DOI: 10.1111/cgf.70402
EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

STAGED: Stress-Tensor Assisted Global–local-global solver for interactive Elastic shape Design

Liangwang Ruan[1] and Bin Wang[2] and Tiantian Liu[2] and Baoquan Chen[3]†

[1]School of Computer Science, Peking University, China
[2] Independent Researcher, China [3] School of Intelligent Science and Technology, Peking University, China

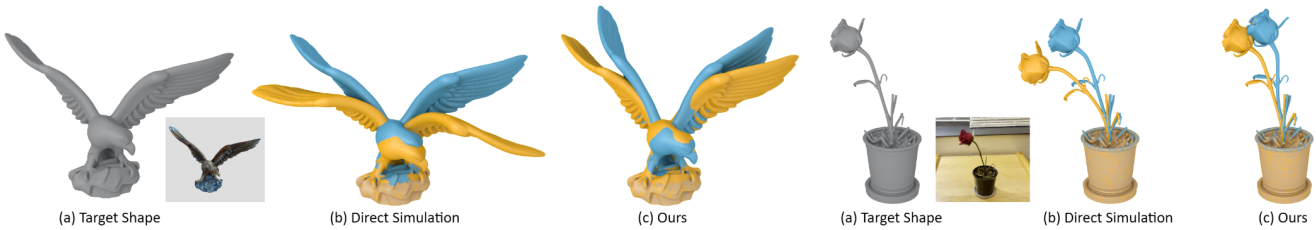


Figure 1: *Physics-aligned 3D reconstruction. (a) A 3D mesh reconstructed from a single-view image using generative model Hunyuan3D 2.5 [LZL*25]. (b) The assigned undeformed model (blue) sags under gravity, resulting in a mismatch between the simulated shape (yellow) and the observed target shape (gray). (c) Given the material properties and target shape (gray), our method efficiently computes a physically plausible rest shape (blue), producing simulated results (yellow) that closely align with the target (gray). Input images courtesy of [GWM*24, ZYW*24].*

Abstract

We present an efficient and scalable method for the inverse shape design problem of elastic objects, with broad applicability to diverse materials and interactive editing. The core idea is to decouple material nonlinearity from geometry optimization by introducing the Cauchy stress tensor as an auxiliary variable. We design a three-stage scheme that iteratively optimizes the stress tensors and the rest shape, with each stage being well-posed and efficiently-solvable. To address the lack of a theoretical convergence guarantee arising from the decoupled energy formulation, we incorporate a relaxation method that ensures robust stability in practice. As a result, our method achieves a 3× speedup over the state-of-the-art asymptotic method [Jia21] on a model with 40k vertices and 112k elements (Fig. 2), and exhibits near-linear scalability to large systems (Fig. 8). We demonstrate applications including rest shape design for various materials (ranging from standard models to complex spline-based materials [XSZB15]), interactive material and force editing, and elastic object reconstruction from images.

CCS Concepts

• ***Computing methodologies*** → ***Physical simulation;*** • ***Applied computing*** → ***Computer-aided design;***

1. Introduction

Elastic objects are ubiquitous in our surroundings, serving diverse functions under various external loads such as gravity, compression and distortion. To physically fabricate such objects, one must first determine their undeformed rest shape in the absence of external forces. Inverse shape design addresses this task systematically, eliminating the need for manual trial-and-error. Given an elastic object with a known constitutive model, the goal is to solve the static equilibrium equation:

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{X}) + \boldsymbol{f}_{\text{ext}} = 0, \tag{1}$$

where $\boldsymbol{f}$ is the internal elastic force derived from the constitutive model Ψ, $\boldsymbol{f}_{\text{ext}}$ represents external loads, $\boldsymbol{x}$ is the observed deformed shape, and $\boldsymbol{X}$ is the unknown rest shape to be recovered.

Eq. 1 also arises in forward simulation, but with $\boldsymbol{x}$ unknown

† corresponding author

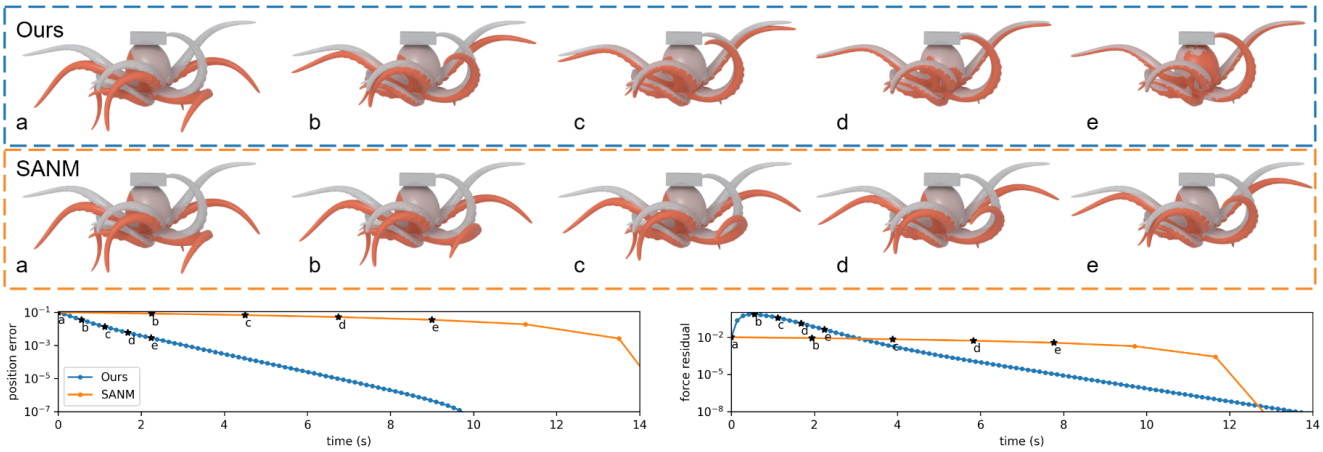


Figure 2: *Convergence analysis in comparison to SANM [Jia21]. An octopus model with its top fixed is inverted from a gravity-balanced state (orange) to its gravity-free rest shape (gray) with both our method and SANM. Our method rapidly reduces the position error within first few iterations, whereas SANM converges more slowly. Including precomputation, our method takes only 5.1 seconds to reduce the RMS position error below* 10^{-3} *(around* $1/1000$ *of the model scale), compared to 15 seconds for SANM, achieving a 3× speedup.*

and $\boldsymbol{X}$ given. In that case, Eq. 1 can be reformulated as a scalar energy minimization problem because $\boldsymbol{f}$ is conservative with respect to $\boldsymbol{x}$. This property enables the use of efficient optimization-based solvers [BML*14, LBK17]. In the inverse setting, $\boldsymbol{X}$ is unknown, $\boldsymbol{f}$ is no longer conservative w.r.t $\boldsymbol{X}$. This fundamental difference prevents the direct application of fast forward solvers. Previous methods have attempted to address this challenge through least-squares optimization [TKA11, WWY*15], Lagrangian multiplier [STBG12, ZCT21], and asymptotic expansion [CZXZ14, Jia21]. However, these approaches often suffer from slow convergence due to dense Hessians [TKA11], larger indefinite system due to Lagrangian multiplier [ZCT21], or limited generality for downstream applications [Jia21].

In contrast, we tackle this challenge by reformulating Eq. 1 into a multi-stage optimization problem and designing efficient solvers for each stage. Inspired by the recently proposed sag-free initializing method [HTYW22] and multi-stage solvers in forward simulations [BML*14, BN21], we introduce the Cauchy stress tensor $\boldsymbol{\sigma}$ as auxiliary variables, reformulating Eq. 1 as:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\sigma}(\boldsymbol{x}, \boldsymbol{X})) + \boldsymbol{f}_{\text{ext}} = 0. \tag{2}$$

Rather than solving directly for $\boldsymbol{X}$, we alternate between updating $\boldsymbol{X}$ and $\boldsymbol{\sigma}$ via Eq. 2. This splitting strategy offers two key advantages. First, the force can be seen as a function of $\boldsymbol{\sigma}$ instead of $\boldsymbol{X}$. This function is much simpler, as it bypasses the constitutive model. Second, The relation between $\boldsymbol{\sigma}$, $\boldsymbol{x}$ and $\boldsymbol{X}$ can be localized to every element, because $\boldsymbol{\sigma}$ depends only on the deformation gradient $\mathbf{F}$. This localized relation enables parallel computation. The introduction of $\boldsymbol{\sigma}$ decomposes the original problem into two subproblems. Each subproblem becomes simpler and can be solved with more efficient solvers. In the context of forward simulations, this idea closely aligns with the philosophy of Projective Dynamics (PD) [LBOK13, BML*14, NOB16].

Based on this formulation, we design a three-stage solver as detailed in Sec. 3: (1) *Force stage*: update stress tensor $\boldsymbol{\sigma}$ given force $\boldsymbol{f}$; (2) *Stress stage*: update local deformations given $\boldsymbol{\sigma}$; (3) *Geometry stage*: update rest shape $\boldsymbol{X}$ by "stitching" local deformations. These three stages are iterated until convergence. The force and geometry stages require global linear solves and the stress stage is fully localized and parallelizable. We therefore call this three-stage solver a global-local-global solver. Though each stage is well-formulated, the combined multi-stage solver is not theoretically guaranteed to converge, as there is no consistent global energy to optimize. To mitigate this problem, we further introduce a relaxation strategy as detailed in Sec. 3.4, enabling our method to stably converge in practice.

As a result, our three-stage solver achieves a 3× acceleration compared to the previous state-of-the-art inverse shape design method [Jia21] in the example of Fig. 2, while also demonstrating near-linear scalability (Fig. 8). Moreover, since only the stress stage requires a constitutive model, our framework can easily incorporate a wide range of isotropic materials, including stable Neo-Hookean [SGK18], StVK [SB12], and complex spline-based materials [XSZB15]. The method naturally prevents unphysical element inversions and over-stretching artifacts during the optimization. These properties enable diverse applications: fast rest shape design for large-scale systems, interactive editing of material properties and external forces, and integration with 3D generation model like Hunyuan3D [LZL*25] to reconstruct physically plausible rest shapes from images. In summary, our inverse shape design solver makes the following contributions:

1. a splitting formulation of the inverse design problem that decouples material nonlinearity from geometry optimization;
2. a three-stage solver to solve the subproblems, validated through various experiments;

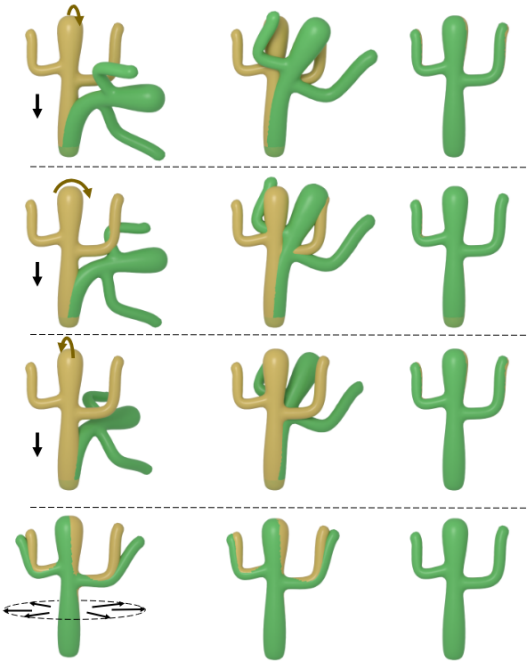


Figure 3: *The same cactus model (yellow) has different static shapes (green, first column) under different external loads and initial perturbations. In the first three rows, the cactus is subjected to the same gravity but with different initial perturbations, while in the last row, the cactus is subjected to a centrifugal force. These four rows therefore exhibit different static shapes to begin with. Regardless of the different static shapes, our method consistently recovers the same rest shape (green, third column) that matches the ground truth shape. The second row shows intermediate states during the convergence process.*

3. an efficient implementation that supports editing of both material properties and external forces interactively.

2. Related Work

2.1. Inverse Shape Design for Animation

Inverse shape design for elastic objects has been a longstanding research topic in computer animations [TKA11, DJBDDT13, HTYW22]. Artists often specify the static equilibrium state of an object(like flowers or hairs), which can lead to unwanted sagging artifacts when gravity is directly applied to the equilibrium state. Twigg *et al*. [TKA11] formulate this problem for mass-spring systems and propose an algorithm to determine the proper rest length for each spring. Hadap *et al*. [Had06] use a multi-body reduced model from robotics to compute the zero-gravity rest shape of strands. Derouet *et al*. [DJBDDT13] further account for internal frictional contact in hairs, introducing a convex second-order cone quadratic program (CSOCQP) solver to convert hair geometry into a physics-based model. Ly *et al*. [LCBD*18] study the inverse design for elastic shells, also considering frictional contact. Takahashi *et al*. [TB25] employ Gauss-Newton iterations to solve the inverse design problem for the Discrete Elastic Rod(DER) model of hairs. These methods were specifically tailored for slender structures such as hair or thin shells. More generally, Hsu *et al*. [HTYW22] propose a two-stage method for sag-free elastic initialization. Their key observation is that removing sagging artifacts only requires local rest shape, such as spring rest length or tetrahedral deformation, rather than a full global rest shape. This insight greatly simplifies the inverse shape design problem and reduces geometry nonlinearity. Hsu *et al*. [HWP*23] further extend this technique to strand-based hair simulation. Despite its simplicity and efficiency, this formulation can not recover a global rest shape, which limits its applicability in animation. We'll discussion the differences of our work and [HTYW22] in Sec 4.1.5.

2.2. Fabrication-oriented Design

The goal of fabrication-oriented design is to produce geometries, mechanical structures, and material distributions that satisfy the fabrication constraints and achieve desired physical functionalities. Depending on the tasks and settings, various methods have been proposed, such as 3D printing objects that can remain stable under gravity [PWLSH13], designing balloons that inflate into prescribed shapes [STBG12], or bending wires to match a target curve [TSBU24], etc. In this work, we focus on solving Eq. 1, i.e. recovering rest shapes with given static shapes and elastic materials. Wang *et al*. [WWY*15] formulate Eq. 1 as a least squares optimization problem and solve it using Newton's method. Because the mapping from $\boldsymbol{f}$ to $\boldsymbol{X}$ is highly nonlinear, squaring Eq. 1 produces an ill-conditioned problem that is difficult to optimize. For simplicity, Mukherjee *et al*. [MWW18] directly apply the Newton–Raphson method to solve Eq. 1, but this leads to a non-symmetric Hessian matrix equation in each iteration and still suffers from sever material nonlinearity. Instead of treating $\boldsymbol{x}$ as known, some works jointly solve for both $\boldsymbol{x}$ and $\boldsymbol{X}$ [STBG12, STK*14, ZCT21], optimize the consistency between $\boldsymbol{x}$ and the target deformed shape while treating force equilibrium as constraints. For constraint handling, Skouras *et al*. [STBG12] uses the Augmented Lagrangian method, Skouras *et al*. [STK*14] and Zehnder *et al*. [ZCT21] use SQP (Sequential Quadratic Programming). These methods introduce more free variables to optimize, or even indefinite system when using SQP. In contrast, Chen *et al*. [CZXZ14] and Jia *et al*. [Jia21] propose asymptotic methods to solve Eq. 1. Their core idea is to approximate the local nonlinearity of Eq. 1 using polynomial expansions, thereby avoiding the ill-conditioning that often arises in optimization-based approaches. Asymptotic methods achieve the fastest convergence speed among existing techniques for inverse shape design, but they require closed-form constitutive models, limiting their generality. Recently, although the framework of [GWM*24] has been developed to reconstruct 3D physical objects from a single image, it still rely on the conventional first-order stochastic gradient descent for optimization. In contrast to these works, without requiring costly global nonlinear optimization, our method employs a multi-staged strategy to decouple material nonlinearity from optimizations, achieving more efficient convergence.

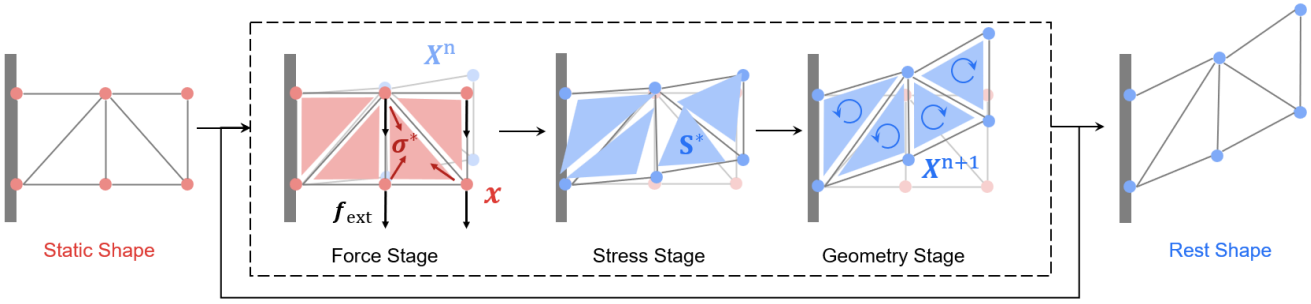


Figure 4: *Pipeline overview: given an input static shape, our algorithm iteratively performs the three stages described in Sec. 3 until converging to the final rest shape.*

2.3. Local-Global Solver

For forward elastic simulation, a variety of efficient optimization-based solvers have been developed [BML*14, LBK17, LGL*19, RWLC24]. Among them, we mainly draw inspiration from the idea of local–global iterations [BML*14]. Liu *et al*. [LBOK13] introduce spring directions as auxiliary local variables that are updated iteratively alongside global positions. Although this introduces additional DoFs, the subproblem of local and global variables are both straightforward to solve, yielding orders of magnitude speedups over Newton's method. Building on this idea, Bouszis *et al*. [BML*14] formulate Projective Dynamics (PD), extending the local-global framework to a broader range of problems, including elastic bodies, clothes, UV mapping, etc. Narain *et al*. [NOB16] prove that PD can theoretically be viewed as a special case of the Alternating Direction Method of Multipliers (ADMM), and extend it to handle more nonlinear materials. Liu *et al*. [LBK17] alternatively interpret PD as a quasi-Newton method, and introduce L-BFGS to simulate general materials. Brandt *et al*. [BEH18] combines PD with scalable sparse subspace bases, achieving real-time performance for large scale elastic animations. Brown *et al*. [BN21] observe that PD's performances degrades significantly in quasi-static problems with large rotations, and propose rotation-free nonlinear local variables as a remedy. More recently, Hsu *et al*. [HTYW22] adopt the local-global framework for sage-free initialization in general elastic simulations. Unlike iterative solvers such as ours, their method performs only a single global step and followed by a local step, producing approximate per-element rest shapes rather than a true global rest shape, as detailed in Sec. 4.1.5.

3. Method

For an elastic object discretized with a linear tetrahedral mesh (V,E), given its static equilibrium configuration $\boldsymbol{x} \in \mathbb{R}^{3|V|}$, constitutive model $\Psi(\cdot)$, and external forces $\boldsymbol{f}_{\text{ext}} \in \mathbb{R}^{3|V|}$, our objective is to recover the undeformed rest shape $\boldsymbol{X} \in \mathbb{R}^{3|V|}$. Starting from an initial guess $\boldsymbol{X}^0$, the algorithm iteratively performs three stages until the rest shape $\boldsymbol{X}^k$ converges:

1. Force Stage: compute the Cauchy stress field from $\boldsymbol{X}^k$, and update it to $\boldsymbol{\sigma}^*$ such that equilibrium with the external force $\boldsymbol{f}_{\text{ext}}$ is satisfied.
2. Stress Stage: compute the local deformation of each tetrahedron according to its constitutive model $\Psi(\cdot)$ to obtain the desired stress tensor $\boldsymbol{\sigma}^*$. This stage is executed in parallel.
3. Geometry Stage: update the global configuration from $\boldsymbol{X}^k$ to $\boldsymbol{X}^{k+1}$ based on the computed local deformations.

The pipeline is illustrated in Fig. 4 with its pseudo code in Alg. 1. Note that by introducing multiple stages, our method does not optimize a consistent global energy. We will explain each stage in detail in the following sections.

3.1. Force Stage

Given the rest shape $\boldsymbol{X}^k$ from the previous iteration and the static shape $\boldsymbol{x}$, we compute the deformation gradient $\mathbf{F}_e \in \mathbb{R}^{3\times3}$ for each element $e \in E$, along with the Cauchy stress tensor $\boldsymbol{\sigma}_e \in \mathbb{R}^{3\times3}$ defined as [SB12]:

$$\boldsymbol{\sigma}_e = \frac{1}{J_e}\mathbf{P}_e\mathbf{F}_e^\top = \frac{1}{|\det \mathbf{F}_e|}\frac{\partial \Psi}{\partial \mathbf{F}_e}\mathbf{F}_e^\top, \tag{3}$$

where $\mathbf{P}_e = \frac{\partial \Psi}{\partial \mathbf{F}_e}$ is the first Piola-Kirchhoff stress tensor of element e. The Cauchy stress describes the stress state in the deformed configuration and is used to compute the elastic forces on vertices:

$$\boldsymbol{f} = \mathbf{B}\boldsymbol{\sigma}, \tag{4}$$

where $\boldsymbol{f} \in \mathbb{R}^{3|V|}$ is the vector of elastic forces on all vertices, $\boldsymbol{\sigma} \in \mathbb{R}^{9|E|}$ is the vectorized form of the Cauchy stress tensors over all elements, and $\mathbf{B} \in \mathbb{R}^{3|V|\times9|E|}$ is a linear mapping matrix that depends solely on the deformed shape $\boldsymbol{x}$. For the precise definition of $\mathbf{B}$, please refer to the Appendix. Since the rest shape is fixed, $\mathbf{B}$ remains constant throughout the optimization.

Because the tetrahedral mesh typically contains more elements than vertices, Eq. 2 forms an underdetermined system for $\boldsymbol{\sigma}$. The force stage adjust the stress $\boldsymbol{\sigma}^k$ (from $\boldsymbol{X}^k$) to satisfy equilibrium with external forces, leading to the following weighted least-squares problem:

$$\min \|\boldsymbol{\sigma}^* - \boldsymbol{\sigma}^k\|_{\mathbf{K}}^2 \; s.t. \; \mathbf{B}\boldsymbol{\sigma}^* + \boldsymbol{f}_{\text{ext}} = 0, \tag{5}$$

where $\mathbf{K} \in \mathbb{R}^{9|E|\times9|E|}$ is a diagonal weighting matrix with $\mathbf{K}_e = v_e^2\mathbf{I}$

Algorithm 1 Inverse Shape Design

Require: static shape $\boldsymbol{x}$, external force $\boldsymbol{f}_{\text{ext}}$, constitutive model Ψ
Ensure: rest shape $\boldsymbol{X}$
Assemble $\mathbf{B}$ and factorize $\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top$ in Eq. (6)
$\boldsymbol{X}^0 \leftarrow \boldsymbol{x}$
repeat
  ▷ *Force stage* ◁
  $\boldsymbol{\sigma}^k \leftarrow \boldsymbol{\sigma}(\boldsymbol{x}, \boldsymbol{X}^k)$
  $\boldsymbol{\sigma}^* \leftarrow$ solve Eq. (13)
  ▷ *Stress stage* ◁
  for $e \in E$ **do** ▷ *in parallel*
    $\mathbf{U}_e^*, \mathbf{D}_e^* \leftarrow \text{SVD}(\boldsymbol{\sigma}_e^*)$
    $\boldsymbol{\Sigma}_e \leftarrow$ solve Eq. (9)
    $\mathbf{S}_e^* \leftarrow \mathbf{U}_e^* \boldsymbol{\Sigma}_e (\mathbf{U}_e^*)^\top$
  ▷ *Geometry stage* ◁
  $\boldsymbol{X}^{k+1} \leftarrow$ solve Eq. (12)
until $\|\boldsymbol{X}^{k+1} - \boldsymbol{X}^k\| \le \varepsilon$
return $\boldsymbol{X}^{k+1}$

for each element, and v_e is the volume of element e in the static configuration. The constraint in Eq. 5 is the force balancing condition in static shape, the minimization means the updated $\boldsymbol{\sigma}^*$ should stay close to the previous $\boldsymbol{\sigma}^k$, and $\mathbf{K}$ is intuitively chosen for measuring the volume-integrated differences of stress fields. Mathematically, this formulation projects $\boldsymbol{\sigma}^k$ onto the affine manifold of force-balanced stresses.

The optimization problem in Eq. 5 is a standard quadratic programming problem with only equality constraints, which has a closed form solution:

$$\boldsymbol{\sigma}^* = \boldsymbol{\sigma}^k - \mathbf{K}^{-1}\mathbf{B}^\top \left(\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top\right)^{-1} \left(\boldsymbol{f}_{\text{ext}} + \mathbf{B}\boldsymbol{\sigma}^k\right). \tag{6}$$

The main computational cost lies in solving the linear system with matrix $\mathbf{A} = \mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top$. Fortunately, since $\mathbf{B}$ and $\mathbf{K}$ are constant and $\mathbf{A}$ is positive semi-definite, its Cholesky factorization can be precomputed, greatly accelerating the evaluation of Eq. 6. To enforce stress symmetry(Newton's third law), each $\boldsymbol{\sigma}_e$ is parameterized by its six unique components (diagonal and upper-triangular part), with $\mathbf{B}$ and $\mathbf{K}$ modified accordingly. This reduces the dimension of $\mathbf{A}$ from $9|E|$ to $6|E|$, resulting in a performance gain. For more details, please refer to the Appendix.

3.2. Stress Stage

The stress stage recovers element-wise deformation gradients $\mathbf{F}_e$ that reproduce the target stresses $\boldsymbol{\sigma}_e^*$ from the force stage. Since elements are independent, this computation is naturally parallelizable. For notational simplicity, we omit the subscript e for all quantities in this section.

For isotropic materials, the constitutive model is rotational-invariant:

$$\Psi(\mathbf{F}) = \Psi(\mathbf{F}\mathbf{R}) = \Psi(\boldsymbol{\Sigma}), \tag{7}$$

where $\mathbf{R}$ is a rotation and $\mathbf{F} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ is the singular value decomposition of $\mathbf{F}$. The first Piola–Kirchhoff stress $\mathbf{P}$ is $\mathbf{P} = \mathbf{U}\frac{\partial \Psi}{\partial \boldsymbol{\Sigma}}\mathbf{V}^\top$, leading to the following expression for the Cauchy stress:

$$\boldsymbol{\sigma} = \frac{1}{J}\mathbf{P}\mathbf{F}^\top = \mathbf{U}\frac{1}{J}\frac{\partial \Psi}{\partial \boldsymbol{\Sigma}}\boldsymbol{\Sigma}\mathbf{U}^\top, \tag{8}$$

where $J = \det\mathbf{F} = \det\boldsymbol{\Sigma}$. This equation can be interpreted as the eigen decomposition of $\boldsymbol{\sigma}$, whose eigenvalues are given by $\frac{1}{J}\frac{\partial \Psi}{\partial \boldsymbol{\Sigma}}\boldsymbol{\Sigma}$. With the target stress $\boldsymbol{\sigma}^*$ and its eigen decomposition $\boldsymbol{\sigma}^* = \mathbf{U}^*\mathbf{D}^*(\mathbf{U}^*)^\top$ provided, we solve for $\boldsymbol{\Sigma}$ via:

$$\frac{1}{J}\frac{\partial \Psi}{\partial \boldsymbol{\Sigma}}\boldsymbol{\Sigma} = \mathbf{D}^*. \tag{9}$$

For certain constitutive models like the one used in [HWP*23], this equation has a closed-form solution. But for general materials, we have to numerically solve it. We employ the Powell's hybrid method from the MINPACK [MSHG84] library to solve this equation, which requires only the residual and gradient of Eq. 9. This allows our method to support any analytical constitutive models expressed in terms of singular values of $\mathbf{F}$, including Neo-Hookean [SGK18], StVK [SB12], as well as more complex materials like spline-based [XSZB15] or even neural-network-based materials [MCD*23], so long as the first Piola stress tensor and its gradient can be numerically evaluated. If a valid solution to Eq. 9 can't be found, we assign the identity matrix to $\boldsymbol{\Sigma}$, though such cases are rare in practice. Strain limiting is enforced by imposing box constraints on $\boldsymbol{\Sigma}$, thereby preventing unphysical element inversion or excessive stretching.

Once $\boldsymbol{\Sigma}$ and $\mathbf{U}^*$ are obtained, $\mathbf{V}$ remains undetermined. Rewriting,

$$\mathbf{F} = \mathbf{U}^*\boldsymbol{\Sigma}\mathbf{V}^\top = \left[\mathbf{U}^*\boldsymbol{\Sigma}(\mathbf{U}^*)^\top\right]\left[\mathbf{U}^*\mathbf{V}^\top\right] = \mathbf{S}^*\mathbf{R}, \tag{10}$$

where $\mathbf{S}^* = \mathbf{U}^*\boldsymbol{\Sigma}(\mathbf{U}^*)^\top$ is known, and $\mathbf{R}$ is an arbitrary rotation. This reveals that only the symmetric part of $\mathbf{F}$'s polar decomposition is determined at this stage, while the rotational part is left free. This reflects the fact that rigid rotations of the rest configuration do not affect stresses or forces in the deformed state. The specific rotation $\mathbf{R}$ will be resolved in the subsequent geometry stage based on additional constraints.

3.3. Geometry Stage

From the stress stage, we obtain the local symmetric deformation $\mathbf{S}_e^*$ independently for each element. These local deformations should be "stuck" together, with individual rotations, to get a global deformation. The geometry stage then seeks a global rest shape $\boldsymbol{X}^{k+1}$ consistent with these local deformations. We formulate this as a pure geometry optimization problem. Instead of tracking the deformation gradient from the rest shape to the deformed configuration $\mathbf{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}$, we consider the inverse mapping $\mathbf{N} = \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{x}}$. With polar decomposition $\mathbf{N}_e = \mathbf{R}_e\mathbf{S}_e$, we enforce consistency by matching $\mathbf{S}_e$ with $(\mathbf{S}_e^*)^{-1}$. Following [TKL22], we formulate the objective in mixed variational form:

$$\begin{aligned} W(\boldsymbol{X}, \mathbf{S}, \boldsymbol{\lambda}) &= \sum_e \left[\frac{1}{2}\|\mathbf{S}_e - (\mathbf{S}_e^*)^{-1}\|^2 - \boldsymbol{\lambda}_e^\top \boldsymbol{c}_e\right] v_e, \\ \boldsymbol{c}_e(\boldsymbol{X}, \mathbf{S}) &= \text{Polar}(\mathbf{N}_e(\boldsymbol{X})) - \mathbf{S}_e. \end{aligned} \tag{11}$$

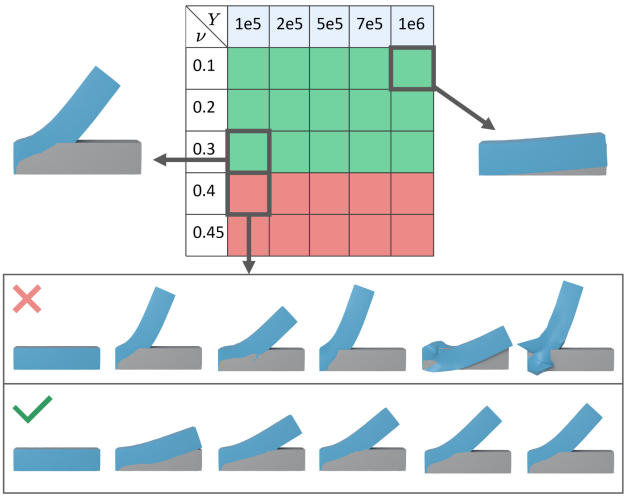


Figure 5: *Ablation study of relaxation (Sec. 3.4). Bar model with left end fixed under gravity (gray). Our algorithm recovers its rest shape (blue) across varying Young's modulus Y and Poisson's ratios ν. The table reports the convergence result without relaxation: green (success) and red (failure). Our solver struggles to converge when $\nu \geq 0.4$ due to numerical instability. Once relaxation is added back, our solver becomes robust across all tested parameters. The bottom figure shows the converging sequence when $\nu = 0.4$, the first row (without relaxation) fails to converge, while the second row (with relaxation) converges successfully.*

Here $\mathbf{S}$ is treated as an independent variable and $\boldsymbol{\lambda}$ as a Lagrangian multiplier. The optimization problem is then:

$$\min_{\boldsymbol{X},\mathbf{S}} \max_{\boldsymbol{\lambda}} W(\boldsymbol{X},\mathbf{S},\boldsymbol{\lambda}). \tag{12}$$

This formulation implicitly incorporates rotations through the constraint term, enabling efficient handling of local rigid motions. Using sequential quadratic programming (SQP) with Schur complement reduction, the problem simplifies to solving linear systems in terms of $\boldsymbol{X}$ along, with $\mathbf{S}$ and $\boldsymbol{\lambda}$ eliminated. Since Eq. 11 matches the formulation of [TKL22], we refer the readers to that work for numerical details.

3.4. Relaxation

Although each stage is well-formulated, the algorithm may fail to converge in certain scenarios. The main reason is that we do not have a consistent global energy to optimize, each stage only solves a subset of unknown parameters. As illustrated in Fig. 5, when fixing the left end of a horizontal bar and computing its gravity-free rest shape, convergency fails for Poisson's ratios ≥ 0.4. For instance, at $\nu = 0.4$, the bar exhibits oscillatory behavior followed by divergence, indicative of overshooting during iterative updates. This instability parallels well-known issues in nonlinear forward simulations, where overly aggressive descent steps inject spurious energy into the system, leading to numerical blow-up. A standard stabilization technique in nonlinear solvers is step-size control via

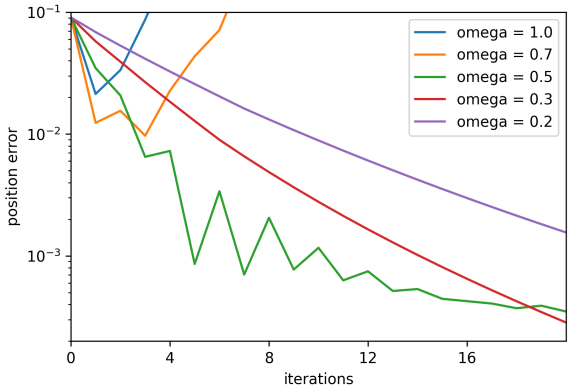


Figure 6: *Convergence behavior with different relaxation parameters ω in the bar example(Fig. 5). Smaller ω leads to smoother but slower convergence. We adopt $\omega = 0.3$ to balance robustness and efficiency.*

relaxation (or damping), which mitigate overshooting by scaling updates. We apply relaxation to $\boldsymbol{\sigma}$ in Eq. 6 as:

$$\boldsymbol{\sigma}^* = \boldsymbol{\sigma}^k - \omega \mathbf{K}^{-1}\mathbf{B}^\top \left(\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top\right)^{-1} \left(\boldsymbol{f}_{\text{ext}} + \mathbf{B}\boldsymbol{\sigma}^k\right). \tag{13}$$

with $\omega \in (0,1]$ the relaxation parameter. Unlike forward simulation, where ω can be adaptively selected via line search, inverse shape design lacks a variational objective that permits such adjustment. We therefore determine ω empirically. As shown in Fig. 6, decreasing ω suppresses oscillations, while excessively small values overdamp progress. In all experiments, we fix $\omega = 0.3$, which consistently achieves stable convergence with negligible efficiency loss across all tested scenarios.

4. Results

In this section, we show validation tests and applications of our method. We implement our algorithm using C++ with Eigen [GJ*10] and MKL [Int24] library to accelerate linear algebra computation. We use OpenMP [DM98] to parallelize the stress stage and the linear system assembly in the geometry stage. We use pybind11 [J*24] to create python frontend, and Taichi [HLA*19] library for GUI. All experiments, including comparison with [Jia21] are performed on an AMD Ryzen 9 7950X 16-Core Processor. For all examples, we use density $\rho = 10^2 \text{kg/m}^3$ and gravity $g = 9.8\text{m/s}^2$. The specific configuration for each example is listed in Tab. 1. Without special statement, we use RMS displacement error below $\varepsilon = 10^{-3}$ in line 14 of Alg. 1 as our stop criteria. All the models in the experiments are normalized to around 1m size.

4.1. Validation

4.1.1. Correctness

We validate the correctness of our method by forward simulation. Starting from an initial shape A, we first run a forward simulation

Table 1: *Simulation parameters for each example. $|V|$, $|E|$: are the number of mesh vertices and elements. SNH: stable Neo-Hookean [SGK18]. NH: Neo-Hookean [Jia21]. Coro: corotational material model [SB12]. Spline: spline-based material model [XSZB15]. Time: total optimization time include precomputation.*

Example	$\|V\|$	$\|E\|$	Material	Young's modulus (Pa)	Poisson's ratio	Time(s)
octopus (Fig. 7)	40k	112k	NH	1e5	0.45	5.1
bar (Fig. 5)	0.6k	2k	SNH	1e5-1e6	0.1-0.45	0.2-0.3
cactus (Fig. 3)	5k	24k	SNH	6e4	0.45	1.4
armadillo (Fig. 8)	4k-77k	15k-340k	NH	4e5	0.45	1.4-50.1
plant (Fig. 11)	10k	33k	SNH/Coro/StVK/Spline	1e6	0.1-0.45	1.5-1.8
eagle (Fig. 1)	16k	57k	SNH	5e5	0.45	2.6
rose (Fig. 1)	39k (24k fixed)	149k	SNH	1e8	0.45	3.5

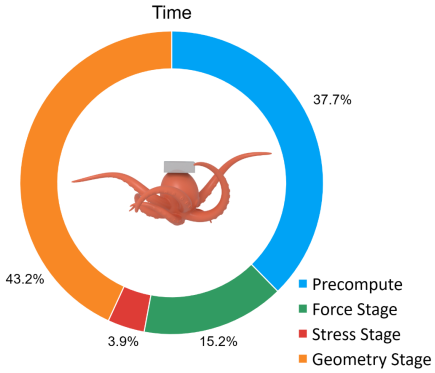


Figure 7: *Time breakdown for the octopus example(Fig. 2). The geometry stage dominates the runtime, followed by precomputation.*

to reach an equilibrium state B. Then we run our algorithm from B to get the corresponding rest shape C using the same material. The same initial shape A might converge to different equilibrium state B due to different initial perturbations, which further complicates this problem. As shown in Fig. 3, the cactus model can bend to different directions (first three rows) under the same gravity. In the last row of Fig. 3, we also show a case loaded with a centrifugal force, which points radially outward from the y-axis with length proportional to the distance from the y-axis, acting on the cactus model. For all cases, our method can converge to the same rest shape C, which is indistinct to the ground truth shape A, showing the correctness and robustness of our method.

4.1.2. Efficiency

A detailed time breakdown of the octopus example(Fig. 2) is presented in Fig. 7. In the precomputation phase, the mesh is loaded, $\mathbf{B}$ is assembled and $\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top$ in Eq. 13 is prefactorized. Since the Hessian matrix of the SQP problem in Eq. 11 shares the same sparsity pattern as $\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^\top$, we pre-assemble a sparse Hessian for Eq. 11 and setup a mapping table linking vertex pairs to corresponding entries in the Hessian. This mapping facilitates efficient parallel Hessian assembly during runtime. In the multi-stage iterative process, both the force stage(benefiting from a closed-form solution) and the stress stage(executed in parallel) contribute only marginally to the total computation time. The majority of the time is spent in the geometry stage, which involves optimizing the nonlinear energy in Eq. 11.

It is worth noting that Eq. 11 is not the only possible formulation. An alternative is the PD-like energy minimization defined as:

$$\min_{\boldsymbol{X}} W_{PD}(\boldsymbol{X}) = \min_{\boldsymbol{X}} \sum_e \min_{\mathbf{R}_e} \frac{1}{2} \|\mathbf{N}_e(\boldsymbol{X}) - \mathbf{R}_e(\mathbf{S}_e^*)^{-1}\|^2 v_e, \quad (14)$$

where $\mathbf{R}_e$ is the local rotation of each element. Despite its conceptual simplicity, we demonstrate experimentally that our chosen formulation(Eq. 11) is more computationally efficient. As shown in Fig. 9, our solver converges significantly faster than both the standard projective dynamics (PD) [BML*14] and L-BFGS solver [LBK17] applied to this alternative PD-like energy in Eq. 14.

4.1.3. Convergence

We study the convergence behavior of our method in comparison with SANM [Jia21]. SANM is an optimized open-source implementation of symbolic asymptotic numerical solver, which converges orders of magnitude faster than Newton-type solvers such as the Levenberg-Marquardt solver for the inverse shape design problem [CZXZ14,Jia21]. Therefore, we choose SANM as our baseline. For the octopus model with 40k vertices and 112k elements, we obtain its ground truth gravity-free rest shape $\boldsymbol{X}_{\text{gt}}$ by running SANM until the force residual satisfies $\|\boldsymbol{f} + \boldsymbol{f}_{\text{ext}}\|_{RMS} < 10^{-12}$. At each iteration, we compute both the position error $\|\boldsymbol{X}^k - \boldsymbol{X}_{\text{gt}}\|_{RMS}$ and the force residual for our method and SANM, and present the result in Fig. 2. Our method efficiently reduces the position error within the first few iterations, while SANM struggles to make progress early on. Interestingly, we observe that the force residual for our method initially increases, even as the position error decreases. This phenomenon highlights the ill-conditioned nature of the inverse shape design problem and further supports our motivation to decouple material nonlinearity from geometry optimization.

4.1.4. Scaling Test

We conduct a scaling test to evaluate the efficiency of our method against the state-of-the-art multi-thread asymptotic solver SANM [Jia21]. As shown in Fig. 8, we use the armadillo model with different mesh resolution. In each case, the same bottom part of the model was fixed, and the task is to compute its gravity-free rest shape. We measured both the total solver time (including precomputation) and memory usage. For fairness, we employ the RMS

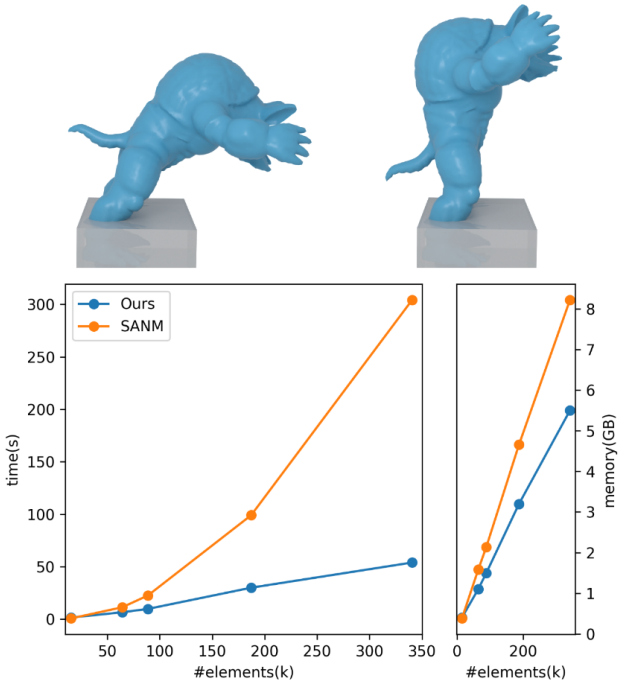


Figure 8: *Scalability of time and memory costs compared with SANM [Jia21]. Our method demonstrates near-linear scalability in both metric, while SANM incurs significantly higher cost on high resolution meshes.*

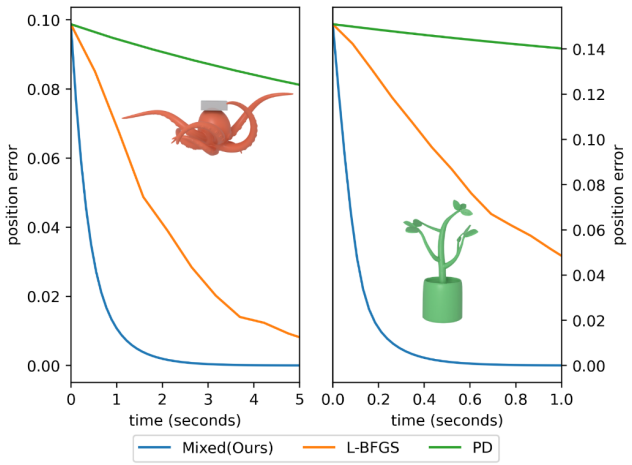


Figure 9: *Comparison of different solvers in the geometry stage. Mixed(ours): mixed variational formulation (Eq. 11). L-BFGS: L-BFGS solver [LBK17] for Eq. 14. PD: projective dynamics solver [BML*14] for Eq. 14. Our mixed solver achieves the fastest convergence.*

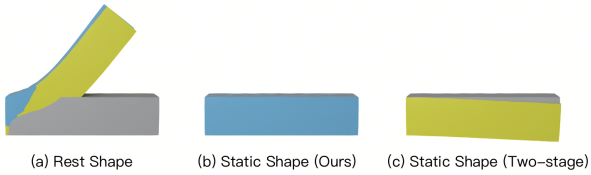


Figure 10: *Comparison with two-stage initialization [HTYW22]. The static shape of a hanging bar model with its left end fixed is given. (a) Our method (blue) and two-stage initialization [HTYW22] (yellow) produce different rest shapes. (b) With the rest shape from our method, the simulated static shape (blue) matches the input shape (gray). (c) With the rest shape from the two-stage initialization (yellow), the simulated static shape does not match the input shape (gray).*

force residual used in SANM as the stop criterion for both methods, and set the threshold to 10^{-4} as a practically low value.

Our method demonstrates near-linear scalability in both time and memory, while SANM requires substantially more time to process high resolution meshes. For the largest system, consisting of 77k vertices and 340k elements, our algorithm converges in only 50.1 seconds(including precomputation), compared to 301 seconds for SANM, yielding a 6× speedup. Moreover, our algorithm also exhibits a lower memory footprint than SANM in all cases.

4.1.5. Comparison to Two-stage Initialization [HTYW22]

As explained in Sec. 2.3, our method draws inspiration from the two-stage initialization method [HTYW22]. The two-stage initialization method aims to remove sagging artifact without acquiring a global rest shape. It operates in two parts: a global linear optimization solves a linear problem to enforce static equilibrium under the external loads, followed by local nonlinear corrections to compute each tetrahedron's local rest shape, i.e., the $\mathbf{D}_m$ matrix in $\mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1}$ for each tetrahedron. To obtain an approximate global rest shape $\tilde{\boldsymbol{X}}$, additional forward simulation from the static shape $\boldsymbol{x}$ under zero external forces is required. Note that $\tilde{\boldsymbol{X}}$ is only an approximation of the true rest shape $\boldsymbol{X}$, as internal elastic forces still exist. This highlights the key difference between our method and two-stage initialization: our method iteratively converges to the true rest shape $\boldsymbol{X}$, whereas the two-stage method can only produce an approximation, as demonstrated by the bar example in Fig. 10. Our improvement is enabled by three innovations: the relaxed projection formulation for updating $\boldsymbol{\sigma}$ in Eq. 13, the rotation analysis of deformation gradient in Sec. 3.2, and the introduction of the geometry stage in Sec. 3.3.

4.2. Application

4.2.1. Various Materials

Our method supports a wide range of isotropic material models. To evaluate this, we performed reconstructions from the same input static shape and under identical external force conditions, while varying only the constitutive model including stable Neo-

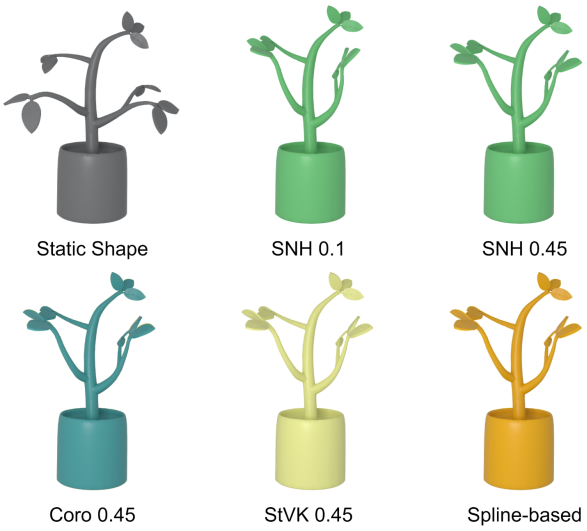


Figure 11: *Rest shape recovery of the plant model (gray: static shape) under different constitutive models (SNH, Coro, StVK, spline-based; see Tab. 1. Numbers indicate the Poisson's ratio.)*

Hookean (SNH) [SGK18], StVK [SB12], and Corotational material (Coro) [SB12] and the Poisson's ratio (from 0.1 to 0.45). As shown in Fig. 11, the resulting rest shape remains nearly identical across different materials with the same stiffness but significantly different Poisson's ratios. This demonstrates that the reconstruction is dominated primarily by the material's stiffness(Young's modulus), with minimal influence from the specific constitutive model or Poisson's ratio.

We further demonstrate the applicability of our method using a spline-based material model [XSZB15], with corresponding result shown in Fig. 11. Xu *et al.* [XSZB15] proposed a general formulation for isotropic material defined by three scaler functions f, g, and h:

$$\begin{aligned}\Psi(\lambda_0,\lambda_1,\lambda_2) = f(\lambda_0)+f(\lambda_1)+f(\lambda_2)\\ +g(\lambda_0\lambda_1)+g(\lambda_0\lambda_2)+g(\lambda_1\lambda_2)\\ +h(\lambda_0\lambda_1\lambda_2),\end{aligned} \tag{15}$$

where λ_i denote the singular values of the deformation gradient $\mathbf{F}$. In this formulation, each scalar function is defined by a spline curve, which reduces the complexity of the nonlinear material space to intuitive scale strain-stress relationships and greatly simplifying the material design process. Owing to our physics-geometry decoupled multi-stage strategy, the spline functions can be directly incorporated into the stress stage without requiring modifications to the other two stages. As illustrated in Fig. 11, we show the result using a spline-based material designed to emulate Neo-Hookean material behavior. For further details regarding spline definition and implementation, we refer readers to the original paper [XSZB15]. It should be noted that the SANM [Jia21] solver cannot accommodate such spline-based material, as it relies on an analytical expansion of the constitutive model, a requirement that spline-based functions do not meet.

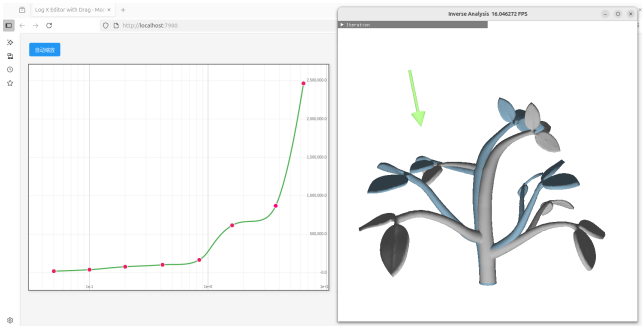


Figure 12: *Interactive editing interface. Users can adjust material splines, modify the direction and magnitude of external forces, and instantly preview the updated rest shape from the viewer.*

4.2.2. Interactive Editing

As a design tool, our method benefits from the multi-stage iterative strategy to enable efficient editing of material properties and external forces. The graphical interface(Fig. 12) allows users to adjust parameters, such as material splines or force conditions, and see the resulting rest shape update immediately. This interactivity is made possible by the core speed of our algorithm, which is further enhanced by a warm-start initialization from previous solutions. Although our algorithm also supports interactive editing of the static shape in theory, this functionality is more limited. Any change to the static shape requires the recomputation of the matrix $\mathbf{B}$ and the subsequent re-factorization of $\mathbf{B}\mathbf{K}^{-1}\mathbf{B}^{\top}$ in Eq. 13 as these matrices are static shape dependent. This additional step incurs a computational cost, hindering real-time performance.

4.2.3. Reconstruction from Image

Recently advances in the deep learning community have enabled the reconstruction of 3D models from a single image [XLX*24, LZL*25]. However, such reconstructions often overlook the fact that elastic objects deform under gravity. Our method can serve as an efficient post processing step to recover their undeformed shapes, which can then be used to create virtual twins of the real world objects. As shown in Fig. 1, we use Hunyuan3D 2.5 [LZL*25] to reconstruct a 3D mesh from a single image, tetrahedralize the mesh using Houdini [Sid23], and finally apply our algorithm to obtain its rest shape. In this example, we manually set the Young's modulus and Poisson's ratio of the material for demonstration purposes. These parameters can alternatively be obtained through real world measurement or estimated using visual-language foundation models such as GPT-4o [Ope25].

5. Conclusion & Limitation

In this work, we present a three-stage inverse shape design framework that efficiently decouples material nonlinearity from geometry optimization through the introduction of stress tensor variables.

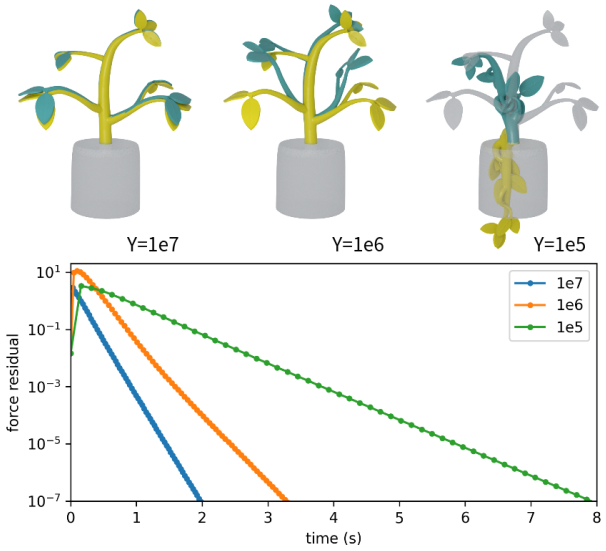
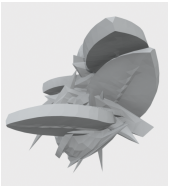


Figure 13: *Failure case. When the Young's modulus decreases from 10^7 to 10^5, our method still produces numerically valid rest shapes (the force residual in Eq. 1 can be sufficiently low). However, when the Young's modulus further decreases to 10^5 where no feasible rest-shape configuration exists, the resulting rest shape no longer follows the intended standing trend observed in the 10^6 and 10^7 cases. Instead, it collapses into a curled configuration, and the forward simulation result deviates significantly from the target shape although with its force residual converged.*

Our method achieves over a 3× speedup compared to previous methods [Jia21] on a model with 40k vertices and 112k elements, while supporting a wider range of materials, including user-defined spline-based materials, and enabling interactive editing of external forces and material properties. The decoupled formulation further allows parallel stress updates and geometry optimization, ensuring near-linear scalability to large meshes. Our experiments validate the physical correctness of the method and demonstrate applications in fabrication-oriented design and 3D reconstruction from images. Several future directions are worthy of exploration. First, the decoupling of stress and geometry also suggests opportunities for efficient joint material-shape optimization, as in [WWY*15], which is a critical step toward reconstructing physics-plausible twins of real-world objects. Second, currently we don't have collisions and frictions in our implementation, but in theory the same technique from [HTYW22] can be applied in our force stage to support frictional contacts. Last but not least, the $\mathbf{K}$ matrix in Eq. 5 is intuitively chosen for simplicity, whose impact on the convergence is not studied. A more rigorous analysis of the choice of $\mathbf{K}$ could potentially speed up convergence.

Our framework also has several limitations. First, it lacks a rigorous theoretical proof of convergence for the stress-geometry splitting strategy. We cannot guarantee Eq. 1 always has a solution. Second, even there is a solution, our method does not always converge to that solution. Although our experiments show that the relaxation mechanism can robustly improve convergence stability (with $\omega = 0.3$ ensuing convergency across all test cases), the relaxation parameter ω in Eq. 13 is still manually chosen and there is no theoretical guarantee of robustness under all conditions. Third, our method still struggles in certain extreme cases. As shown in Fig. 13, when the Young's modulus of the plant model is reduced to 10^5, the material becomes too soft that no reasonable rest shape can achieve the target shape under loading. In this case, our method converges to a nominal valid solution (with the force residual converged). However, the resulting rest shape does not continue the expected trend observed in the 10^6 and 10^7 cases, it collapses into a curled configuration, and the forward simulation produces a "sagging" static shape that deviates from the input "standing" shape. When the Young's modulus is further reduced to $1e4$, our algorithm fails to converge to any rest shape. This observation is partly due to the fact that the input shape, although force-balancing, is not a stationary static configuration. For comparison, SANM [Jia21] gives the same result as our method for Young's modulus 10^7, 10^6 and 10^5, and even finds an extremely ill-conditioned solution for the 10^4 case (force residual below 10^{-12}, shown on the right), but eventually fails for Young's modulus 10^3. Though SANM outperforms ours method in the 10^4 case, such unphysical scenario never exists in reality. A more comprehensive theoretical analysis of such ill-posed cases is an important direction for future work. Finally, our method currently supports only isotropic materials. Extending it to anisotropic materials, which are rotation-dependent, presents a significant challenge to core idea of stress-geometry splitting.



Appendix A: Force Stage Details

In this appendix we give a detailed deduction of Eq. 4. Following [SB12], for a single tetrahedron, we can compute the deformation gradient $\mathbf{F}$ as:

$$\begin{aligned} \mathbf{F} &= \mathbf{D}_s \mathbf{D}_m^{-1}, \\ \mathbf{D}_s &= \begin{pmatrix} \boldsymbol{x}_1 - \boldsymbol{x}_4 & \boldsymbol{x}_2 - \boldsymbol{x}_4 & \boldsymbol{x}_3 - \boldsymbol{x}_4 \end{pmatrix}, \\ \mathbf{D}_m &= \begin{pmatrix} \boldsymbol{X}_1 - \boldsymbol{X}_4 & \boldsymbol{X}_2 - \boldsymbol{X}_4 & \boldsymbol{X}_3 - \boldsymbol{X}_4 \end{pmatrix}. \end{aligned} \tag{A.1}$$

Then the relation of forces on each vertices and the first Piola-Kirchhoff $\mathbf{P}$ can be written as:

$$\begin{aligned} (\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3) &= -\mathbf{P}\mathbf{D}_m^{-\top} V_m, \\ \boldsymbol{f}_4 &= -\boldsymbol{f}_1 - \boldsymbol{f}_2 - \boldsymbol{f}_3, \end{aligned} \tag{A.2}$$

where $V_m = \det(\mathbf{D}_m)$ is the rest volume of the tetrahedron. Applying Eq. 3, we have:

$$\begin{aligned} (\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3) &= -\mathbf{P}\mathbf{D}_m^{-\top} V_m \\ &= -J\boldsymbol{\sigma}\mathbf{F}^{-\top}\mathbf{D}_m^{-\top} V_m \\ &= -\boldsymbol{\sigma}\mathbf{D}_s^{-\top} V_s. \end{aligned} \tag{A.3}$$

Here we also use the relation $J = \frac{V_s}{V_m}$ where $V_s = \det(\mathbf{D}_s)$ is the deformed volume of the tetrahedron. Defining a new matrix $\mathbf{G} \in$

$\mathbb{R}^{4\times3}$, we can write Eq. A.3 in a more compact form:

$$\begin{aligned} \boldsymbol{f} &= \begin{pmatrix} \boldsymbol{f}_1 & \boldsymbol{f}_2 & \boldsymbol{f}_3 & \boldsymbol{f}_4 \end{pmatrix} = -V_s \boldsymbol{\sigma} \mathbf{G}^\top \in \mathbb{R}^{3\times4}, \\ \mathbf{G} &= \begin{pmatrix} \mathbf{D}_s^{-1} \\ -(1,1,1)^\top \mathbf{D}_s^{-1} \end{pmatrix} \in \mathbb{R}^{4\times3}. \end{aligned} \tag{A.4}$$

With the Kronecker product $\otimes$, we can compute the vectorized $\mathrm{vec}(\boldsymbol{f}) \in \mathbb{R}^{12}$ as:

$$\mathrm{vec}(\boldsymbol{f}) = -\mathrm{vec}(V_s \boldsymbol{\sigma} \mathbf{G}^\top) = -V_s (\mathbf{G} \otimes \mathbf{I}_3) \cdot \mathrm{vec}(\boldsymbol{\sigma}), \tag{A.5}$$

in which $\mathbf{I}_3 \in \mathbb{R}^{3\times3}$ is the identity matrix. From Eq. A.5, we can see $\mathrm{vec}(\boldsymbol{f})$ and $\mathrm{vec}(\boldsymbol{\sigma})$ have a linear relation, and the coefficient $-V_s(\mathbf{G} \otimes \mathbf{I}_3) \in \mathbb{R}^{12\times9}$ depends only on the deformed shape. Assembling $-V_s(\mathbf{G} \otimes \mathbf{I}_3)$ for all tetrahedrons, we can get Eq. 4.

According to the Cauchy's fundamental lemma, the Cauchy stress tensor must be a symmetric tensor, which gives:

$$\boldsymbol{\sigma} = \begin{pmatrix} s_0 & s_3 & s_4 \\ s_3 & s_1 & s_5 \\ s_4 & s_4 & s_2 \end{pmatrix}, \ \mathrm{vec}(\boldsymbol{\sigma}) = \mathbf{T}\boldsymbol{s}. \tag{A.6}$$

Here $\boldsymbol{s} \in \mathbb{R}^6$ is the 6 independent DoF, $\mathbf{T} \in \mathbb{R}^{9\times6}$ is the selection matrix. Combine Eq. A.5 and Eq. A.6, we ends up with the relation of $\boldsymbol{f}$ to $\boldsymbol{s}$, and Eq. 5 becomes the optimization for $\boldsymbol{s}$ accordingly.

References

[BEH18] BRANDT C., EISEMANN E., HILDEBRANDT K.: Hyper-reduced projective dynamics. *ACM Trans. Graph. 37*, 4 (July 2018). URL: https://doi.org/10.1145/3197517.3201387, doi:10.1145/3197517.3201387. 4

[BML*14] BOUAZIZ S., MARTIN S., LIU T., KAVAN L., PAULY M.: Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph. 33*, 4 (July 2014). URL: https://doi.org/10.1145/2601097.2601116, doi:10.1145/2601097.2601116. 2, 4, 7, 8

[BN21] BROWN G. E., NARAIN R.: Wrapd: weighted rotation-aware admm for parameterization and deformation. *ACM Trans. Graph. 40*, 4 (July 2021). URL: https://doi.org/10.1145/3450626.3459942, doi:10.1145/3450626.3459942. 2, 4

[CZXZ14] CHEN X., ZHENG C., XU W., ZHOU K.: An asymptotic numerical method for inverse elastic shape design. *ACM Trans. Graph. 33*, 4 (July 2014). URL: https://doi.org/10.1145/2601097.2601189, doi:10.1145/2601097.2601189. 2, 3, 7

[DJBDDT13] DEROUET-JOURDAN A., BERTAILS-DESCOUBES F., DAVIET G., THOLLOT J.: Inverse dynamic hair modeling with frictional contact. *ACM Trans. Graph. 32*, 6 (Nov. 2013). URL: https://doi.org/10.1145/2508363.2508398, doi:10.1145/2508363.2508398. 3

[DM98] DAGUM L., MENON R.: Openmp: an industry standard api for shared-memory programming. *IEEE Computational Science and Engineering 5*, 1 (1998), 46–55. doi:10.1109/99.660313. 6

[GJ*10] GUENNEBAUD G., JACOB B., ET AL.: Eigen v3. http://eigen.tuxfamily.org, 2010. 6

[GWM*24] GUO M., WANG B., MA P., ZHANG T., OWENS C. E., GAN C., TENENBAUM J. B., HE K., MATUSIK W.: Physically compatible 3d object modeling from a single image. *Advances in Neural Information Processing Systems 37* (2024). 1, 3

[Had06] HADAP S.: Oriented strands: dynamics of stiff multi-body system. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2006), SCA '06, Eurographics Association, p. 91–100. 3

[HLA*19] HU Y., LI T.-M., ANDERSON L., RAGAN-KELLEY J., DURAND F.: Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Trans. Graph. 38*, 6 (Nov. 2019). URL: https://doi.org/10.1145/3355089.3356506, doi:10.1145/3355089.3356506. 6

[HTYW22] HSU J., TRUONG N., YUKSEL C., WU K.: A general two-stage initialization for sag-free deformable simulations. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2022) 41*, 4 (07 2022), 64:1–64:13. URL: https://doi.org/10.1145/3528223.3530165, doi:10.1145/3528223.3530165. 2, 3, 4, 8, 10

[HWP*23] HSU J., WANG T., PAN Z., GAO X., YUKSEL C., WU K.: Sag-free initialization for strand-based hybrid hair simulation. *ACM Trans. Graph. 42*, 4 (July 2023). URL: https://doi.org/10.1145/3592143, doi:10.1145/3592143. 3, 5

[Int24] INTEL: oneapi math kernel library(onemkl). https://www.intel.com/content/www/us/en/developer/tools/oneapi/onemkl.html, 2024. 6

[J*24] JAKOB W., ET AL.: pybind11. https://pybind11.readthedocs.io/en/stable/#, 2024. 6

[Jia21] JIA K.: Sanm: a symbolic asymptotic numerical solver with applications in mesh deformation. *ACM Trans. Graph. 40*, 4 (July 2021). URL: https://doi.org/10.1145/3450626.3459755, doi:10.1145/3450626.3459755. 1, 2, 3, 6, 7, 8, 9, 10

[LBK17] LIU T., BOUAZIZ S., KAVAN L.: Quasi-newton methods for real-time simulation of hyperelastic materials. *ACM Transactions on Graphics (TOG) 36*, 3 (2017), 23. 2, 4, 7, 8

[LBOK13] LIU T., BARGTEIL A. W., O'BRIEN J. F., KAVAN L.: Fast simulation of mass-spring systems. *ACM Trans. Graph. 32*, 6 (Nov. 2013). URL: https://doi.org/10.1145/2508363.2508406, doi:10.1145/2508363.2508406. 2, 4

[LCBD*18] LY M., CASATI R., BERTAILS-DESCOUBES F., SKOURAS M., BOISSIEUX L.: Inverse elastic shell design with contact and friction. *ACM Trans. Graph. 37*, 6 (Dec. 2018). URL: https://doi.org/10.1145/3272127.3275036, doi:10.1145/3272127.3275036. 3

[LGL*19] LI M., GAO M., LANGLOIS T., JIANG C., KAUFMAN D. M.: Decomposed optimization time integrator for large-step elastodynamics. *ACM Trans. Graph. 38*, 4 (July 2019). URL: https://doi.org/10.1145/3306346.3322951, doi:10.1145/3306346.3322951. 4

[LZL*25] LAI Z., ZHAO Y., LIU H., ZHAO Z., LIN Q., SHI H., YANG X., YANG M., YANG S., FENG Y., ZHANG S., HUANG X., LUO D., YANG F., YANG F., WANG L., LIU S., TANG Y., CAI Y., HE Z., LIU T., LIU Y., JIANG J., LINUS, HUANG J., GUO C.: Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details, 2025. URL: https://arxiv.org/abs/2506.16504, arXiv:2506.16504. 1, 2, 9

[MCD*23] MA P., CHEN P. Y., DENG B., TENENBAUM J. B., DU T., GAN C., MATUSIK W.: Learning neural constitutive laws from motion observations for generalizable pde dynamics. In *International Conference on Machine Learning* (2023), PMLR. 5

[MSHG84] MORÉ J. J., SORENSEN D. C., HILLSTROM K. E., GARBOW B. S.: The minpack project. In *Sources and Development of Mathematical Software*, Cowell W. J., (Ed.). Prentice-Hall, 1984, pp. 88–111. 5

[MWW18] MUKHERJEE R., WU L., WANG H.: Interactive two-way shape design of elastic bodies. *Proc. ACM Comput. Graph. Interact. Tech. 1*, 1 (July 2018). URL: https://doi.org/10.1145/3203196, doi:10.1145/3203196. 3

[NOB16] NARAIN R., OVERBY M., BROWN G. E.: Admm $\supseteq$ projective dynamics: fast simulation of general constitutive models. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2016), SCA '16, Eurographics Association, p. 21–28. 2, 4

[Ope25] OpenAI: Gpt-4o, 2025. Accessed: 2025-08-31. URL: https://openai.com/gpt-4o. 9

[PWLSH13] Prévost R., Whiting E., Lefebvre S., Sorkine-Hornung O.: Make it stand: balancing shapes for 3d fabrication. *ACM Trans. Graph. 32*, 4 (July 2013). URL: https://doi.org/10.1145/2461912.2461957, doi:10.1145/2461912.2461957. 3

[RWLC24] Ruan L., Wang B., Liu T., Chen B.: Minnie: a mixed multigrid method for real-time simulation of nonlinear near-incompressible elastics. *ACM Trans. Graph. 43*, 6 (dec 2024). URL: https://doi.org/10.1145/3687758, doi:10.1145/3687758. 4

[SB12] Sifakis E., Barbic J.: Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses* (New York, NY, USA, 2012), SIGGRAPH '12, Association for Computing Machinery. URL: https://doi.org/10.1145/2343483.2343501, doi:10.1145/2343483.2343501. 2, 4, 5, 7, 9, 10

[SGK18] Smith B., Goes F. D., Kim T.: Stable neo-hookean flesh simulation. *ACM Trans. Graph. 37*, 2 (Mar. 2018). URL: https://doi.org/10.1145/3180491, doi:10.1145/3180491. 2, 5, 7, 9

[Sid23] Side Effects Software Inc.: Houdini, 2023. Version 19.5. URL: https://www.sidefx.com. 9

[STBG12] Skouras M., Thomaszewski B., Bickel B., Gross M.: Computational design of rubber balloons. *Computer Graphics Forum 31*, 2pt4 (2012), 835–844. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2012.03064.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2012.03064.x, doi:https://doi.org/10.1111/j.1467-8659.2012.03064.x. 2, 3

[STK*14] Skouras M., Thomaszewski B., Kaufmann P., Garg A., Bickel B., Grinspun E., Gross M.: Designing inflatable structures. *ACM Trans. Graph. 33*, 4 (July 2014). URL: https://doi.org/10.1145/2601097.2601166, doi:10.1145/2601097.2601166. 3

[TB25] Takahashi T., Batty C.: Rest shape optimization for sag-free discrete elastic rods. *Computer Graphics Forum n/a*, n/a (2025), e70019. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.70019, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.70019, doi:https://doi.org/10.1111/cgf.70019. 3

[TKA11] Twigg C. D., Kačić-Alesić Z.: Optimization for sag-free simulations. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2011), SCA '11, Association for Computing Machinery, p. 225–236. URL: https://doi.org/10.1145/2019406.2019437, doi:10.1145/2019406.2019437. 2, 3

[TKL22] Trusty T., Kaufman D., Levin D. I.: Mixed variational finite elements for implicit simulation of deformables. In *SIGGRAPH Asia 2022 Conference Papers* (New York, NY, USA, 2022), SA '22, Association for Computing Machinery. URL: https://doi.org/10.1145/3550469.3555418, doi:10.1145/3550469.3555418. 5, 6

[TSBU24] Tojo K., Shamir A., Bickel B., Umetani N.: Fabricable 3d wire art. In *ACM SIGGRAPH 2024 Conference Papers* (New York, NY, USA, 2024), SIGGRAPH '24, Association for Computing Machinery. URL: https://doi.org/10.1145/3641519.3657453, doi:10.1145/3641519.3657453. 3

[WWY*15] Wang B., Wu L., Yin K., Ascher U., Liu L., Huang H.: Deformation capture and modeling of soft objects. *ACM Trans. Graph. 34*, 4 (July 2015). URL: https://doi.org/10.1145/2766911, doi:10.1145/2766911. 2, 3, 10

[XLX*24] Xiang J., Lv Z., Xu S., Deng Y., Wang R., Zhang B., Chen D., Tong X., Yang J.: Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506* (2024). 9

[XSZB15] Xu H., Sin F., Zhu Y., Barbič J.: Nonlinear material design using principal stretches. *ACM Trans. on Graphics (SIGGRAPH 2015) 34*, 4 (2015). 1, 2, 5, 7, 9

[ZCT21] Zehnder J., Coros S., Thomaszewski B.: Sgn: Sparse gauss-newton for accelerated sensitivity analysis. *ACM Trans. Graph. 41*, 1 (Sept. 2021). URL: https://doi.org/10.1145/3470005, doi:10.1145/3470005. 2, 3

[ZYW*24] Zhang T., Yu H.-X., Wu R., Feng B. Y., Zheng C., Snavely N., Wu J., Freeman W. T.: PhysDreamer: Physics-based interaction with 3d objects via video generation. In *European Conference on Computer Vision* (2024), Springer. 1