EfM: Simulation-Free Elastic Parameter Estimation from Observed Motion

Anonymous authors
Paper under double-blind review

Abstract

Reconstructing an object's 3D geometry and appearance captures how it looks, but its material properties are needed to simulate how it responds to new forces. Existing video-based material estimators either rely on costly inverse simulation or use feed-forward predictors whose material estimates are not explicitly constrained to satisfy the observed dynamics. We present **EfM** (**E**lasticity **f**rom **M**otion), which directly recovers the elastic parameters of freely moving objects from multi-view video without running a simulator. EfM makes this direct solve possible with two components: an F-GNN that infers unobserved volumetric deformation from visible surface motion, and a space-time weak formulation that avoids explicit acceleration estimation from observed surface trajectories. Together, they reduce material estimation to a single linear solve for Young's modulus and Poisson's ratio. On PAC-NeRF and Spring-Gaus-Diverse, EfM reduces Young's modulus error by $7.5\times$ and $3.3\times$ over the best baseline, respectively, predicts unseen future motion most accurately, and fits a sequence in 0.74 s, three orders of magnitude faster than inverse simulation.

1 Introduction

Recent advances in 3D vision have enabled remarkably accurate reconstruction of an object's geometry and appearance (Mildenhall et al., 2020; Kerbl et al., 2023), but recovering its physical properties from observation remains underexplored. Knowing how stiff or compressible an object is reveals physical properties that appearance alone cannot capture. Unlike a reconstruction that captures observed geometry and appearance, a material-aware model can be simulated under new forces, as required for robot manipulation, content creation, and digital twins (Jiang et al., 2025; Xie et al., 2024). Motivated by this need, a growing body of work has explored material estimation from video (Li et al., 2023; Zhong et al., 2024; Cai et al., 2024; Zhang et al., 2024; Chen et al., 2025; Le et al., 2025; Lv et al., 2026; Wang et al., 2026).

For common elastic models, the governing equation is linear in the material parameters, suggesting a direct route to their estimation. Sag-free initialization (Hsu et al., 2022; 2023) does exactly this in a single linear solve, assuming a simulation asset in static equilibrium under gravity. Such a direct solve, however, assumes access to the object's full volumetric state, whereas video provides only partial surface observations of a moving object. To handle video observations, most existing methods instead rely on inverse simulation (Li et al., 2023; Zhong et al., 2024; Chen et al., 2025), repeatedly simulating the object and back-propagating the discrepancy with the observed motion through thousands of simulation steps. This makes fitting computationally expensive and sensitive to initialization. To avoid this cost, feed-forward methods predict the material in about a second (Lv et al., 2026; Le et al., 2025; Wang et al., 2026), but their predictions are not explicitly constrained by the observed dynamics. Extending the direct solve to video could therefore provide fast material estimation without costly per-scene inverse simulation or learning a direct mapping from observations to material parameters (Table 1).

To address this, we introduce **EfM** (**E**lasticity **f**rom **M**otion), a framework that extends direct equation-based material estimation to freely moving objects observed in multi-view video. EfM estimates Young's modulus E and Poisson's ratio ν in a single linear solve, without running a simulator. Applying the direct solve to video requires two quantities that are not directly observed: the interior deformation and the acceleration. EfM predicts the interior deformation with the F-GNN from visible surface motion, and avoids explicit acceleration estimation using a space-time weak form (Pierron & Grédiac, 2012; Messenger & Bortz, 2021). Together, these components allow the governing equation to determine the material directly from the observed dynamics.

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

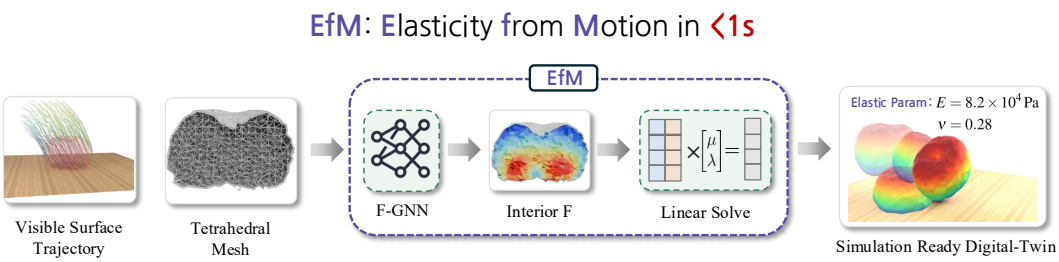


Figure 1: **EfM estimates the elastic parameters of an object from its observed motion in under a second.** Given the trajectory of the visible surface and a tetrahedral mesh of the object, EfM predicts the deformation gradient $\mathbf{F}$ of every element with the F-GNN and recovers the material in a single linear solve, without running a simulator. The estimated Young's modulus E and Poisson's ratio ν make the object ready to simulate.

Table 1: **Material-estimation paradigms.** *Surface only* means the method needs only the observed surface and not the state of the interior, *Tied to motion* means the estimate is constrained by the observed dynamics, and *No init.* means the result does not depend on an initialization. Prior equation-based methods require the full state of a simulation asset and solve the static equation. Only EfM satisfies all three.

Paradigm	Surface only	Tied to motion	No init.	Cost (s)
Inverse simulation (Sec. 2.1)	✓	✓	–	$\sim 10^3$
Feed-forward prediction (Sec. 2.2)	✓	–	✓	<1
Equation-based estimation (Sec. 2.3)	–	–	✓	<1
EfM (ours) (Sec. 3)	✓	✓	✓	**0.74**

We evaluate EfM on PAC-NeRF (Li et al., 2023) and Spring-Gaus-Diverse, a new benchmark with a wider range of materials, against Vid2Sim (Chen et al., 2025), Spring-Gaus (Zhong et al., 2024), ReconPhys (Wang et al., 2026), and PhysGM (Lv et al., 2026). EfM reduces Young's modulus error by $7.5\times$ and $3.3\times$, respectively, predicts unseen future motion most accurately, and fits each sequence in 0.74 s, three orders of magnitude faster than inverse simulation.

Our contributions can be summarized as follows.

- We present EfM, which directly solves the governing equation to estimate the elastic parameters of a freely moving object from multi-view video. To our knowledge, EfM is the first direct equation-based material estimator for freely moving objects in ordinary video.
- We introduce the F-GNN to predict the unobserved interior deformation from visible surface motion, providing the deformation gradient of every volumetric element without taking the material parameters as input.
- We formulate the governing equation in a space-time weak form that avoids differentiating the observed trajectories, enabling direct material estimation at ordinary video frame rates.

2 Related Work

2.1 Inverse Simulation-Based Material Estimation

Most existing methods estimate material parameters from video by inverse simulation (Wang et al., 2015; Jatavallabhula et al., 2021; Qiao et al., 2022; Kaneko, 2024). They run a differentiable simulator (Hu et al., 2020) and update the parameters by back-propagating the difference from the video through the simulation. PAC-NeRF (Li et al., 2023) couples a differentiable material point method with a neural radiance field, and GIC (Cai et al., 2024) and PhysTwin (Jiang et al., 2025) apply the same fit to Gaussian-based geometry and RGB-D video, respectively, whereas video-diffusion-based works replace the observed video by the prior of a video diffusion model (Zhang et al., 2024; Liu et al., 2024; Huang et al., 2025; Lin et al., 2025). Spring-Gaus (Zhong et al., 2024) represents the object as spring-mass 3D Gaussians and optimizes the spring stiffness against multi-view video, and Vid2Sim (Chen et al., 2025) refines a feed-forward prediction of (E, ν) by back-propagating a rendering loss through a reduced continuum simulator. While inverse simulation ties the estimate

to the observed motion, every update back-propagates through the $\sim 10^3$ simulation steps of the sequence, so fitting one sequence takes tens of minutes. The objective is also non-convex in the material, so the estimate depends on the initialization. EfM keeps the tie to the observed motion but solves the governing equation for the material directly, so that it needs no simulation or initialization and fits a sequence in under a second.

2.2 Feed-Forward Material Estimation

Another line of work skips the per-scene optimization and predicts the material with a pretrained network in a single forward pass, in about a second. PhysGM (Lv et al., 2026) predicts (E, ν) from one or a few still images, and Pixie (Le et al., 2025) and related works (Zhai et al., 2024; Dagli et al., 2026) predict a material field over a reconstructed 3D scene. ReconPhys (Wang et al., 2026) reads a monocular video and predicts the stiffness, friction, mass and damping of a spring-mass model. While these methods are fast, nothing requires their estimate to agree with the observed motion. PhysGM and Pixie never see the motion, so two objects that look alike but move differently receive the same material, and ReconPhys produces its estimate with a network rather than from the physics of the video. EfM runs at a comparable cost (Sec. 4.3), but it solves the governing equation on the observed motion, so that the estimate reflects how the object moves rather than how it looks.

2.3 Equation-Based Material Estimation

A few works estimate the material by solving the governing equation of the simulator directly, without running it, and they are the most related to ours. Sag-free initialization (Hsu et al., 2022; 2023) takes a simulation asset held at rest by gravity and solves this static balance for its internal stress, from which it recovers the rest state or the stiffness. Its direct solve requires the state of every point of the asset, including its interior, which a simulation asset provides but a video does not. In experimental mechanics, the virtual fields method (Pierron & Grédiac, 2012; Avril et al., 2008) solves the same kind of equation for the material of a moving specimen. Its dynamic tests, however, need dedicated equipment, such as a thin flat specimen whose full deformation a camera can see, filmed at more than 10^6 frames per second (Fletcher et al., 2019). EfM brings this direct solve to everyday objects moving freely in ordinary multi-view video by predicting the interior that video does not show, so that the material is estimated in under a second without any special equipment.

2.4 Interior State Prediction from Observation

Applying such a direct solve to video also needs the deformation inside the object, which cameras cannot observe. One option is to extend the surface motion inwards by geometric interpolation, such as harmonic (Joshi et al., 2007) or as-rigid-as-possible (Sorkine et al., 2007) deformation. However, the interior strain of a loaded body is not the smoothest field that matches its surface, so the solve receives a wrong input (Sec. 4.5). Another option is a learned simulator (Pfaff et al., 2021; Sanchez-Gonzalez et al., 2020), but it is trained for a given material, so using it inside a material estimator would be circular. EfM instead trains a network that predicts the deformation gradient of every element from the visible surface motion, without access to the material (Sec. 3.2).

3 EfM: Simulation-Free Elastic Parameter Estimation from Observed Motion

Setup. Following the baselines we compare against (Zhong et al., 2024; Chen et al., 2025; Wang et al., 2026; Lv et al., 2026), we start from a reconstruction of the object at the first frame, which an off-the-shelf front-end provides. We discretize this reconstruction into a tetrahedral mesh with nodes $\{\mathbf{X}_i\}_{i=1}^{N}$, including interior nodes that are not directly observable in the video. The object is then recorded by N_{view} calibrated cameras $C_1, \ldots, C_{N_{\text{view}}}$ for T frames, and we take as input the trajectory of the surface nodes that these cameras can see:

$$\mathcal{O} = \big\{\mathbf{x}_i^t : i \in \mathcal{V}^t\big\}_{t=1}^{T}, \quad \mathcal{V}^t = \big\{i : \text{node } i \text{ is visible from at least one camera at time } t\big\}, \qquad (1)$$

where $\mathbf{x}_i^t$ is the position of node i at frame t. This observation is partial in two ways: interior nodes are never observed, and surface nodes are observed only when visible to at least one camera.

Goal and outline. Given the partial surface trajectories $\mathcal{O}$, EfM estimates the elastic parameters (E, ν) without running a simulator or iterative material fitting.

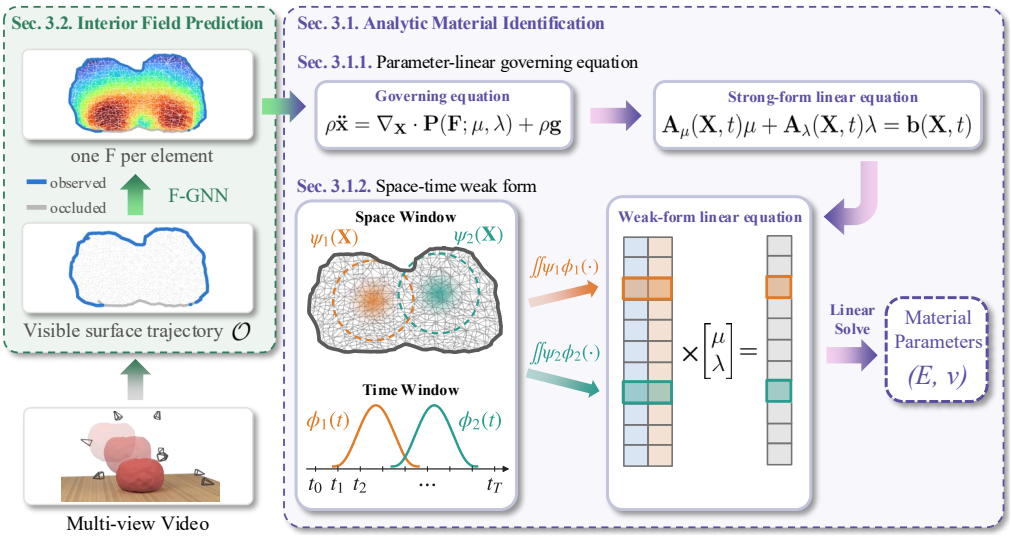


Figure 2: **Overview of EfM.** The input is the trajectory $\mathcal{O}$ of the surface visible in multi-view video (blue; Eq. 1). From it, the F-GNN predicts the deformation gradient $\mathbf{F}$ of every element of a tetrahedral mesh, including the occluded surface and the interior (left; Sec. 3.2), and the space-time weak form then gives (E, ν) by a single least-squares solve, without simulation (right; Sec. 3.1).

As illustrated in Fig. 2, our pipeline first predicts the volumetric deformation gradient from the observed surface motion (Sec. 3.2) and then recovers the material parameters through a single linear solve (Sec. 3.1). We describe the solver first, as it determines what the network has to supply.

3.1 Analytic Material Identification

3.1.1 Parameter-Linear Governing Equation

Given the material, a simulator searches for the motion that satisfies the governing equation at every material point $\mathbf{X}$ and every time t. Here $\mathbf{X}$ is the position of a point of the object at rest and $\mathbf{x}(\mathbf{X}, t)$ is where that point is at time t, so the mesh nodes $\mathbf{X}_i$ and their tracked positions $\mathbf{x}_i^t$ are samples of the two. The same equation can instead be solved for the material, given the motion. Sag-free initialization (Hsu et al., 2022; 2023) already does this. It takes the shape of a simulation asset, assumes that the asset is at rest under gravity in that shape, and solves this static balance in a single linear solve for the internal stress, from which it recovers either the rest state or the stiffness when the other is given. We write the material as the Lamé pair (μ, λ), where μ sets the resistance to shear and λ the resistance to a change of volume, and which converts in closed form to Young's modulus E and Poisson's ratio ν (Appendix B.1).

$$\underbrace{\mathbf{0}}_{\text{static}} = \nabla_{\mathbf{X}} \cdot \mathbf{P}(\mathbf{F};\mu,\lambda) + \rho\,\mathbf{g} \qquad \longrightarrow \qquad \underbrace{\rho\,\ddot{\mathbf{x}}}_{\text{ours}} = \nabla_{\mathbf{X}} \cdot \mathbf{P}(\mathbf{F};\mu,\lambda) + \rho\,\mathbf{g}. \tag{2}$$

Here ρ is the density, $\mathbf{g}$ the gravitational acceleration and $\ddot{\mathbf{x}}$ the acceleration. $\mathbf{F} = \partial\mathbf{x}/\partial\mathbf{X}$ is the deformation gradient, which measures how far the material around a point is stretched from its rest shape, and the stress $\mathbf{P}$ is that stretch scaled by μ and λ. To solve for them, a static observation is not enough, and we have to observe the object while it accelerates.

A single pose does not reveal the rest shape, so it does not reveal the stretch either. A small stretch of a stiff object and a large stretch of a soft one produce the same stress and hold the same pose. Observing dynamic motion provides the additional constraint needed to resolve this ambiguity. The acceleration and the density give the total force per unit volume, gravity accounts for part of it, and the rest is the elastic force that μ and λ scale. Observing the acceleration therefore constrains the elastic force and, given $\mathbf{F}$, determines μ and λ. We keep the direct solve and replace the static balance with the dynamic one, the right-hand form of Eq. 2. The rest state is held at the frame-0 reconstruction, so the only unknown left is (μ, λ). Once $\mathbf{F}$ is known, this balance is easily solved, because the stress $\mathbf{P}$ is *linear* in μ and λ. The benchmark simulators use Neo-Hookean energies, for which

$$\mathbf{P}(\mathbf{F};\mu,\lambda) = \mu \underbrace{\left(\mathbf{F} - \mathbf{F}^{-\top}\right)}_{\mathbf{P}_\mu(\mathbf{F})} + \lambda \underbrace{\ln J\,\mathbf{F}^{-\top}}_{\mathbf{P}_\lambda(\mathbf{F})}, \qquad J = \det \mathbf{F}. \tag{3}$$

Here $\mathbf{P}_\mu$ and $\mathbf{P}_\lambda$ are the stresses per unit μ and per unit λ, and J is the ratio of the current to the rest volume. Substituting Eq. 3 into the dynamic balance gives $\mathbf{A}_\mu(\mathbf{X}, t)\,\mu + \mathbf{A}_\lambda(\mathbf{X}, t)\,\lambda = \mathbf{b}(\mathbf{X}, t)$ at every point and time. $\mathbf{A}_\mu = \nabla_{\mathbf{X}} \cdot \mathbf{P}_\mu(\mathbf{F})$ and $\mathbf{A}_\lambda = \nabla_{\mathbf{X}} \cdot \mathbf{P}_\lambda(\mathbf{F})$ are the elastic forces per unit volume that a unit μ and a unit λ produce, and $\mathbf{b} = \rho\,(\ddot{\mathbf{x}} - \mathbf{g})$ is the force per unit volume that they must supply. Once $\mathbf{F}$ and $\ddot{\mathbf{x}}$ are known, all coefficients are determined by the motion and known quantities, so stacking this equation over the observed sequence gives $\mathbf{A}\,[\mu, \lambda]^\top = \mathbf{b}$. Its least-squares solution gives the Lamé parameters (μ, λ), from which (E, ν) follow in closed form. Unlike inverse simulation, which searches for the material by repeatedly running a simulator, EfM never runs a simulator, so there is nothing to initialize and no local minimum to escape.

The dynamic balance, however, requires the acceleration $\ddot{\mathbf{x}}$, while our observation $\mathcal{O}$ provides only positions. A natural estimate is the centered finite difference $\ddot{\mathbf{x}}(t_f) \approx (\mathbf{x}_{f+1} - 2\mathbf{x}_f + \mathbf{x}_{f-1})/\Delta t^2$. However, second-order finite differences amplify trajectory noise and can poorly approximate rapidly varying accelerations, particularly around impacts. Another missing quantity is the interior deformation gradient $\mathbf{F}$, required to compute $\mathbf{P}_\mu$ and $\mathbf{P}_\lambda$ but unobserved from video. Sag-free initialization (Hsu et al., 2022; 2023) avoids this issue by starting from a volumetric simulation asset. We address both missing quantities algorithmically: Sec. 3.1.2 avoids explicit acceleration estimation, while Sec. 3.2 predicts the interior $\mathbf{F}$ from visible surface motion. Until then, we treat $\mathbf{F}$ as known.

3.1.2 Space-Time Weak Form

Directly estimating acceleration from observed positions requires second-order differentiation, which is sensitive to trajectory noise and rapidly varying motion. We therefore avoid computing acceleration explicitly and instead enforce Eq. 2 in a weak form over local regions of space and time.

We use separable space-time windows $\varphi_{m,w}(\mathbf{X}, t) = \psi_m(\mathbf{X})\,\phi_w(t)$, where ψ_m and ϕ_w are smooth, compactly supported windows centered at $\mathbf{X}_m$ and t_w, respectively. Fig. 2 illustrates two spatial and two temporal windows. Integration by parts transfers the derivatives onto these known windows. In time, the weighted integral of $\rho\ddot{\mathbf{x}}$ with ϕ_w equals that of $\rho\mathbf{x}$ with $\ddot{\phi}_w$, so the inertial term requires only positions. In space, the weighted integral of the stress divergence $\nabla_{\mathbf{X}} \cdot \mathbf{P}$ with ψ_m equals minus that of the stress $\mathbf{P}$ with $\nabla_{\mathbf{X}}\psi_m$, so the elastic term requires the stress itself rather than its spatial derivative. Each window pair then gives

$$\underbrace{\sum_f \ddot{\phi}_w(t_f) \int_{\Omega_0} \rho\,\psi_m\,\mathbf{x}(t_f)}_{\mathbf{b}_{mw}} \;=\; -\sum_f \phi_w(t_f) \int_{\Omega_0} \mathbf{P}(\mathbf{F}) : \nabla_{\mathbf{X}}\psi_m \;+\; (\text{gravity}), \tag{4}$$

where the sum runs over the observed frames t_f and Ω_0 is the object at the first frame. This is the weak form of the governing equation, taken in space and time together. It is a well-established tool for fitting PDEs to measured data without differentiating the data, and has been used in materials testing (Pierron & Grédiac, 2012) and equation discovery (Messenger & Bortz, 2021).

The left-hand side uses the observed positions only through a weighted sum, so no acceleration is explicitly computed. The right-hand side is linear in (μ, λ) by Eq. 3, so each window pair adds three rows, one per spatial component, to the system $\mathbf{A}\,[\mu, \lambda]^\top = \mathbf{b}$. In Fig. 2, the pairs (ψ_1, ϕ_1) and (ψ_2, ϕ_2) fill the rows of their color. With M spatial windows and W time windows, the system has $3MW$ rows and still only two unknowns. As the ablation in Sec. 4.4 shows, the temporal weak form is important for robust estimation from sampled trajectories.

The remaining details of how the integrals in Eq. 4 are evaluated follow the setting of Sag-free initialization (Hsu et al., 2022; 2023), and Appendix B gives them in full. We evaluate the space integral with the discretization of the simulator that will use the estimate, the material point method (Hu et al., 2018) in our experiments. Sag-free initialization (Hsu et al., 2022) assembles the same equation on mass-spring systems (Terzopoulos et al., 1987), finite elements (FEM) (Sifakis & Barbic, 2012), the material point method (MPM) (Stomakhin et al., 2013) and position-based dynamics (PBD) (Macklin et al., 2016). Our rows add only the inertial term to theirs, so EfM inherits this generality and applies to any of these simulators by assembling on its discretization. The contact force that the ground applies during an impact is not observed either, so, following Sag-free initialization, we include it as an additional unknown of the same linear system and solve for it with (μ, λ).

3.2 F-GNN: Interior Field Prediction

Solving the governing equation for the material requires the deformation gradient $\mathbf{F}$ at every point inside the object, and video cannot observe it. The F-GNN predicts this field (Fig. 2, left). It takes the visible surface trajectory $\mathcal{O}$ and the tetrahedral mesh of the first frame, and returns $\mathbf{F}$ for every element at every frame. A simple alternative is to extend the observed surface motion inwards with a smooth interpolation and differentiate it. However, the interior strain of a loaded body need not be the smoothest field consistent with its boundary motion, so interpolation can provide an inaccurate input to the solve. Replacing the network by such an interpolation degrades the estimate (Table 2, row J, Sec. 4.5).

The F-GNN predicts how the object is deformed, not what it is made of, leaving material identification to the governing equation. Importantly, the F-GNN does not take the material parameters as input; it predicts the kinematic state needed by the solver from the observed surface motion. The material parameters are then identified by enforcing the balance between inertia and internal force in Sec. 3.1. A network that regressed the material directly would bypass this physical constraint, as feed-forward predictors do (Sec. 2.2). Our training target is therefore the deformation gradient used by the simulator to compute stress, exactly the quantity required by Eq. 3.

Architecture. To carry the information on the observed surface inwards, the network passes messages between neighbors on the mesh, as graph networks for mesh-based simulation do (Pfaff et al., 2021). We adopt the finite element-inspired hypergraph network (FEIH-GNN) of Gao et al. (2024), whose graph has nodes and elements as its two kinds of vertices and connects each element to its corner nodes. Each round of message passing updates an element from its corner nodes and then each node from its elements, much as a finite element solver assembles element contributions at the nodes. The network therefore maintains an element-level state, matching the domain on which $\mathbf{F}$ is defined. We keep this message passing, extended from 2D cells to tetrahedra, but whereas FEIH-GNN predicts the next flow state of a fluid, our decoder reads each element and returns its $\mathbf{F}$ at the current frame.

Inputs and outputs. The network takes a geometric estimate of $\mathbf{F}$ as input and improves on it. In every frame we first fill the unobserved nodes by harmonic interpolation, with the visible nodes as boundary values, which gives a deformation gradient $\mathbf{F}_{\text{fill}}$ for each element. Each node carries its rest position, whether a camera sees it, and its displacements over five consecutive frames, so the network can infer local velocity and acceleration without being given them. Each element carries its rest volume and $\mathbf{F}_{\text{fill}}$, and each node–element connection carries the offset of the node from the element center in the rest and current frames. Before entering the network, the displacements and $\mathbf{F}_{\text{fill}} - \mathbf{I}$ are divided by the typical strain of the sequence, so that a stiff object that barely deforms and a soft one that deforms a lot give inputs of similar magnitude. The decoder outputs a rotation $\mathbf{R}$ and a symmetric log-stretch $\mathbf{S}$ for each element, and we assemble $\mathbf{F} = \mathbf{R}\exp(\mathbf{S})$ after scaling $\mathbf{S}$ back. This parameterization ensures $\det \mathbf{F} > 0$, preventing element inversion, and initializes $\mathbf{F}$ to $\mathbf{I}$.

Training. We train on simulated sequences whose geometries never appear in the benchmarks. The training set has 264 sequences of 27 Objaverse (Deitke et al., 2023) objects simulated with the material point method, and the loss compares the predicted and true $\mathbf{F}$ in rotation and in log-strain. In the final stage of training, half of the steps see the full surface and the other half see only the part visible from one to three random directions, with noise added to the surface positions, so that the network learns to work from partial observations. One checkpoint serves every object in both benchmarks, and Appendix B.6 gives the architecture, the loss and the schedule in full.

The network is the only learned part of EfM. The solve of Sec. 3.1 has no learned parameter and no per-sequence tuning, and it uses the same windows for every object in both benchmarks and for the real captures.

4 Experiments

We evaluate whether EfM recovers material and future motion more accurately than existing video-based estimators, at a cost close to that of feed-forward prediction.

4.1 Experiment Settings

Datasets. We evaluate on two benchmarks that provide the ground-truth material of every object, filmed as it falls, hits the ground and deforms. **PAC-NeRF** (Li et al., 2023) is a public benchmark for video-based material estimation, and we use its elastic split, ten objects whose Young's modulus

Table 2: **Quantitative comparison on PAC-NeRF and Spring-Gaus-Diverse.** CD ($\times 10^3\,\mathrm{mm}^2$) is measured on the observed (rec) and unseen future (pred) windows after rolling each method's (E, ν) through the same simulator, and rows marked † use their own solver. Time is the fitting time per sequence on one GPU, averaged over the two benchmarks. Bold and underline mark the best and second best among competing methods (A–F), and rows G–J are ablations (Sec. 4.4).

	Method	PAC-NeRF				Spring-Gaus-Diverse				Time (s)
		$\lvert\log E\rvert \downarrow$	$\lvert\Delta\nu\rvert \downarrow$	$\mathrm{CD}_{\mathrm{rec}} \downarrow$	$\mathrm{CD}_{\mathrm{pred}} \downarrow$	$\lvert\log E\rvert \downarrow$	$\lvert\Delta\nu\rvert \downarrow$	$\mathrm{CD}_{\mathrm{rec}} \downarrow$	$\mathrm{CD}_{\mathrm{pred}} \downarrow$	
A	**EfM (ours)**	**0.072**	**0.033**	**0.15**	**0.20**	**0.214**	**0.073**	**0.20**	**0.38**	0.74
B	Vid2Sim (Stage I)	0.564	0.144	0.76	3.45	<u>0.698</u>	<u>0.156</u>	<u>0.62</u>	<u>0.97</u>	0.12
C	Vid2Sim (Stage I + II)	<u>0.542</u>	0.161	0.77	3.32	0.733	0.161	0.82	1.27	1035
D	PhysGM	0.805	<u>0.072</u>	1.16	3.28	1.014	0.231	1.54	3.61	0.11
E	Spring-Gaus†	–	–	<u>0.23</u>	<u>1.03</u>	–	–	1.23	9.61	3376
F	ReconPhys†	–	–	1.10	6.40	–	–	2.10	9.87	0.28
G	Direct solve, no weak form	0.772	0.055	0.93	4.72	0.849	0.110	1.43	2.15	–
H	− spatial weak form	0.632	0.049	0.64	3.30	0.869	0.098	1.49	2.21	–
I	− temporal weak form	0.220	0.037	0.17	0.48	0.304	0.064	0.26	0.57	–
J	− learned $\mathbf{F}$ (Sec. 3.2)	0.086	0.162	0.15	0.35	0.664	0.189	1.15	1.61	–

Table 3: **Accuracy of the predicted interior deformation.** Relative errors of the learned field and the harmonic interpolation of row J against the simulator's deformation gradient, in strain and in the stress that the solve takes as input.

Interior field	PAC-NeRF		Spring-Gaus-Diverse	
	Strain error ↓	Stress error ↓	Strain error ↓	Stress error ↓
Harmonic interpolation (Joshi et al., 2007)	0.220	1.232	0.172	2.494
Learned (ours)	**0.194**	**0.821**	**0.063**	**0.411**

ranges from 2.3×10^4 to 4.2×10^6 Pa, with the first eleven frames for fitting and the last five for prediction. The benchmark released with Spring-Gaus (Zhong et al., 2024) was designed for reconstruction rather than material estimation, and its materials span only one order of magnitude in E (an analysis of it is in Appendix A.5). We therefore propose **Spring-Gaus-Diverse**, which simulates the same geometries under the same protocol with diverse materials, E from 2.1×10^4 to 1.7×10^6 Pa and ν from 0.10 to 0.43. We keep the original split of twenty frames for fitting and ten for prediction, and we will release the benchmark (Appendix B.7).

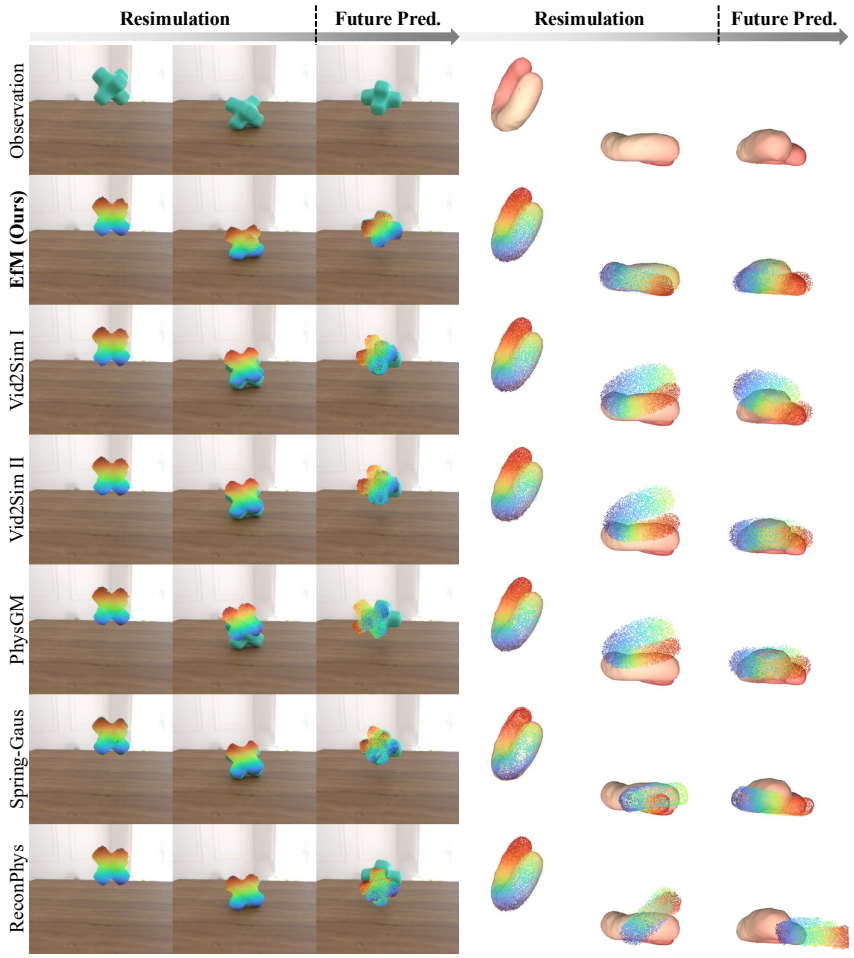
Observation protocol. All methods start from the same frame-0 geometry and initial velocity and fit over the same window, so the comparison isolates how the material is estimated. The baselines read the rendered video, whereas EfM receives only the trajectory of the surface nodes that the benchmark's cameras see.

Baselines. We compare with **Vid2Sim** (Chen et al., 2025), which predicts the material with a feed-forward network (Stage I) and refines it by inverse simulation (Stage II), reporting both stages. **Spring-Gaus** (Zhong et al., 2024) fits a spring-mass model by inverse simulation, and **ReconPhys** (Wang et al., 2026) predicts spring parameters from a monocular video in a single forward pass. Neither returns (E, ν), so both are rolled out with their own solvers (†). **PhysGM** (Lv et al., 2026) predicts (E, ν) from still images.

Evaluation metrics. Following Vid2Sim (Chen et al., 2025), we measure material accuracy by the error in Young's modulus on a logarithmic scale, $|\log_{10}(E/E^*)|$, since E varies by orders of magnitude across objects, and by the absolute error in Poisson's ratio, $|\Delta\nu|$. Following Spring-Gaus (Zhong et al., 2024), we measure motion accuracy by rolling each method's (E, ν) through the same simulator and computing the Chamfer distance to the ground truth on the observed window ($\mathrm{CD}_{\mathrm{rec}}$) and on the unseen future window ($\mathrm{CD}_{\mathrm{pred}}$). Cost is the wall-clock time of material fitting per sequence, excluding model loading and the rendering or tracking that produces the input. Every metric is the mean over the objects of each benchmark.

4.2 Material and Future-Motion Accuracy

As shown in Table 2, EfM outperforms all baselines on both benchmarks, in the estimated material and in the resimulated and predicted motion. Its error in Young's modulus is $7.5\times$ lower than that of the best baseline on PAC-NeRF and $3.3\times$ lower on Spring-Gaus-Diverse. The material it recovers also predicts the unseen motion better than any baseline.



(a) Qualitative comparisons on PAC-NeRF. **(b) Qualitative comparisons on SG-Diverse.**

Figure 3: **Qualitative comparison** on (a) PAC-NeRF and (b) Spring-Gaus-Diverse. Each method's estimated material is rolled out from the same initial state, and its particles, colored by height in the first frame, are overlaid on the benchmark rendering, which shows through wherever the prediction deviates. Vid2Sim I and II are rows B and C of Table 2.

The results also expose the limitations of the baselines discussed in Sec. 2. Vid2Sim's inverse simulation barely moves its feed-forward guess and even worsens it on Spring-Gaus-Diverse, so its accuracy is set by the initialization rather than measured from the motion. PhysGM, which sees no motion, returns nearly the same Young's modulus for every object. Spring-Gaus comes closest to EfM in motion on PAC-NeRF but predicts the future less accurately, and ReconPhys, which never fits the motion, has the largest prediction errors. Fig. 3 shows the same pattern on one sequence from each benchmark (more in Appendix A.1), where EfM follows the object into the future window, whereas Vid2Sim and PhysGM overestimate the stiffness of the Spring-Gaus-Diverse object, which then bounces too high.

4.3 Computational Cost

EfM fits a sequence in under a second, three orders of magnitude faster than inverse simulation (Table 2), because it assembles one linear system and solves it once instead of iterating through $\sim 10^3$ simulation steps per update. The fit takes 0.52 s for the 11 observed frames of PAC-NeRF and 0.97 s for the 20 of Spring-Gaus-Diverse, ~ 21 frames per second in both cases, which is close to the capture rate of the videos, although EfM solves over the whole observed window at once rather than frame by frame. Feed-forward predictors are only a few times faster but far less accurate.

4.4 Ablation Study

Rows G–J of Table 2 remove the components of Sec. 3 one at a time. Without either weak form (row G), the direct solve is less accurate than Vid2Sim on both benchmarks. The spatial weak form recovers most of this gap on both benchmarks (row I), the temporal one alone helps only on

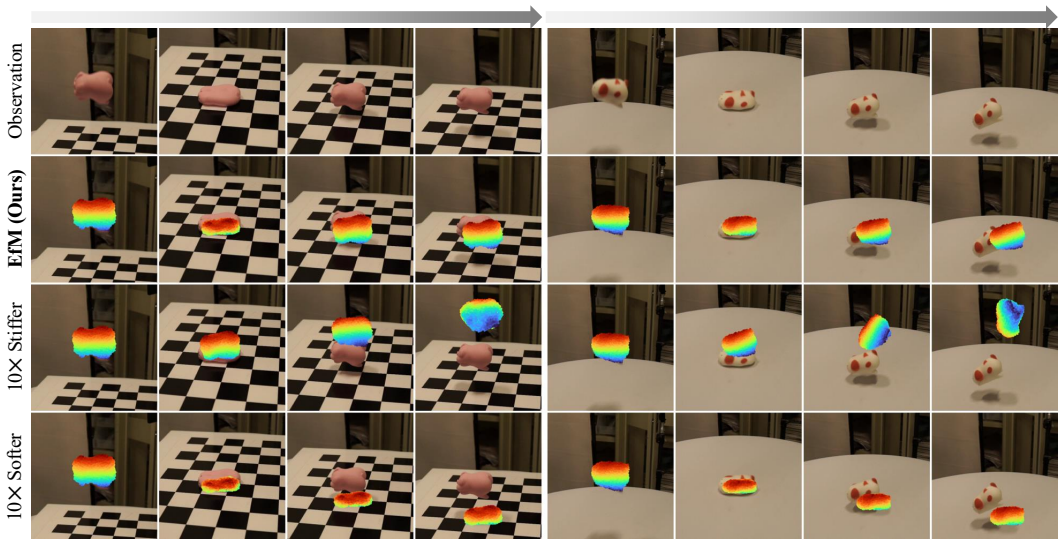


Figure 4: **Real captured video** (Sec. 4.6). Two objects shown before the fall and one, five and ten frames after the first contact with the table. Each re-simulation starts from the first frame and is overlaid on it, colored by height in the first frame. The $10\times$ stiffer material bounces too high and the $10\times$ softer one sinks, whereas the estimated material follows the real object.

PAC-NeRF (row H), and only together do they reach the accuracy of the full method. Replacing the learned interior field by interpolation (row J) degrades every metric on Spring-Gaus-Diverse and, on PAC-NeRF, Poisson's ratio and the future motion, and Sec. 4.5 explains why.

4.5 Accuracy of the Predicted Interior Field

Table 3 explains row J by comparing the learned field and the harmonic interpolation it replaces with the deformation gradient of the simulator. The learned field is more accurate on both benchmarks, most of all in stress error, the input of the solve, which is six times lower on Spring-Gaus-Diverse.

4.6 Real Captured Video

We finally apply EfM to the real captures of Spring-Gaus (Zhong et al., 2024), five objects dropped onto a table and filmed from three views. Since the captures have no ground-truth material, we re-simulate each drop with the estimated Young's modulus, and with it made $10\times$ stiffer and $10\times$ softer, and measure the silhouette IoU against the real object masks, adding the same stiffness-proportional viscous damping to every object and all three materials since real objects are not perfectly elastic (Appendix A.2). As shown in Fig. 4 and Table 4, the estimated material reproduces all five objects best (mean IoU 0.61, against 0.37 for the stiffer and 0.46 for the softer material), and the difference shows after the impact, where the stiffer material bounces too high and the softer one sinks.

5 Conclusion

We present EfM, which estimates the elastic parameters of a freely moving object from multi-view video without running a simulator. Unlike inverse simulation, which takes tens of minutes per sequence, and feed-forward prediction, which ignores the observed motion, EfM solves the governing equation directly for the material in a single linear solve. It supplies the two quantities that video lacks, the interior deformation, predicted by the F-GNN from the visible surface motion, and the acceleration, which a space-time weak form of the equation never computes. On two benchmarks, EfM recovers material and future motion more accurately than state-of-the-art baselines, three orders of magnitude faster than inverse simulation, which makes it practical for large object collections.

Limitations. Like most of the baselines, EfM estimates a homogeneous, isotropic Neo-Hookean material, and richer models such as plasticity and spatially varying material add columns to the same linear system, which we leave to future work. EfM also depends on the accuracy of the tracked surface, although its error grows gradually with tracking noise (Appendix A.3), and it does not yet estimate dissipation, which re-simulating a real object requires (Sec. 4.6).

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing, such as improving grammar and wording of text drafted by the authors. We have not used generative AI tools for retrieval and discovery, research ideation or execution, drafting sections of the paper, generating synthetic datasets, or proving mathematical claims. All AI-assisted text was reviewed by the authors, and we take responsibility for the final content of this work.

REFERENCES

Stéphane Avril, Marc Bonnet, Anne-Sophie Bretelle, Michel Grédiac, François Hild, Patrick Ienny, Félix Latourte, Didier Lemosse, Stéphane Pagano, Emmanuel Pagnacco, et al. Overview of identification methods of mechanical parameters based on full-field measurements. *Experimental Mechanics*, 48(4):381–402, 2008.

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

Chuhao Chen, Zhiyang Dou, Chen Wang, Yiming Huang, Anjun Chen, Qiao Feng, Jiatao Gu, and Lingjie Liu. Vid2sim: Generalizable, video-based reconstruction of appearance, geometry and physics for mesh-free simulation. In *CVPR*, 2025.

Rishit Dagli, Donglai Xiang, Vismay Modi, Charles Loop, Clement Fuji Tsang, Anka He Chen, Anita Hu, Gavriel State, David I. W. Levin, and Maria Shugrina. Vomp: Predicting volumetric mechanical property fields. In *ICLR*, 2026.

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. In *CVPR*, 2023.

L Fletcher, J Van-Blitterswyk, and F Pierron. A novel image-based inertial impact test (ibii) for the transverse properties of composites at high strain rates. *Journal of Dynamic Behavior of Materials*, 5(1):65–92, 2019.

Rui Gao, Indu Kant Deo, and Rajeev K Jaiman. A finite element-inspired hypergraph neural network: Application to fluid dynamics simulations. *Journal of Computational Physics*, 504:112866, 2024.

Jerry Hsu, Nghia Truong, Cem Yuksel, and Kui Wu. A general two-stage initialization for sag-free deformable simulations. In *SIGGRAPH*, 2022.

Jerry Hsu, Tongtong Wang, Zherong Pan, Xifeng Gao, Cem Yuksel, and Kui Wu. Sag-free initialization for strand-based hybrid hair simulation. In *SIGGRAPH*, 2023.

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. In *SIGGRAPH*, 2018.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. In *ICLR*, 2020.

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 *AAAI*, 2025.

Krishna Murthy Jatavallabhula, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jerome Parent-Levesque, Kevin Xie, Kenny Erleben, Liam Paull, Florian Shkurti, Derek Nowrouzezahrai, and Sanja Fidler. gradsim: Differentiable simulation for system identification and visuomotor control. In *ICLR*, 2021.

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

Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. In *SIGGRAPH*, 2007.

Takuhiro Kaneko. Improving physics-augmented continuum neural radiance field-based geometry-agnostic system identification with lagrangian particle optimization. In *CVPR*, 2024.

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

Jahyeok Koo, Inès Hyeonsu Kim, Mungyeom Kim, Junghyun Park, Seohyun Park, Jaeyeong Kim, Jung Yi, Seokju Cho, and Seungryong Kim. Mv-tap: Tracking any point in multi-view videos. In *CVPR*, 2026.

Long Le, Ryan Lucas, Chen Wang, Chuhao Chen, Dinesh Jayaraman, Eric Eaton, and Lingjie Liu. Pixie: Fast and generalizable supervised learning of 3d physics from pixels. *arXiv preprint arXiv:2508.17437*, 2025.

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

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

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

Chunji Lv, Zequn Chen, Donglin Di, Weinan Zhang, Hao Li, Wei Chen, Yinjie Lei, and Changsheng Li. Physgm: Large physical gaussian model for feed-forward 4d synthesis. In *CVPR*, 2026.

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*, 2016.

Daniel A Messenger and David M Bortz. Weak sindy for partial differential equations. *Journal of computational physics*, 443:110525, 2021.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. In *ICLR*, 2021.

F. Pierron and M. Grédiac. *The Virtual Fields Method: Extracting Constitutive Mechanical Parameters from Full-field Deformation Measurements*. Springer New York, 2012. ISBN 9781461418238.

Yi-Ling Qiao, Alexander Gao, and Ming C. Lin. Neuphysics: Editable neural geometry and physics from monocular videos. In *NeurIPS*, 2022.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *ICML*, 2020.

Eftychios Sifakis and Jernej Barbic. Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *SIGGRAPH Courses*, 2012.

Olga Sorkine, Marc Alexa, et al. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, 2007.

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. In *SIGGRAPH*, 2013.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH*, 1987.

Bin Wang, Longhua Wu, KangKang Yin, Uri Ascher, Libin Liu, and Hui Huang. Deformation capture and modeling of soft objects. In *SIGGRAPH*, 2015.

Boyuan Wang, Xiaofeng Wang, Yongkang Li, Zheng Zhu, Yifan Chang, Angen Ye, Guosheng Zhao, Chaojun Ni, Guan Huang, Yijie Ren, Yueqi Duan, and Xingang Wang. Reconphys: Reconstruct appearance and physical attributes from single video. In *ECCV*, 2026.

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

Albert J. Zhai, Yuan Shen, Emily Y. Chen, Gloria X. Wang, Xinlei Wang, Sheng Wang, Kaiyu Guan, and Shenlong Wang. Physical property understanding from language-embedded feature fields. In *CVPR*, 2024.

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

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

A Additional Results

A.1 Additional Qualitative Results

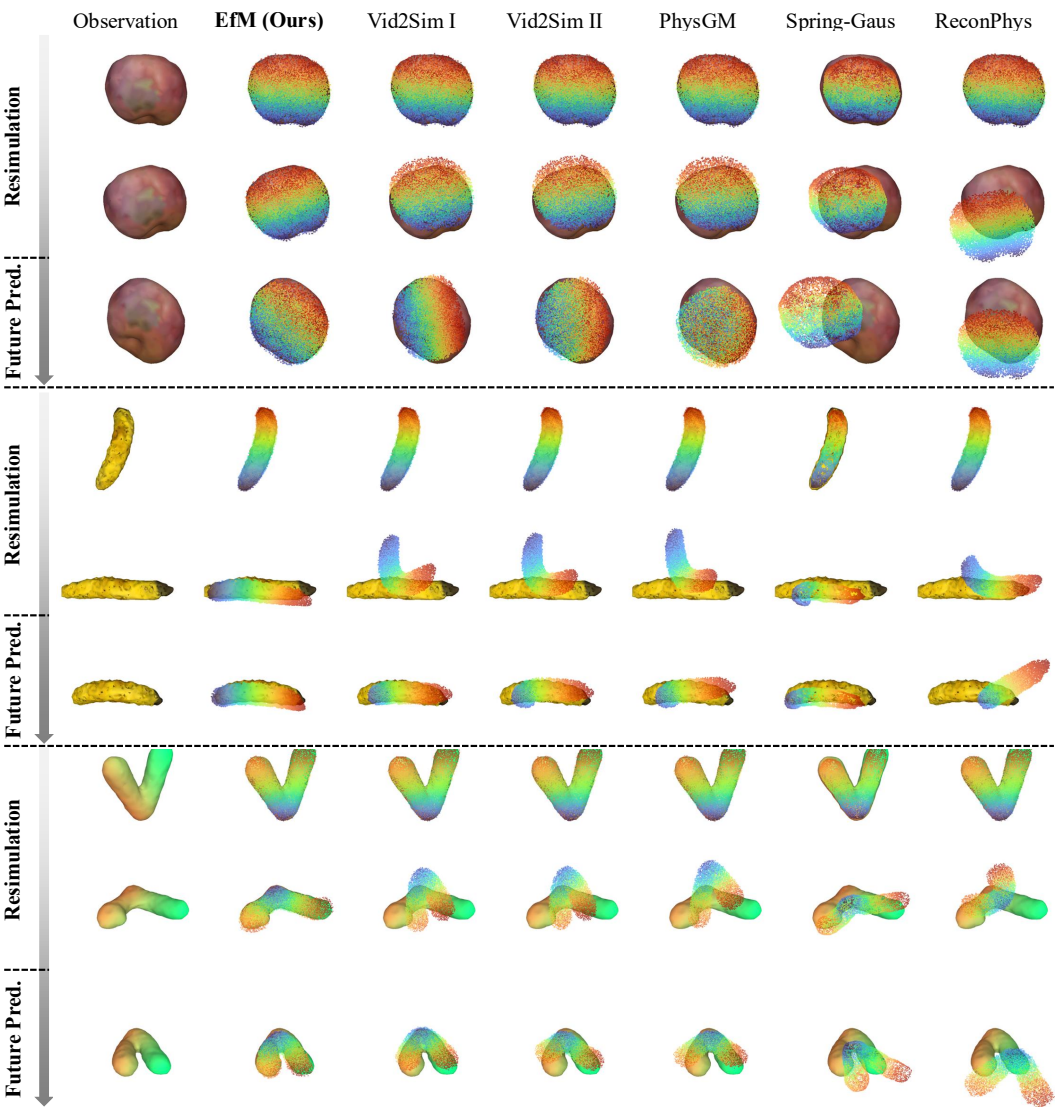
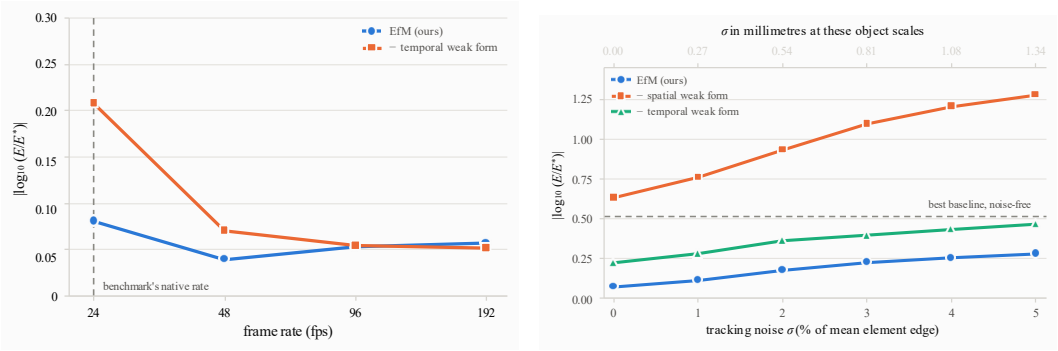


Figure 5: **Additional qualitative comparison** on Spring-Gaus-Diverse (*apple*, *banana* and *V*, from top to bottom), in the same format as Fig. 3. Each method's estimated material is rolled out from the same initial state, and its particles are overlaid on the benchmark rendering. For each object, the first two rows lie in the observed window and the last row in the unseen future window.

To show that EfM also follows objects of other shapes, we report three more Spring-Gaus-Diverse sequences in Fig. 5, with shapes that differ from the one in Fig. 3. EfM follows each object through the fall and the impact, and it keeps following the object in the unseen future window. The baselines fail in the ways discussed in Sec. 4.2. Vid2Sim and PhysGM overestimate the stiffness of the banana and the V, so part of each object bounces up after the impact while the observed object stays on the ground. Spring-Gaus and ReconPhys, which simulate with their own spring-mass models, drift sideways or sink below the observed shape, and the drift grows in the future window, most visibly for the apple and the V.

Table 4: **Real captured video: silhouette IoU against the real object masks** (Sec. 4.6). Each object is re-simulated with the estimated Young's modulus and with it made 10× stiffer and 10× softer, with the same damping in all three. The IoU is averaged over the three views. *All* averages frames 0–19, *impact* the frames up to four after the first contact, and *rebound* the remaining frames. Bold marks the best of the three.

	Ours			10× stiffer			10× softer		
Object	All	Impact	Rebound	All	Impact	Rebound	All	Impact	Rebound
burger	**0.72**	0.72	**0.73**	0.37	0.61	0.00	0.55	**0.79**	0.18
pig	**0.67**	**0.77**	**0.54**	0.39	0.65	0.01	0.39	0.65	0.01
bun	**0.57**	0.69	**0.42**	0.31	0.57	0.00	0.56	**0.78**	0.30
dog	**0.58**	**0.75**	**0.37**	0.29	0.53	0.00	0.38	0.66	0.04
potato	**0.50**	0.68	**0.25**	0.46	**0.74**	0.05	0.39	0.62	0.04
Mean	**0.61**	**0.72**	**0.46**	0.37	0.62	0.01	0.46	0.70	0.11



(a) Frame rate (b) Tracking noise

Figure 6: **Robustness to frame rate and tracking noise** (Sec. A.3). Both panels plot the mean $|\log E|$ over the ten PAC-NeRF sequences. (a) The sequences are re-simulated at 192 fps and subsampled over the same 0.417 s, with and without the temporal weak form. Values differ slightly from Table 2 because the sequences are re-simulated. (b) Gaussian noise is added to the observed surface, with and without the spatial weak form. The bottom axis gives σ relative to each object's mean element edge and the top axis the same noise in millimetres.

A.2 Real Captured Video

To show that the result of Sec. 4.6 holds for each object and not only on average, we report the silhouette IoU for every object in Table 4. EfM estimates the material from all 20 frames with the same solve as in the benchmarks, and all frames lie in this window, since the captures have no held-out frames. Since the captures provide no particle states, we obtain the input trajectory by tracking surface points in the three videos with MV-TAP (Koo et al., 2026), triangulating them, and registering the mesh of the first frame to these points and to the object masks. Each re-simulation is projected with the calibrated cameras of the capture, and the IoU is averaged over the three views. Because EfM estimates only the elastic parameters, we add a viscous damping $\eta = \tau E$ with the same $\tau = 4.5$ ms for every object, where E is the estimated modulus, and the stiffer and softer materials keep this η, so that only Young's modulus differs between the three.

A.3 Robustness to Frame Rate and Tracking Error

To show how much each weak form contributes and how far EfM tolerates an imperfect capture, we vary the frame rate and the tracking noise of the PAC-NeRF sequences and report the error in Fig. 6.

The temporal weak form halves the frame rate that EfM needs (Fig. 6a). At a low frame rate the impact spans only a few frames, so an acceleration computed by finite differences of the positions is inaccurate, and the temporal weak form avoids computing it. We re-simulate the ten PAC-NeRF sequences at 192 fps with the protocol of the benchmark and subsample them over the same time interval, so that only the frame rate changes, and run EfM with and without the temporal weak form on the same data. At the native 24 fps of the benchmark, EfM reaches $|\log E| = 0.080$, and removing

Table 5: **Number of camera views** (Sec. A.4). Each row uses K cameras of the benchmark's own rig, chosen by farthest-point sampling over the camera positions. *Vis.* is the percentage of surface nodes that at least one of them sees at frame 0. All values are means over the benchmark objects.

Cameras K	PAC-NeRF		Spring-Gaus-Diverse	
	Vis. (%)	$\lvert\log E\rvert \downarrow$	Vis. (%)	$\lvert\log E\rvert \downarrow$
1	26.4	0.178	30.2	0.592
2	50.8	0.110	55.3	0.347
3	57.2	0.073	66.3	0.300
5	62.1	0.070	71.7	0.221
8	62.4	0.072	73.3	0.216
all (11 / 10)	62.5	0.072	73.6	0.214

the temporal weak form raises the error to 0.208. Without the temporal weak form, EfM needs 48 fps to reach a similar error. Above about 96 fps the two curves meet, because finite differences are then accurate on their own. EfM therefore works at ordinary video rates rather than at the high-speed rates that dynamic full-field identification usually assumes (Sec. 2.3).

The spatial weak form makes EfM robust to tracking error (Fig. 6b). We add independent Gaussian noise to every observed surface node at every frame. We set its standard deviation σ relative to the mean element edge of each object, because the solve computes strain across elements and the resulting strain error depends on σ relative to that edge. At σ equal to 5% of the mean edge, about 1.3 mm for these objects, the error of EfM grows from 0.072 to 0.277, which is still about half the error of the best baseline without noise. Without the spatial weak form, the error grows more than three times as fast with noise and exceeds that of the best baseline even without noise. The two weak forms thus serve different purposes. The temporal weak form lowers the required frame rate, and the spatial weak form adds tolerance to tracking error.

A.4 Ablation on the Number of Camera Views

To show how many cameras EfM needs, we report its accuracy when it observes each sequence with only K cameras of the benchmark rig. Every setting thus observes a real subset of the real rig rather than an idealized one. Table 5 reports accuracy together with the fraction of the surface that those K cameras see. Two things follow. First, accuracy improves monotonically with K and saturates at $K = 5$, and the visible fraction saturates at the same place, so the plateau is a property of the rig geometry rather than of the estimator: past five cameras the rig mostly re-observes surface it already sees. Second, even the full rig leaves roughly a third of the surface unseen, and the interior is unobserved at every K, which is the regime the interior field of Sec. 3.2 is built for.

A.5 Material Identifiability Across Benchmarks

To show whether each method's estimates follow the true material, and why we do not rank methods on the benchmark released with Spring-Gaus, we plot the estimated against the true material for every object of the three benchmarks in Fig. 7. EfM is the only method whose estimates rise with the true material on all three benchmarks, including the released one. The estimates of EfM rise with the true Young's modulus ($r = 0.99$ on PAC-NeRF, 0.92 on Spring-Gaus-Diverse and 0.33 on the released benchmark), whereas Vid2Sim and PhysGM return nearly the same value for every object ($r \leq 0.18$ on all three). The released benchmark is too narrow to rank methods by mean error. Its materials span one order of magnitude in E, from 10^5 to 10^6 Pa, with four values of ν, whereas PAC-NeRF and Spring-Gaus-Diverse span about two. On such a narrow range, an estimate that stays near the middle of the range reaches a low mean error without following the object. We therefore report material accuracy on Spring-Gaus-Diverse, which keeps the geometries and the protocol of the released benchmark and only widens its materials.

A.6 Vid2Sim with a Trajectory Objective

To show that the gap to Vid2Sim does not come from the different observation it uses, we give Vid2Sim the same observation as EfM and report the result in Table 6. This does not close the gap. In

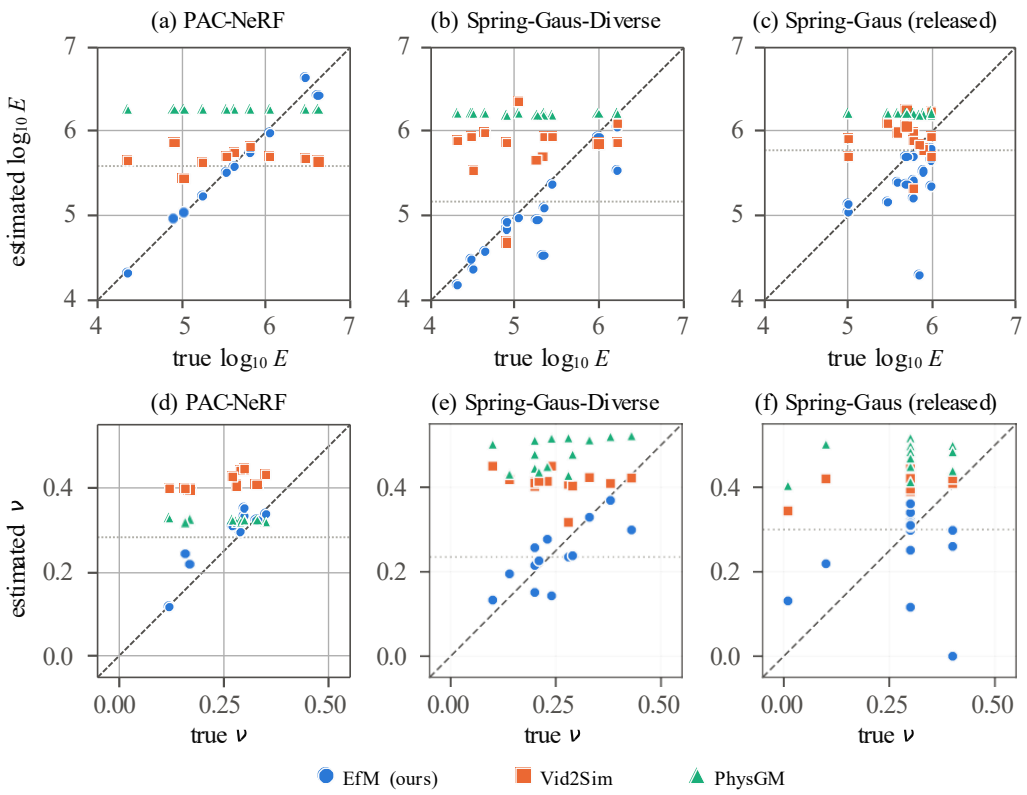


Figure 7: **Material identifiability across benchmarks.** Estimated against true Young's modulus (top) and Poisson's ratio (bottom) for every object of the three benchmarks. The dashed line marks a perfect estimate.

Table 6: **Vid2Sim with a trajectory objective.** We run Stage I + II of Vid2Sim with its original rendering objective (row C of Table 2) and with the rendering loss replaced by the squared distance to our observation $\mathcal{O}$. All values are means over the benchmark objects, and CD is in units of 10^3 mm^2, with r the observed window and p the unseen future window.

	PAC-NeRF				Spring-Gaus-Diverse			
Method	$\lvert\log E\rvert$	$\lvert\Delta\nu\rvert$	$\mathrm{CD_r}$	$\mathrm{CD_p}$	$\lvert\log E\rvert$	$\lvert\Delta\nu\rvert$	$\mathrm{CD_r}$	$\mathrm{CD_p}$
EfM (ours)	**0.072**	**0.033**	**0.15**	**0.20**	**0.214**	**0.073**	**0.20**	**0.38**
Vid2Sim, rendering objective	0.542	0.161	0.77	3.32	0.733	0.161	0.82	1.27
Vid2Sim, trajectory objective	0.515	0.143	0.62	3.19	0.604	0.151	0.60	1.42

Table 2, Vid2Sim runs as released, so its Stage II refinement fits the rendered video. Here we replace that rendering loss by the squared distance between Vid2Sim's simulated points and our observation $\mathcal{O}$, the surface points visible to the camera rig, and keep everything else unchanged. The trajectory objective mainly improves the material estimate of Vid2Sim, lowering $\lvert\log E\rvert$ from 0.542 to 0.515 on PAC-NeRF and from 0.733 to 0.604 on Spring-Gaus-Diverse, but EfM still has 2.8–$7.2\times$ lower $\lvert\log E\rvert$ and 3.7–$16\times$ lower $\mathrm{CD}_{\mathrm{pred}}$ than the trajectory variant.

B Implementation Details

This section lists every quantity needed to reproduce the solve (Sec. 3.1.2) and the network (Sec. 3.2). All values are fixed across objects and benchmarks, and no quantity of the estimator is tuned per sequence.

B.1 Constitutive Model

Both benchmark simulators use the compressible Neo-Hookean model, whose first Piola–Kirchhoff stress is

$$\mathbf{P} = \mu\,(\mathbf{F} - \mathbf{F}^{-\top}) + \lambda\,\ln J\,\mathbf{F}^{-\top}, \qquad J = \det \mathbf{F}. \tag{5}$$

The assembly below spreads the Kirchhoff stress $\boldsymbol{\tau} = \mathbf{P}\mathbf{F}^{\top}$, which is also linear in the two parameters,

$$\boldsymbol{\tau} = \mu\,\underbrace{(\mathbf{F}\mathbf{F}^{\top} - \mathbf{I})}_{\boldsymbol{\tau}_\mu} + \lambda\,\underbrace{\ln J\,\mathbf{I}}_{\boldsymbol{\tau}_\lambda}, \tag{6}$$

so every row of $\mathbf{A}$ has one entry computed from $\boldsymbol{\tau}_\mu$ and one from $\boldsymbol{\tau}_\lambda$. The Lamé pair converts to Young's modulus and Poisson's ratio by $E = \mu(3\lambda + 2\mu)/(\lambda + \mu)$ and $\nu = \lambda/\big(2(\lambda + \mu)\big)$.

B.2 Windows

The $M = 100$ spatial centers $\mathbf{X}_m$ are chosen among the interior nodes by farthest-point sampling in the first frame. Each spatial window is the Wendland C^2 function

$$\psi_m(\mathbf{X}) = \Big(1 - \tfrac{r}{R_s}\Big)_+^4 \Big(1 + \tfrac{4r}{R_s}\Big), \qquad r = \|\mathbf{X} - \mathbf{X}_m\|, \tag{7}$$

with radius R_s equal to 0.3 times the diagonal of the bounding box of the interior nodes. The temporal windows cover the observation window of length $T = (N_f - 1)\,\Delta t$, with $N_f = 11$ frames on PAC-NeRF and $N_f = 20$ on Spring-Gaus. The $W = 20$ centers t_w are spaced uniformly in $[R_t,\, T - R_t]$ with radius $R_t = 0.3\,T$, and each window and its second derivative are

$$\phi_w(t) = (1 - u^2)_+^3, \qquad \ddot{\phi}_w(t) = -\frac{6}{R_t^2}\,(1 - u^2)(1 - 5u^2)\ \ (|u| \le 1), \qquad u = \frac{t - t_w}{R_t}. \tag{8}$$

Every temporal window lies inside $[0, T]$, and ϕ_w and $\dot{\phi}_w$ vanish at its edges, so integration by parts in time leaves no boundary term. Time integrals are evaluated as sums over the observed frames with weight Δt.

B.3 Assembly of the Space-Time Weak Form

We evaluate the space integral in Eq. 4 on a background grid with the moving-least-squares transfer of MLS-MPM (Hu et al., 2018), as the material-point form of Sag-free initialization does (Hsu et al., 2022). Every mesh node n is a material point with lumped mass $m_n = \rho \sum_{e \ni n} V_e/4$ and volume $V_n = m_n/\rho$, where V_e is the rest volume of element e and ρ is the density of the benchmark (1000kg/m^3). The stress of each element is averaged to its four nodes, and elements whose smallest dihedral angle is below 15° or that invert in any frame are left out. The grid spacing is $\Delta x_g = 1.25\,(V/N)^{1/3}$, where V is the total volume and N the number of nodes, so that the grid resolves the material points that are available. With the quadratic B-spline weights w_{in} of MLS-MPM over a $3 \times 3 \times 3$ neighborhood, a stress field $\boldsymbol{\tau}$ produces the grid force and, transferred back, the force on material point n,

$$\mathbf{f}_i = -\sum_n w_{in}\,V_n\,\frac{4}{\Delta x_g^2}\,\boldsymbol{\tau}_n\,(\mathbf{x}_i - \mathbf{x}_n), \qquad \mathbf{f}_n = m_n \sum_i w_{in}\,\frac{\mathbf{f}_i}{m_i}, \qquad m_i = \sum_n w_{in}\,m_n. \tag{9}$$

Grid nodes at or below the ground layer are excluded from the second sum, because the simulator overwrites their velocity with the boundary condition. $\mathbf{f}_n$ is the internal force the simulator itself applies to material point n, and the spatial weighting of Eq. 4 becomes a sum over nodes. Writing $\mathbf{f}_n^\mu$ and $\mathbf{f}_n^\lambda$ for the forces of $\boldsymbol{\tau}_\mu$ and $\boldsymbol{\tau}_\lambda$, window (m, w) contributes

$$\begin{aligned} \mathbf{A}_{mw} &= \Delta t \sum_f \phi_w(t_f) \sum_n \psi_m(\mathbf{X}_n)\big[\,\mathbf{f}_n^\mu(t_f)\ \ \mathbf{f}_n^\lambda(t_f)\,\big], \\ \mathbf{b}_{mw} &= \Delta t \sum_n \psi_m(\mathbf{X}_n)\,m_n\Big[\sum_f \ddot{\phi}_w(t_f)\,\mathbf{x}_n(t_f) - \mathbf{g}\sum_f \phi_w(t_f)\Big], \end{aligned} \tag{10}$$

three rows each, one per spatial component, for $3MW = 6000$ rows in total. All assembly and solving is done in double precision. We assemble on this discretization because the estimated material is used, and evaluated, in an MLS-MPM simulator. For another simulator, the same rows are assembled on that simulator's discretization instead, and nothing else in the solve changes.

B.4 Contact as an Unknown of the Solve

Every sequence we evaluate involves an impact, and the traction the ground applies is not observed. Because the governing equation is linear in that traction, we place it in the same linear system as additional unknowns, one three-vector per contacting node and frame, and solve for it jointly with (μ, λ). A node counts as contacting in frame f if its height is below $(j_g + 1.5)\,\Delta x_b$, where Δx_b is the grid spacing of the benchmark simulator (20 mm on PAC-NeRF, 12.5 mm on Spring-Gaus) and j_g is the index of the highest grid layer at or below the lowest observed point. The factor 1.5 is the support of the quadratic transfer kernel, so this band contains every material point the simulator's ground condition can reach. In the rows of component d, the unknown of node n at frame f has the column entries $\Delta t\,\phi_w(t_f)\,\psi_m(\mathbf{X}_n)$, and zero in the other two components. The contact force is thus estimated from the observed motion rather than prescribed by a surface normal, a friction cone or a coefficient, so the material estimate does not depend on which contact model a simulator would later use. The contact block is eliminated before the solve rather than stored. With $\mathbf{C}$ the matrix of contact columns, we take an orthonormal basis $\mathbf{Q}$ of its range from the eigendecomposition of $\mathbf{C}^\top\mathbf{C}$, keeping eigenvalues above 10^{-6} of the largest, and replace $\mathbf{A}$ and $\mathbf{b}$ by $(\mathbf{I} - \mathbf{Q}\mathbf{Q}^\top)\mathbf{A}$ and $(\mathbf{I} - \mathbf{Q}\mathbf{Q}^\top)\mathbf{b}$ in each component. Solving the projected two-column system gives the same (μ, λ) as solving jointly with all contact unknowns (the Frisch–Waugh–Lovell theorem), at the size of a problem without contact. This treatment is adopted from Hsu et al. (2022), which directly solves the material-point form with contact as an unknown of the same system.

B.5 Solve

We solve the projected system by linear least squares, and whenever the solution lies in the feasible range $\nu \in [0, 0.45]$ and $E \in [10^2, 10^9]$ Pa, which contains every material in both benchmarks, this unconstrained solution is the reported estimate. Otherwise we solve the same least squares subject to this range, and since E follows in closed form for fixed ν, this reduces to a one-dimensional bounded minimization over ν.

B.6 Interior Field Network

Input. In each frame, the visible surface nodes are found by ray casting against the surface mesh from the benchmark's own cameras, keeping only points inside a camera's view frustum. All other nodes, the occluded surface and the interior, are filled by harmonic interpolation. With the displacements of the visible nodes as boundary values, we solve the graph Laplacian of the tetrahedral mesh, with unit edge weights, for the remaining displacements, and the filled positions give a deformation gradient $\mathbf{F}_{\text{fill}}$ per element.

Graph and features. Nodes and elements form two vertex sets, and each element is connected to its four corners. Each node carries 20 features: its rest position normalized by the object size (3), a one-hot flag for visible or filled (2), and its displacement from rest at frames $t-2, \ldots, t+2$ (15). Each element carries its log rest volume and $\mathbf{F}_{\text{fill}} - \mathbf{I}$ (10), and each node–element incidence carries the offset of the corner from the element centroid in the rest and current configurations (6). Displacements and $\mathbf{F}_{\text{fill}} - \mathbf{I}$ are divided by a per-sequence strain scale s, the root-mean-square Green strain of the harmonic fill over the sequence, so that inputs from stiff and soft objects have comparable magnitude.

Architecture. The processor follows FEIH-GNN (Gao et al., 2024) and runs 10 rounds of message passing with width 128. In each round, every element is updated from its four corners and incidences, every incidence from its node and element, and every node from the mean of its incidences, all with residual connections. All update functions are MLPs with two hidden layers, SiLU activations and layer normalization at the output. The decoder is an MLP with two hidden layers and no output normalization, and returns 12 values per element. Six give a rotation $\mathbf{R}$ through the continuous 6D representation and six give a symmetric log-stretch $\mathbf{S}$, and the output is $\mathbf{F} = \mathbf{R}\exp(s\,\mathbf{S})$. The last decoder layer is initialized to zero, so the network starts from $\mathbf{F} = \mathbf{I}$. The network has 2.45 M parameters.

Loss. With $\mathbf{E} = \mathbf{F}_{\text{pred}}\mathbf{F}_{\text{gt}}^{-1}$ for each element, the loss combines a rotation term on the skew part of $\mathbf{E}$ (weight 10), the relative error of the symmetric log-strain (weight 1), a volume term on $\ln J$ (weight 0.2), and an alignment term $\max(0,\, 3 - \operatorname{tr}\mathbf{E})$ (weight 2) that also penalizes the 180° rotations the

skew term cannot see. The label $\mathbf{F}_{\text{gt}}$ is the constitutive deformation gradient of the simulator, averaged from its particles to each element.

Training data. The training set has 264 sequences on 27 Objaverse geometries, simulated with MLS-MPM and the Neo-Hookean model (time step 2×10^{-4} s, $\rho = 1000\,\text{kg}\,\text{m}^{-3}$) and recorded at 30 fps for about one second. The materials span E from 1.0×10^4 to 9.9×10^5 Pa and ν from 0.20 to 0.50, and the objects measure 0.19 to 0.44 m along their longest side. Validation uses 30 sequences on 3 held-out geometries, and no benchmark object appears in training.

Schedule. We train with Adam in two stages and keep the checkpoint with the lowest validation loss. The first stage trains on the full clean surface for 200 epochs with a cosine learning-rate schedule starting from 10^{-4}. The second stage fine-tunes the network for 40 epochs, again from 10^{-4} with cosine decay, while randomizing the observation. For each training sequence we precompute five partial observations, each seen from 1 to 3 camera directions drawn uniformly on the sphere and each with spatially correlated surface noise of median magnitude up to 3% of the object size, and every training step uses one of them with probability 0.5 and the full clean surface otherwise.

B.7 Benchmarks and Observation

Spring-Gaus-Diverse. The benchmark keeps the fourteen geometries and the generation settings of the released Spring-Gaus benchmark, MPM with a 12.5 mm grid and eight particles per cell, ten camera views and thirty frames at 512^2. It changes the materials, which cover E from 2.1×10^4 to 1.7×10^6 Pa and ν from 0.10 to 0.43.

Observation. EfM's input is taken from the released particle states of each benchmark. A surface node counts as visible when a ray from one of the benchmark's cameras (eleven on PAC-NeRF, ten on Spring-Gaus) reaches it before any other part of the surface.