and weights w_k. Cubic B-spline basis functions are used to define the particle-to-grid and grid-to-particle transfer weights w_{ip} and their gradients ∇w_{ip}. A single time step from t^n to $t^{n+1} = t^n + \Delta t$ consists of particle-to-grid transfer (P2G), grid update, and grid-to-particle transfer (G2P). The explicit algorithm is outlined as follows.

Particle-to-Grid Transfer. We first reset all grid quantities to zero, then accumulate particle mass and momentum via APIC [18]:

$$\begin{aligned} m_i^n &= \sum_p w_{ip}^n m_p, \\ m_i^n \mathbf{v}_i^n &= \sum_p w_{ip}^n m_p \left(\mathbf{v}_p^n + \mathbf{C}_p^n (\mathbf{x}_i - \mathbf{x}_p^n) \right). \end{aligned} \tag{3}$$

For the thermal field we transfer a mass-weighted nodal temperature and its gradient:

$$m_i^n T_i^n = \sum_p w_{ip}^n m_p \left(T_p^n + (\mathbf{x}_i - \mathbf{x}_p^n)^\top \nabla T_p^n \right). \tag{4}$$

After the accumulation, grid velocities and temperatures are normalised by mass for nodes with $m_i^n > 0$

$$T_i^n \leftarrow (m_i^n T_i^n)/m_i^n, \; v_i^n \leftarrow (m_i^n v_i^n)/m_i^n.$$

Grid Update. Mechanical grid velocities are updated under internal and external forces:

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n - \frac{\Delta t}{m_i^n} \sum_p \boldsymbol{\tau}_p^n \nabla w_{ip}^n V_p^0 + \Delta t \, \mathbf{g}, \tag{5}$$

where $\boldsymbol{\tau}_p^n$ is the Kirchhoff stress computed from the elastic deformation gradient $\mathbf{F}_p^{E,n}$ and the current constitutive model (solid or melted), and $\mathbf{g}$ is the gravity (or other external forces).

Thermal boundary conditions are applied to grid nodes associated with user-specified heat sources by constraining their temperatures to the prescribed source values, $T_i^n = T_{\text{source}}$. In practice, T_{source} is implemented as a Neumann-type heating condition, which incrementally increases the boundary temperature according to the imposed heat flux.

The grid temperatures at time t^{n+1} are then updated by performing one step of the LBM solver, denoted by `LBM_Update`.

Thermal Advection-Diffusion Solver. The temperature field on the grid is advanced with a Lattice Boltzmann solver for the advection-diffusion equation (ADE):

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \alpha \nabla^2 T + S, \tag{6}$$

where $\mathbf{u}$ is an advection velocity, α is the thermal diffusivity, and S is a volumetric heat source. In our pure diffusion experiments we simply set $\mathbf{u} = \mathbf{0}$, but the formulation also supports a fully coupled Navier-Stokes LBM, in which case $\mathbf{u}$ is taken from the fluid velocity field described below.

We employ a standard single-relaxation-time (BGK) advection-diffusion LBM. At each lattice node and direction k, the scalar populations $g_{i,k}$ evolve as:

$$g_{i,k}(\mathbf{x}+\mathbf{c}_k\Delta t,\ t+\Delta t) = g_{i,k}(\mathbf{x},t) - \frac{\Delta t}{\tau_T}\left(g_{i,k} - g_{i,k}^{\mathrm{eq}}\right) + \Delta t\, Q_{i,k}, \tag{7}$$

where τ_T is the thermal relaxation time. The macroscopic temperature and discretized source term are recovered from the moments:

$$T_i = \sum_k g_{i,k}, \qquad q_i = \sum_k Q_{i,k}. \tag{8}$$

The equilibrium distribution is chosen analogously to the standard LB fluid equilibrium, but with a single conserved scalar:

$$g_{i,k}^{\mathrm{eq}} = w_k T_i \left[1 + \frac{\mathbf{c}_k\cdot\mathbf{u}_i}{c_s^2} + \frac{(\mathbf{c}_k\cdot\mathbf{u}_i)^2}{2c_s^4} - \frac{\mathbf{u}_i\cdot\mathbf{u}_i}{2c_s^2}\right], \tag{9}$$

where c_s is the lattice sound speed and $\mathbf{u}_i$ is the prescribed advection velocity at node i (for pure diffusion we set $\mathbf{u}_i = \mathbf{0}$). The source term is discretized as:

$$Q_{i,k} = w_k\, q_i \left(1 - \frac{\Delta t}{2\tau_T}\right). \tag{10}$$

A Chapman-Enskog expansion shows that this scheme recovers the ADE with thermal diffusivity:

$$\alpha = c_s^2\left(\tau_T - \frac{\Delta t}{2}\right). \tag{11}$$

If fluid motion is to be resolved, we augment the above temperature solver with an isothermal LBM for the (weakly compressible) Navier-Stokes equations:

$$\partial_t(\rho\mathbf{u}) + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) = -\nabla p + \nabla\cdot(2\rho\nu\mathbf{D}) + \rho\mathbf{g} + \mathbf{F}_{\mathrm{ext}}, \tag{12}$$

where ρ is the density, ν is the kinematic viscosity, $\mathbf{D}$ is the rate-of-strain tensor, and $\mathbf{F}_{\mathrm{ext}}$ collects body forces (e.g. buoyancy).

We introduce an additional set of distribution functions $f_{i,k}$ living on the same lattice with velocities $\mathbf{c}_k$ and weights w_k. Their evolution is given by:

$$f_{i,k}(\mathbf{x}+\mathbf{c}_k\Delta t,\ t+\Delta t) = f_{i,k}(\mathbf{x},t) - \frac{\Delta t}{\tau_f}\left(f_{i,k} - f_{i,k}^{\mathrm{eq}}\right) + \Delta t\, F_{i,k}, \tag{13}$$

where τ_f is the fluid relaxation time and $F_{i,k}$ is a forcing term. The macroscopic density and velocity are recovered as:

$$\rho_i = \sum_k f_{i,k}, \qquad \rho_i\mathbf{u}_i = \sum_k \mathbf{c}_k f_{i,k} + \frac{\Delta t}{2}\,\mathbf{F}_i, \tag{14}$$

with $\mathbf{F}_i$ the total body force at node i. The equilibrium distribution is the standard second-order isothermal form:

$$f_{i,k}^{\mathrm{eq}} = w_k\rho_i \left[1 + \frac{\mathbf{c}_k\cdot\mathbf{u}_i}{c_s^2} + \frac{(\mathbf{c}_k\cdot\mathbf{u}_i)^2}{2c_s^4} - \frac{\mathbf{u}_i\cdot\mathbf{u}_i}{2c_s^2}\right]. \tag{15}$$

Under this choice, a Chapman-Enskog analysis shows that the LB scheme recovers the Navier-Stokes equations in the low-Mach-number limit, with kinematic viscosity:

$$\nu = c_s^2\left(\tau_f - \frac{\Delta t}{2}\right). \tag{16}$$

Body forces are incorporated via a standard Guo-type forcing term:

$$F_{i,k} = w_k\left[\frac{\mathbf{c}_k - \mathbf{u}_i}{c_s^2} + \frac{(\mathbf{c}_k\cdot\mathbf{u}_i)}{c_s^4}\,\mathbf{c}_k\right]\cdot\mathbf{F}_i. \tag{17}$$

For thermally driven flows, we use a Boussinesq approximation and define:

$$\mathbf{F}_i = \rho_0 \beta \left(T_i - T_{\text{ref}}\right) \mathbf{g} + \mathbf{F}_{\text{other}}, \tag{18}$$

where ρ_0 is a reference density, β is the thermal expansion coefficient, T_{ref} is a reference temperature, and $\mathbf{F}_{\text{other}}$ collects any additional body forces. The buoyancy term couples the temperature LBM and the Navier-Stokes LBM: the temperature field T_i feeds into the body force $\mathbf{F}_i$, while the fluid velocity $\mathbf{u}_i$ is used as the advection velocity in the thermal equilibrium $g_{i,k}^{\text{eq}}$ in Eq. (7). In the special case where fluid motion is neglected, we simply disable the Navier-Stokes solver and set $\mathbf{u}_i = \mathbf{0}$ everywhere, resulting in a purely diffusive temperature evolution.

Grid-to-Particle Transfer and Constitutive Update. After the grid velocities and temperatures have been advanced, we interpolate them back to particles:

$$\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \, \mathbf{v}_i^{n+1}, \tag{19}$$

$$\nabla \mathbf{v}_p^{n+1} = \sum_i \mathbf{v}_i^{n+1} (\nabla w_{ip}^n)^\top \tag{20}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \, \mathbf{v}_p^{n+1}, \tag{21}$$

$$\mathbf{C}_p^{n+1} = \frac{4}{\Delta x^2} \sum_i w_{ip}^n \, \mathbf{v}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^\top, \tag{22}$$

$$T_p^{n+1} = \sum_i w_{ip}^n \, T_i^{n+1}, \tag{23}$$

$$\nabla T_p^{n+1} = \sum_i T_i^{n+1} \left(\nabla w_{ip}^n\right)^\top. \tag{24}$$

The elastic deformation gradient is updated as:

$$\mathbf{F}_p^{E,n+1} = \left(\mathbf{I} + \Delta t \, \nabla \mathbf{v}_p^{n+1}\right) \mathbf{F}_p^{E,n}. \tag{25}$$

Finally, we perform **temperature-driven phase switching**: if the particle temperature exceeds the melting threshold τ_{melt}, we switch its constitutive model from solid to melted state, i.e. from StVK elasticity [21] to a Herschel-Bulkley viscoplastic model [57]; otherwise we keep the solid model:

$$(\tau_p^{n+1}, \mathbf{F}_p^{E,n+1}) = \begin{cases} \text{solidModel}\left(\mathbf{F}_p^{E,n+1}\right), & T_p^{n+1} \le T_{\text{melt}}, \\ \text{meltedModel}\left(\mathbf{F}_p^{E,n+1}\right), & T_p^{n+1} > T_{\text{melt}}. \end{cases} \tag{26}$$

For the StVK elasticity model, we define the Kirchhoff stress τ as:

$$\boldsymbol{\tau} = \mathbf{U}\big(2\mu\boldsymbol{\epsilon} + \lambda \text{tr}(\boldsymbol{\epsilon})\mathbf{I}\big)\mathbf{V}^T, \tag{27}$$

where $\mathbf{F}^E = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^\top$ is the SVD decomposition of $\mathbf{F}$ and $\boldsymbol{\epsilon} = \log(\boldsymbol{\Sigma})$ denotes the Hencky strain. Here μ and λ are the Lamé parameters.

For the Herschel-Bulkley viscoplastic model, we first compute the deviatoric part of the trial Kirchhoff stress:

$$\mathbf{s}_{n+1}^{\text{pre}} = \mu \, \text{dev}[\tau_{n+1}], \qquad s_{n+1}^{\text{pre}} = \left\|\mathbf{s}_{n+1}^{\text{pre}}\right\|_F, \tag{28}$$

where dev$(\cdot)$ denotes the deviatoric part and $\|\cdot\|_F$ the Frobenius norm. If the von Mises equivalent stress satisfies:

$$s_{n+1}^{\text{pre}} - \sqrt{\frac{2}{3}}\,\sigma_Y \le 0, \tag{29}$$

with σ_Y the yield stress, the response remains elastic and the trial state is accepted. Otherwise, plastic flow is activated and we perform a semi-implicit return-mapping step. Define $\tilde{\mu} = \frac{\mu}{3}\,\text{Tr}(\Sigma^2)$, we update the s_{n+1} explicitly as:

$$\eta^{1/h}\left(s_{n+1} - s_{n+1}^{\text{pre}}\right) + 2\tilde{\mu}\,\Delta t\left(s_{n+1} - \sqrt{\frac{2}{3}}\,\sigma_Y\right)^{1/h} = 0, \tag{30}$$

where η is the viscosity parameter and h is the shear rate. We obtain s_{n+1} via bisection. The deviatoric stress is then projected back onto the yield surface as:

$$\hat{\mathbf{s}}_{n+1} = \frac{s_{n+1}}{s_{n+1}^{\text{pre}}}\,\mathbf{s}_{n+1}^{\text{pre}}, \tag{31}$$

and we update the Kirchhoff stress with $\hat{\mathbf{s}}_{n+1}$ as:

$$\boldsymbol{\tau} = \mathbf{U}\,\exp\!\left(\frac{\hat{s}}{2\mu}\,\text{dev}[\boldsymbol{\epsilon}] + \frac{1}{3}\,\text{Tr}(\boldsymbol{\epsilon})\,\mathbf{I}\right)\mathbf{V}^T. \tag{32}$$

C Details for Densification with Implicit Surface Guidance

We provide the full formulation and implementation details for the implicit-surface-guided adaptive densification (Sec. 4.2.2).

Define $\mathcal{G}_s = \{G_i^s = (\mathbf{x}_i^s, \mathbf{R}_i^s, \mathbf{S}_i^s, \mathbf{o}_i^s, \mathbf{c}_i^s)\}$ the Gaussians that lie on the reconstructed object surface in the rest state, and $\Phi_t(\cdot)$ is the deformation mapping at time step t obtained from the MPM simulation. After deformation, we obtain the deformed surface Gaussians:

$$\tilde{\mathcal{G}}_s^{\,t} = \Phi_t(\mathcal{G}_s) = \left\{\tilde{G}_i^s = \left(\tilde{\mathbf{x}}_i^s, \tilde{\mathbf{R}}_i^s, \tilde{\mathbf{S}}_i^s, \tilde{\mathbf{o}}_i^s, \tilde{\mathbf{c}}_i^s\right)\right\}, \tag{33}$$

and associate to each $\tilde{G}_i^s$ a surface normal $\tilde{\mathbf{n}}_i^s$ given by the shortest axis of $\tilde{\mathbf{S}}_i^s$.

Surface Detection. At each rendering step, we maintain two sets of primitives: (i) a set of surfel-like surface Gaussians $\tilde{\mathcal{G}}_s^{\,t}$, and (ii) a set of volumetric internal particles produced by the ray-tracing-based filling scheme. Under extreme deformations, the surface Gaussians separate and expose cracks in the reconstructed surface, thus our goal is to reuse internal particles that lie close to the visible boundary to fill these newly exposed gaps. To this end, we first perform a surface detection that identifies which internal particles can be regarded as surface samples and are thus eligible for subsequent densification.

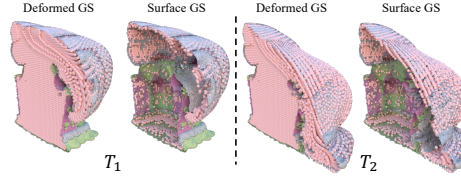

Fig. 2: Surface detection. Our surface detection effectively filters the subset of internal particles, which are treated as candidates to be converted into new surface Gaussians for crack sealing. The internal particles are highlighted in pink for visualization.

We distribute a set of virtual cameras on an object-centric viewing sphere to estimate visibility under deformation. We sample camera azimuths every 45° and elevations every 25°, covering the full 360° around the object. Internal particles $\mathbf{p}_j^t$ are initialized as isotropic Gaussian splats G_j^p with opacity $\mathbf{o}_j = 1$ and radius r_j:

$$r_j = \kappa \, \min\left\{ \min_i \|\mathbf{p}_j^t - \mathbf{x}_i^s\|, \; \min_{k \neq j} \|\mathbf{p}_j^t - \mathbf{p}_k^t\| \right\}, \tag{34}$$

where $\kappa \in (0, 1)$ is a shrinkage ratio, we set $\kappa = 0.5$.

We traverse the virtual cameras, rasterize the Gaussian splats $\{G_p, \; G_s\}$, and record the splat G_i with maximal alpha contribution along the ray for each pixel. The union of these front-most splats over all views yields the set of visible internal particles, which we treat as candidates to be converted into new surface Gaussians for crack sealing. As shown in Fig. 2, our surface detection strategy reliably tracks the evolving surface throughout large deformations.

Implicit Surface from Deformed Gaussians. Given the deformed surface Gaussians $\tilde{\mathcal{G}}_s^t$, we follow the implicit moving least squares (IMLS) formulation for point set surfaces [2, 23, 39] and define an implicit function:

$$f(\mathbf{x}) = \frac{\sum_i G_i^s(\mathbf{x}) \, (\tilde{\mathbf{n}}_i^s)^\top (\mathbf{x} - \tilde{\mathbf{x}}_i^s)}{\sum_i G_i^s(\mathbf{x})}. \tag{35}$$

The zero level set $\mathcal{S}_{\text{IMLS}} = \{\mathbf{x} \mid f(\mathbf{x}) = 0\}$ defines an approximation of the deformed surface. For the spatial weighting kernel, we directly reuse the Gaussian kernel associated with each splat G_i^s:

$$G_i^s(\mathbf{x}) = \begin{cases} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_i^s)^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{x}_i^s)}, & \|\mathbf{x} - \mathbf{x}_i^s\|_2 < h, \\ 0, & \|\mathbf{x} - \mathbf{x}_i^s\|_2 \geq h, \end{cases} \tag{36}$$

where $\boldsymbol{\Sigma}$ denotes the covariance of the Gaussian and h is an adaptive influence radius. For a query point $\mathbf{x}$, we estimate h from the local density of Gaussians as $h(\mathbf{x}) = \eta \, \bar{d}(\mathbf{x})$, where $\bar{d}(\mathbf{x})$ is the average distance to the k nearest surface Gaussians in $\tilde{\mathcal{G}}_s^t$ (we use $k = 32$) and η is a constant scaling factor (we use $\eta = 1.5$).

We then project each candidate internal particle $\mathbf{p}_j^t$, detected by the surface selection stage described above, onto the implicit surface $\mathcal{S}_{\text{IMLS}}$. Let $\mathbf{x}_j^{(0)} = \mathbf{p}_j^t$ be the initialization. We perform a small, fixed number L of Newton-style iterations:

1. Evaluate $f(\mathbf{x}_j^{(\ell)})$.
2. Approximate the gradient of f at $\mathbf{x}_j^{(\ell)}$ by a locally weighted average of normals:

$$\nabla f(\mathbf{x}_j^{(\ell)}) \approx \frac{\sum_i G_i^s(\mathbf{x}_j^{(\ell)}) \, \tilde{\mathbf{n}}_i^s}{\sum_i G_i^s(\mathbf{x}_j^{(\ell)})}. \tag{37}$$

3. Normalize the gradient to obtain the local surface normal:

$$\mathbf{n}_j^{(\ell)} = \frac{\nabla f(\mathbf{x}_j^{(\ell)})}{\left\| \nabla f(\mathbf{x}_j^{(\ell)}) \right\|}. \tag{38}$$

4. Update the position along the normal direction:

$$\mathbf{x}_j^{(\ell+1)} = \mathbf{x}_j^{(\ell)} - f(\mathbf{x}_j^{(\ell)})\,\mathbf{n}_j^{(\ell)}. \tag{39}$$

We terminate the iterations if $|f(\mathbf{x}_j^{(\ell)})| < \varepsilon$ (we use $\varepsilon = 10^{-5}$) or when ℓ reaches $L_{\max}$ (we use $L_{\max} = 5$). The final projected position and normal are denoted by $\mathbf{x}_j^\star$ and $\mathbf{n}_j$, respectively.

Gaussian Kernel Refitting. Given $(\mathbf{x}_j^\star, \mathbf{n}_j, r_j)$, we assign a new Gaussian kernel G_j^p to the crack-filling particle:

- **Position.** The Gaussian center is set to the projected position, $\boldsymbol{\mu}_j = \mathbf{x}_j^\star$.
- **Orientation.** We construct a rotation matrix $\mathbf{R}_j$ whose normal axis is aligned with the surface normal $\mathbf{n}_j$.
- **Scale.** We construct the scale matrix to represent an isotropic disk in the tangent plane, i.e., $\mathbf{S}_j = \mathrm{diag}(r_j, r_j, 0)$.
- **Appearance.** The appearance parameter $\mathbf{c}_j$ is computed as the average of the K nearest surface Gaussians in $\tilde{\mathcal{G}}_s^t$:

$$\mathbf{c}_j = \frac{1}{K} \sum_{i \in \mathcal{N}_K(\mathbf{x}_j^\star)} \tilde{\mathbf{c}}_i^s, \tag{40}$$

  where $\mathcal{N}_K(\mathbf{x}_j^\star)$ denotes the indices of the K nearest neighbors in position space.
- **Opacity.** We initialize the opacity $o_j = 1$.

The resulting Gaussians $\{G_j^p\}$ are finally merged with the existing deformed surface Gaussians to produce the densified representation used for rendering.

We implement the above operation in CUDA, making its computational overhead negligible in practice. Leveraging the IMLS formulation, we further reinterpret the Gaussian Splatting as a signed distance field and apply the Marching Cubes [31] to extract high-quality meshes. As illustrated in Fig. 3, we compare our IMLS-based GS-to-mesh pipeline with a TSDF-fusion-based [37] GS-to-mesh baseline, and our method consistently preserves finer geometric details.

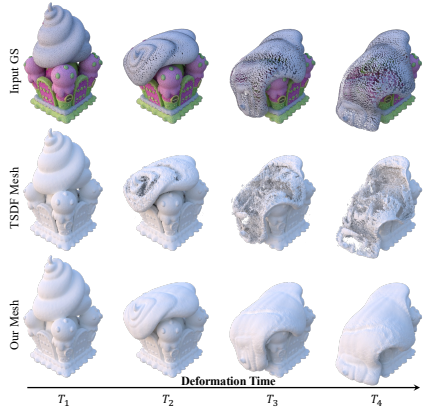

Fig. 3: Comparison of our IMLS GS-to-Mesh with TSDF-based baseline. Compared with meshes obtained via TSDFusion, our approach, which combines Gaussian Splatting with IMLS, produces temporally more continuous and smoother surfaces while simultaneously preserving fine geometric details.

D Computational Cost of Each Stage

Under the parameter settings in supp.Tab. 2, we show the detailed per-scene costs with runtime and peak GPU memory of each stage on an RTX 4070 12GB

Scene\Module	#Splats (+ #Filling)	Internal Filling	MPM + LBM (substep)	IMLS Update	GS Rendering
Candy house (*Synthetic*)	154k (+103k)	475 ms / 0.4 GB	6 ms / 0.6 GB	375 ms / 0.9 GB	3 ms / 0.9 GB
Lego (*Synthetic*)	213k (+13k)	231 ms / 0.5 GB	9 ms / 0.5 GB	419 ms / 0.8 GB	4 ms / 0.8 GB
Toy bulldozer (*Synthetic*)	52k (+47k)	206 ms / 0.3 GB	8 ms / 0.4 GB	175 ms / 0.5 GB	2.0 ms / 0.5 GB
Toy Sculpture (*Real*)	683k (+70k)	662 ms / 1.3 GB	13 ms / 1.2 GB	1084 ms / 2.0 GB	7 ms / 1.9 GB
Kitchen bulldozer (*Real*)	175k (+25k)	288 ms / 0.4 GB	9 ms / 0.5 GB	358 ms / 0.7 GB	3 ms / 0.8 GB
Truck (*Real*)	551k (+21k)	441 ms / 1.1 GB	11 ms / 1.0 GB	851 ms / 1.6 GB	9 ms / 1.5 GB

Table 3: Computational costs.

GPU. In particular, the *MPM+LBM* and *IMLS update* stages are implemented using our custom Warp [34] CUDA kernels. The reported costs are measured per invocation of each stage. Note for real-world scenes, we only consider the segmented foreground object as mentioned in section. A.

E Generalization of Topology-Adaptive Rendering

Our proposed Topology-Adaptive Gaussian Rendering enhances the robustness of Gaussian splats by stabilizing splat geometry under extreme deformation, producing smoother surfaces, and naturally generalizing to other MPM-driven deformations such as elastoplasticity. As shown in Fig. 4, under extreme physical parameter settings, naïve 3DGS breaks down with object fracture and severe needle-like artifacts. In contrast, our method maintains physically plausible motion throughout the deformation process. **Please view the dynamic video in Fig. 4 in Adobe Acrobat Reader.**

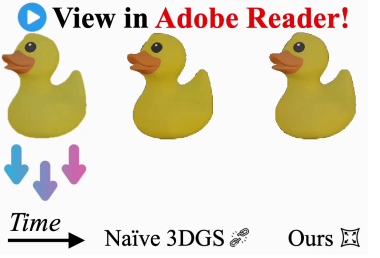

Fig. 4: Generalization to Other MPM-Driven Deformations and Improved Robustness

F Details for Thermophysical Scene Editing

We compare MeGAS with three representative editing pipelines: DGE [8], a text-driven diffusion-based editing method for 3DGS; AutoVFX [15], which leverages an LLM model [1] to generate scripts for external physics engines to perform mesh-based melting and composite the results back into the scene; Runway Gen-4.5 [42], a leading commercial image-to-video generation model that augments a single input image with text-guided dynamic effects.

For DGE, we use the prompt *"Make the (Truck / Sculpture / Lego Dozer) look melted"* with a guidance scale of 15, and optimize the 3DGS model for 15k iterations. Since DGE utilizes an image-level diffusion as prior, it can only perform style editing of static scenes and does not produce explicit dynamic motion. Moreover, its ability to faithfully capture melting, which is inherently a physics-driven phenomenon, is limited and often results in only superficial, texture-level changes. **For AutoVFX**, we use the prompt *"Melt the (Truck / Sculpture / Lego Dozer)"*. AutoVFX first reconstructs a mesh representation of the scene,

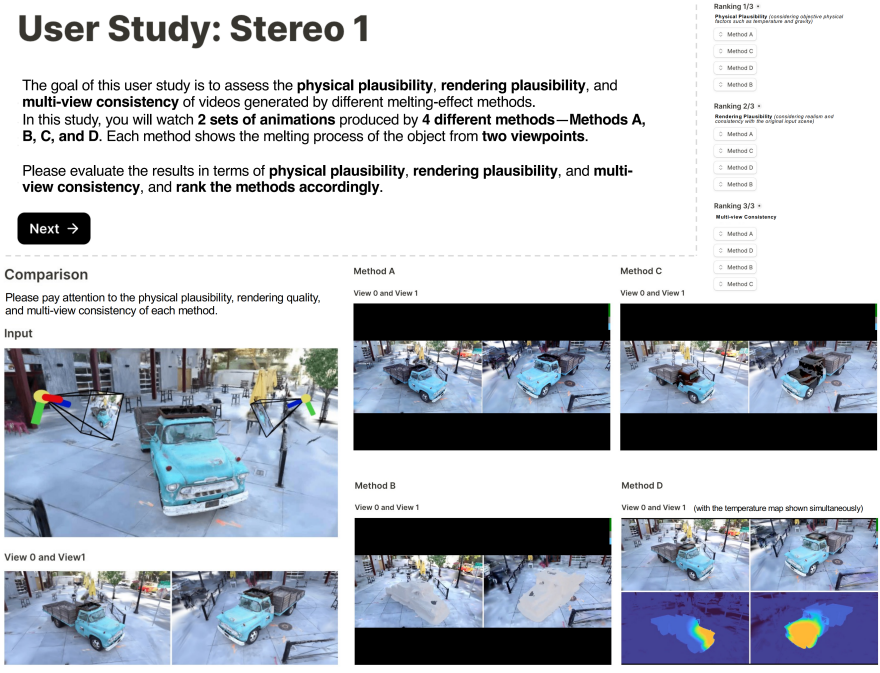

Fig. 5: The user evaluation interface. We rearranged the layout of the interface to present it more clearly in the paper.

then uses GPT-based scripting to automatically segment the foreground object and invoke Blender's FLIP fluid simulation to realize a melting effect. While this pipeline can generate physically plausible deformations on the mesh, it incurs substantial computational overhead and does not explicitly enforce appearance consistency between the simulated melting result and the original scene, leading to noticeable appearance discrepancies in the final composites. **For Runway Gen-4.5**, we input per-view static renderings of the reconstructed scene as image conditions and use the prompt *"The camera keeps still, and over time the (Truck / Sculpture / Lego Dozer) begins to melt"*. Runway Gen-4.5 produces visually vivid dynamic effects, however, the generated motion is not constrained by physical laws and often violates basic physical plausibility. In addition, the model does not preserve the original material appearance of the object consistently, and each viewpoint is edited independently, which leads to temporal and multi-view inconsistencies across the different camera trajectories.

User Study. We recruited 27 participants, and each participant was asked to evaluate 24 videos generated by different methods and from different viewpoints, and to rank the results according to their personal preference. We present the user evaluation interface with one representative example as shown in Fig. 5.

Additional SOTA Video Generation Comparisons. We further compare our method against additional state-of-the-art video generation models, including Veo 3 [13], Wan 2.6 [49] and Kling 2.6 [24], to show that, for highly physics-

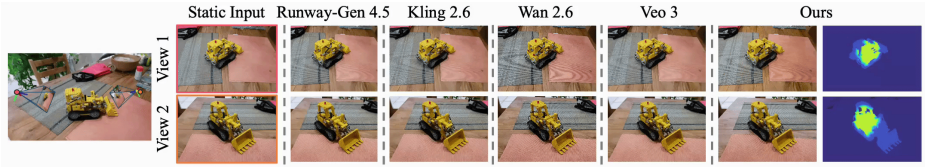

Fig. 6: Comparison with extra video generation models. Please view the dynamic videos in Adobe Acrobat Reader.

grounded animations, although all of them produce vivid visual effects, even recent state-of-the-art video generation models still struggle with controllability, physical plausibility, and multi-view consistency (**please view the dynamic videos in Fig. 6 in Adobe Acrobat Reader**). For specific physical phenomena such as melting, coupling with explicit physics simulators remains essential. This is also supported by prior works, where explicit simulation has been widely adopted as an effective paradigm for modeling specialized physical phenomena, such as in RainyGS [10], FieryGS [44], GaussianSplashing [11], and EnliveningGS [45], enabling data-free, controllable, and physically plausible animations.

G Quantitative Evaluation against Simulation Platform

We add a quantitative evaluation against independent mesh-based thermomechanical simulation references generated by Houdini [46], a professional physical simulation platform, under the same heat-source and thermal-boundary setup as our method, these simulations provide an independent physics-based reference. The temporally averaged surface accuracy shows MeGAS aligns well with the reference while preserving photorealism.

Method	Chamfer Distance ↓	F-score ↑	$\text{Depth}_{\text{RMSE}}$ ↓	Silhouette IoU ↑
PhysGaussian	0.089	0.08	0.130	0.435
Ours	**0.037**	**0.23**	**0.065**	**0.789**

H Handling Different Materials

We further consider multi-material thermophysical editing by performing part-level segmentation and assigning material-specific constitutive models to different object parts. As shown in Fig. 7, object components such as buttons and branches are specified as non-meltable materials (e.g., elastic), while the snow regions remain thermally deformable, producing distinct deformation behaviors across materials and demonstrating that MeGAS can be naturally extended to handle heterogeneous material compositions.

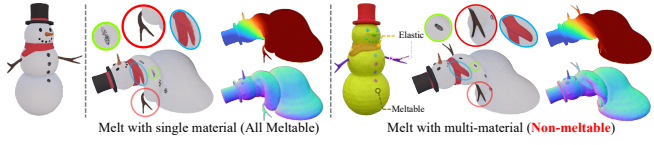

Fig. 7: Multi-material melting.

I Additional Multi-view Results

We provide additional multi-view rendering results for Section 5.2 to further demonstrate cross-view consistency under large deformations as shown in Fig. 8. **Please view the dynamic videos in Adobe Acrobat Reader.**

Fig. 8: Additional multi-view rendering results.

J Additional Results on Real-World Scenes

As shown in Fig. 9, we visualize the thermomechanical dynamics of our method under heat-diffusion control. As the temperature field propagates through the object, phase-change-aware control triggers constitutive model switching, yielding realistic melting-style scene editing. We illustrate the deformation process on real-world scenes, with rendered RGB images, rendered temperature field, and deformation geometries (rendered normals and meshes) at different viewpoints and time steps. Our method produces physically plausible thermomechanical dynamics while maintaining strong temporal and multi-view consistency.

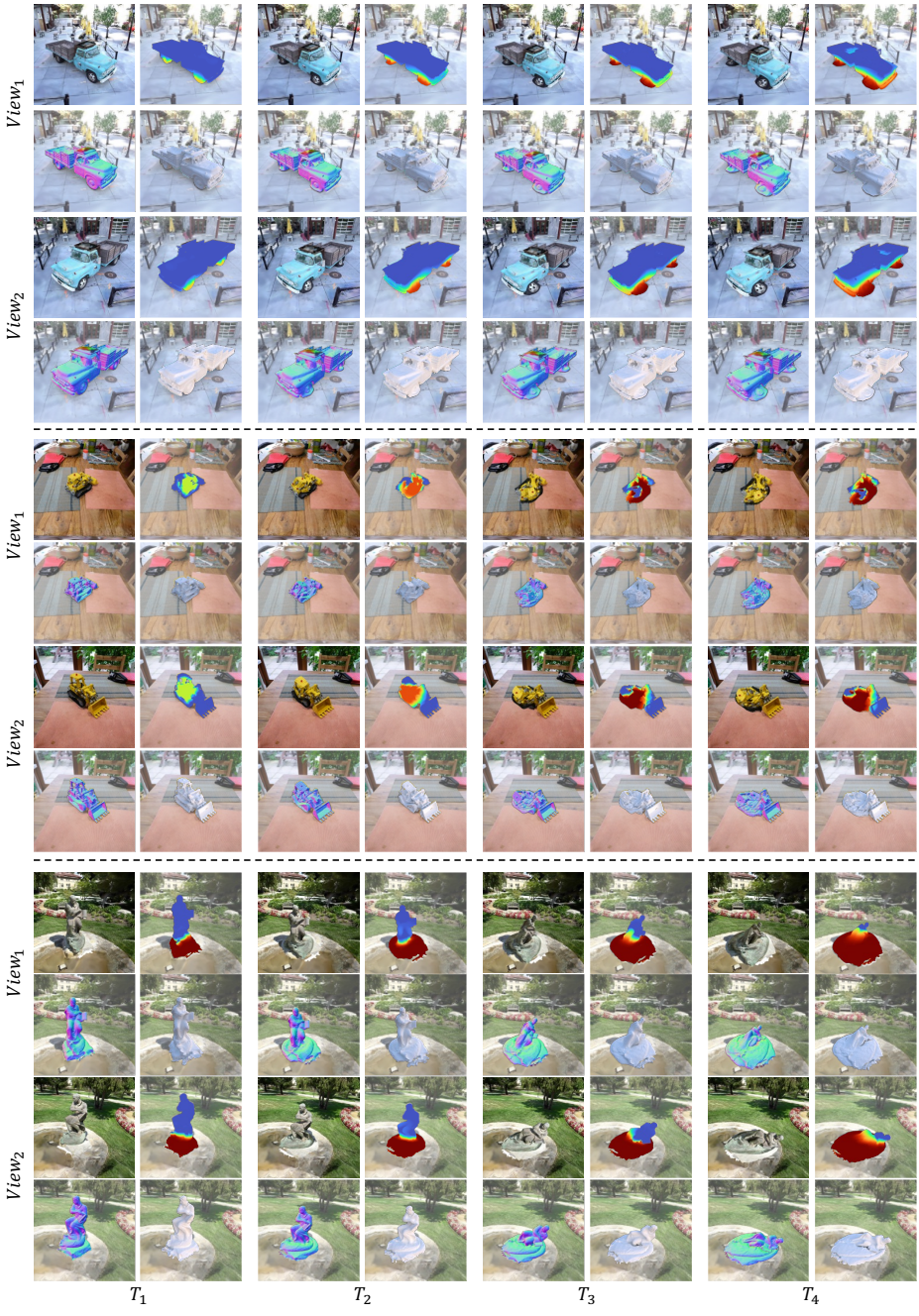

Fig. 9: Our thermophysical scene editing. We illustrate the deformation process on real-world scenes, with rendered RGB images, rendered temperature field, and deformation geometries (rendered normals and meshes) at different viewpoints and time steps.