Stable Cosserat Rods

JERRY HSU, University of Utah, USA
TONGTONG WANG, LIGHTSPEED, China
KUI WU, LIGHTSPEED, USA
CEM YUKSEL, University of Utah, USA

Fig. 1. *Our method can stably handle letters knitted with Cosserat rods with close to 2 million degrees of freedom at an interactive frame rate of 4 fps on an NVIDIA RTX 3090. This results highlights that our method is highly robust, efficient, and easily parallelizable.*

Cosserat rods have become an increasingly popular framework for simulating complex bending and twisting in thin elastic rods, used for hair, tree, and yarn-level cloth models. However, traditional approaches often encounter significant challenges in robustly and efficiently solving for valid quaternion orientations, even when employing small time steps or computationally expensive global solvers. We introduce *stable Cosserat rods*, a new solver that can achieve high accuracy with high stiffness levels and maintain stability under large time steps. It is also inherently suitable for parallelization. Our key contribution is a split position and rotation optimization scheme with a closed-form Gauss-Seidel quasi-static orientation update. This solver significantly improves the numerical stability with Cosserat rods, allowing faster computation and larger time steps. We validate our method across a wide range of applications, including simulations of hair, trees, yarn-level cloth, slingshots, and bridges, demonstrating its ability to handle diverse material behaviors and complex geometries. Furthermore, we show that our method is orders of magnitude faster and more stable than alternative rod solvers, such as extended position-based dynamics and discrete elastic rods.

CCS Concepts: • **Computing methodologies** → **Physical simulation**.

Additional Key Words and Phrases: Cosserat Rods, Thin Elastic Rods

ACM Reference Format:
Jerry Hsu, Tongtong Wang, Kui Wu, and Cem Yuksel. 2025. Stable Cosserat Rods. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25), August 10–14, 2025, Vancouver, BC, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3721238.3730618

Authors' addresses: Jerry Hsu, jerry.hsu.research@gmail.com, University of Utah, Salt Lake City, UT, USA; Tongtong Wang, wangtong923@gmail.com, LIGHTSPEED, Shenzhen, China; Kui Wu, walker.kui.wu@gmail.com, LIGHTSPEED, Los Angeles, CA, USA; Cem Yuksel, cem@cemyuksel.com, University of Utah, Salt Lake City, UT, USA.

SIGGRAPH Conference Papers '25, August 10–14, 2025, Vancouver, BC, Canada

ACM ISBN 979-8-4007-1540-2/2025/08
https://doi.org/10.1145/3721238.3730618

1 INTRODUCTION

One-dimensional curves play an essential role in modern digital worlds. On closer inspection, one can find curves almost everywhere from the blades of grass under a character's foot, to the threads of their clothing, and to each strand of their hair. From the rope they use to the bridge they stand on and the bows and swords they wield, curves find their way into a wide variety of everyday objects.

However, the complex behaviors of *thin elastic rods* (i.e., stretching, shearing, bending, and twisting) introduce moving material frames and nonlinear constraints that make rod simulation both mathematically complex and computationally expensive. As such, many schemes have been proposed to tackle this problem, such as spring models [Iben et al. 2013; Selle et al. 2008], Cosserat rods [Kugelstadt and Schömer 2016], discrete elastic rod (DER) [Bergou et al. 2008], and super-helices [Bertails et al. 2006]. Unfortunately, these methods often suffer from issues of instability, forcing the use of either extremely small time steps or expensive global solvers.

In this paper, we introduce *stable Cosserat rods*, a novel simulation scheme designed to efficiently and robustly handle complex scenarios involving rods, even under large time steps. Similar to the quasi-static Newton material frame update found in DER [Bergou et al. 2008], our key insight is that the angular momentum's contribution to the thin Cosserat rod dynamics is negligible. Therefore, we can split the original optimization problem into two interleaved alternating optimizations for positions and orientations. To solve for orientations, we introduce a closed-form Gauss-Seidel local relaxation approach that employs an adjoint variable derived directly from the unit quaternion constraint. This decouples position from orientation and allows for a much simplified alternating position solve using methods like Vertex Block Descent. When combined with our orientation solve, this ensures both computational efficiency and stability.

We validate our method against traditional Cosserat rod solvers, such as Extended Position-Based Dynamics (XPBD), Vertex Block Descent (VBD), and alternative models like Discrete Elastic Rods

(DER). Even when we augment VBD with stability extensions, our method outperforms these methods by more than an order of magnitude in terms of computational efficiency (by at least 18× in our tests). Even in a worst-case scenario, our method surpasses global solvers, i.e. DER (by a factor of 46× on CPU in our test). Exploiting the embarrassingly parallel nature of our method, we further showcase the speed and robustness of our approach in large-scale GPU simulations of hair and yarn-level cloth.

Notably, we achieve this substantial improvement in stability and performance without compromising accuracy. Moreover, our method is straightforward to implement, requiring only a few lines of code beyond that for mass-spring systems, as fully detailed in our implementation.

2 BACKGROUND

Rod simulation is challenging as it requires the tracking of material frames (orientations) along the curve for twisting and torsional effects. As such, a significant body of research has been dedicated to simulating thin elastic rods. In this section, we briefly summarize the prior work and then present the details of the Cosserat rods method that we build upon.

2.1 Prior Work

Pai [2002] first introduced the Cosserat model to the computer graphics community, employing an implicit representation for rods. This work was extended to the super-helix model [Bertails 2009; Bertails et al. 2006], which represents rods as piecewise helices governed by Lagrangian mechanics. Over time, the Cosserat model has seen several advancements, including using a deformation model for dynamic elastic rods with continuous energies [Spillmann and Teschner 2007], attaching ghost points to edges for torsion tracking [Umetani et al. 2015], and incorporating additional energy terms to control the spatial distribution of material points [Wen et al. 2020]. To improve the efficiency of solving the Cosserat model, Casati and Bertails-Descoubes [2013] proposed a semi-implicit time-stepping scheme based on power expansion. Kugelstadt and Schömer [2016] use extended Extended Position-Based Dynamics (XPBD) [Macklin et al. 2016] to support Cosserat rods, which was later refined for stiff rods [Deul et al. 2018]. Soler et al. [2018] utilized Projective Dynamics (PD) [Bouaziz et al. 2014] with a pre-factored matrix to accelerate global linear system solving. More recently, Zhao et al. [2022] adopted compact representations for Cosserat rods to enhance convergence efficiency.

Also reliant on a global solver, Discrete Elastic Rods (DER) [Bergou et al. 2010, 2008] utilize discrete differential geometry to track segment material frames for bending and twisting explicitly. Iben et al. [2013] extended the idea of DER to incorporate bending springs to operate on smoothed curves, enhancing stability. Similarly, Selle et al. [2008] approximated bending using springs constructed on virtual tetrahedra. More recently, Huang et al. [2023] and Daviet [2023] accelerated DER-based hair simulation on GPUs, employing an efficient semi-implicit DER system and an alternating direction method of multipliers (ADMM), respectively.

Recent advancements in rod simulation have led to the widespread use of Cosserat rods in various applications, including vegetation [Deul et al. 2018; Pirk et al. 2017], hair [Daviet 2023; Han et al. 2019; Hsu et al. 2023, 2024], yarn [Kaldor et al. 2008, 2010; Leaf et al. 2018], and muscle [Angles et al. 2019] simulations. These developments underscore their significance and establish Cosserat rods as the central focus of this paper.

However, regardless of the choice of Cosserat or DER models, it remains mathematically challenging and computationally expensive to simulate the nonlinear constraints governing the complex behaviors of rods. To address the optimization over positions and orientations, global implicit methods often rely on costly solvers, while local iterative methods can suffer from stability issues. In this paper, we present a stable Cosserat rods scheme to efficiently and robustly handle complex scenarios, even under large time steps.

2.2 Cosserat rods

Traditionally, Cosserat rods are modeled as a continuous curve in 3D space, parameterized by s with position $\mathbf{x}(s) : \mathbb{R} \to \mathbb{R}^3$ and unit quaternion orientation $\mathbf{q}(s) : \mathbb{R} \to Q_8$ (i.e., the material frame) [Kugelstadt and Schömer 2016]. This redundancy between position and orientation allows Cosserat rods to handle complex stretching, shearing, bending, and twisting behaviors. In particular, the stretching and shearing strain, $\Gamma(s)$, is defined as

$$\Gamma(s) = \partial_s \mathbf{x}(s) - \mathbf{d}_3(s),$$

where $\mathbf{d}_3(s)$ denotes the normal of the cross-section and ∂_s is the derivative along s. Given a space-fixed world coordinate system with basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$, $\mathbf{d}_\alpha$ can be obtained simply by rotating the α-th standard basis in $\mathbb{R}^3$, $\mathbf{e}_\alpha$, with rotation quaternion $\mathbf{q}(s)$ as $\mathbf{d}_\alpha = \mathbf{q}\mathbf{e}_\alpha\overline{\mathbf{q}}$. By convention, we choose $\alpha = 3$ as our centerline axis. Similarly, a bending and twisting strain, $\Omega(s)$, can be defined as

$$\Omega(s) = \mathfrak{I}\left[2\overline{\mathbf{q}(s)}\partial_s \mathbf{q}(s)\right].$$

Here, $\mathfrak{I}$ retains only the imaginery components and the overline $\overline{\mathbf{q}}$ denotes quaternion conjugation. This Ω is then often referred to as the *Darboux vector* to measure the bending and twisting curvature. With everything combined, the overall material potential energy E^{total} over s can be defined as

$$E^{\text{total}} = \int \tilde{k}^{\text{ss}} \frac{|\Gamma(s)|^2}{2} |\partial_s \mathbf{x}(s)|\, ds + \int \tilde{k}^{\text{bt}} \frac{|\Omega(s)|^2}{2} |\partial_s \mathbf{x}(s)|\, ds,$$

where $\tilde{k}^{\text{ss}}$ and $\tilde{k}^{\text{bt}}$ are the stiffness parameters for stretching/shearing and bending/twisting, respectively.

Cosserat rods are typically discretized as polyline segments with positions on vertices and orientations on segments [Kugelstadt and Schömer 2016]. As shown in Fig. 2, the i-th vertex stores the position $\mathbf{x}_i$. The orientation $\mathbf{q}_i$ is stored in the segment following vertex i, denoted as the i-th segment. $\Gamma(s)$ is further discretized into stretching and shearing constraint

$$C_i^{\text{ss}} = \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{l_i} - \mathbf{d}_{i,3} \tag{1}$$

by using finite differences with segment rest length l_i to approximate $\partial_s \mathbf{x}(s)$. Similarly, $\Omega(s)$ between the segments i and $i+1$ can be

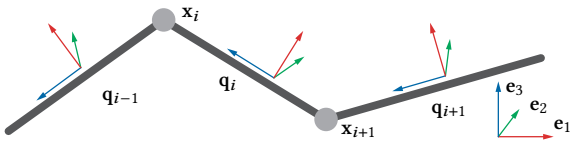

Fig. 2. *Discretized Cosserat rod with position* $\mathbf{x}$ *on vertices and orientation* $\mathbf{q}$ *on segments.* $\mathbf{e}_\alpha$ *is the axis-aligned orthonormal basis.*

discretized as

$$\Omega_i = \Im\left[2\left(\frac{\overline{\mathbf{q}}_{i+1}+\overline{\mathbf{q}}_i}{2}\right)\left(\frac{\mathbf{q}_{i+1}-\mathbf{q}_i}{l_i}\right)\right] = \frac{2}{l_i}\Im\left[\overline{\mathbf{q}}_i\mathbf{q}_{i+1}\right]. \tag{2}$$

which can be further rewritten as a constraint form

$$C_i^{\text{bt}} = \overline{\mathbf{q}}_i\mathbf{q}_{i+1} - \phi_i\mathbf{q}_i^0, \tag{3}$$

$$\phi_i = \begin{cases} +1 & \text{for } \|\overline{\mathbf{q}}_i\mathbf{q}_{i+1} - \mathbf{q}_i^0\|^2 \le \|\overline{\mathbf{q}}_i\mathbf{q}_{i+1} + \mathbf{q}_i^0\|^2 \\ -1 & \text{for } \|\overline{\mathbf{q}}_i\mathbf{q}_{i+1} - \mathbf{q}_i^0\|^2 > \|\overline{\mathbf{q}}_i\mathbf{q}_{i+1} + \mathbf{q}_i^0\|^2 \end{cases}. \tag{4}$$

Semantically, $\overline{\mathbf{q}}_i\mathbf{q}_{i+1}$ is the angle difference between neighboring segments. As such, C_i^{bt} penalizes bending and twisting away from some rest angle $\mathbf{q}_i^0$, e.g., $\overline{\mathbf{q}}_i\mathbf{q}_{i+1}$ when at rest. Furthermore, since the quaternions $\mathbf{q}$ and $-\mathbf{q}$ denote the same angle, ϕ_i, is used to drive the segments towards the closest quaternionic pole. Additionally, Hsu et al. [2023] also suggests including the real component in C^{bt} for better stability.

Finally, the integral over s for the total potential energy E^{total} can be broken down as the sum of the energies E_i^{ss} and E_i^{bt} over all segments i.

$$E^{\text{total}} = \sum_i E_i^{\text{ss}} + \sum_i E_i^{\text{bt}}, \tag{5}$$

where $E_i^{\text{ss}} = \frac{k_i^{\text{ss}}}{2}|C_i^{\text{ss}}|^2$ and $E_i^{\text{bt}} = \frac{k_i^{\text{bt}}}{2}|C_i^{\text{bt}}|^2$, with $k_i^{ss} = \tilde{k}^{ss}l_i$ and $k_i^{bt} = 4\tilde{k}^{bt}/l_i$ for brevity.

The time integration of Cosserat Rods using implicit Euler can be written as an optimization on the variational energy

$$\mathbf{x},\mathbf{q} = \arg\min_{\mathbf{x},\mathbf{q}} \frac{1}{2h^2}||\mathbf{x}-\mathbf{y}||_{\mathbf{M}}^2 + \frac{1}{2h^2}||\mathbf{q}-\mathbf{w}||_{\mathbf{J}}^2 + E^{\text{total}} \quad , \quad \text{s.t. } |\mathbf{q}_i| = 1 \;\forall\; i.$$

Here, the first two terms are the positional and rotational inertial potentials. h is the time step. $\mathbf{M}$ and $\mathbf{J}$ are the lumped generalized mass matrices for position and orientation. Finally, $\mathbf{y} = \mathbf{x} + h\mathbf{v} + h^2\mathbf{a}^{\text{ext}}$ and $\mathbf{w} = \mathbf{q} + h\omega\mathbf{q}/2$ are the inertial positions and orientations [Bouaziz et al. 2014; Soler et al. 2018]. These are computed from the velocities, $\mathbf{v}$ and ω, and acceleration $\mathbf{a}^{\text{ext}}$. Typically, this minimization is done using XPBD. However, due to the complex energy coupling and unit quaternion constraint, Cosserat Rods simulation often requires either extremely small time steps and large iteration counts with XPBD [Hsu et al. 2023; Kugelstadt and Schömer 2016] or slow global solvers like projective dynamics [Soler et al. 2018].

3 STABLE COSSERAT RODS

In this section, we present the theory behind our stable Cosserat rods formulation. Implementation-oriented readers may skip this section and continue to Sec. 4.

Our stable Cosserat rods formulation offers a numerically robust alternative for time integration, which allows larger time steps and requires fewer iterations with substantially improved performance. Similar to the quasi-static Newton material frame update found in DER [Bergou et al. 2008], our key insight is that the angular momentum has a negligible impact on the dynamics of thin Cosserat rods ($\mathbf{J} = \mathbf{0}$). This allows us to simplify the problem by treating the rod rotational inertia as that of *infinitely thin rods* in a quasi-static state. Building on this assumption, we decompose the optimization scheme into separate positional and rotational steps, leveraging a novel quasi-static orientation solver.

The core of our approach then lies in our efficient orientation solver, which employs a Gauss-Seidel-style local relaxation scheme based on closed-form local solutions. Rather than relying on secondary constraint enforcement mechanisms like XPBD, we resolve constraints directly by introducing a new auxiliary variable, λ, from which orientations are derived. We then split the original optimization into two with interleaved alternating minimizations between positions and orientations. The position update minimizes the implicit Euler variational energy given fixed orientations as

$$\mathbf{x} = \arg\min_{\mathbf{x}} \frac{1}{2h^2}||\mathbf{x}-\mathbf{y}||_{\mathbf{M}}^2 + E^{\text{total}}. \tag{6}$$

Similarly, the orientation update solves for the quasi-static state given fixed positions as

$$\mathbf{q} = \arg\min_{\mathbf{q}} E^{\text{total}} \quad \text{s.t. } |\mathbf{q}_i| = 1 \;\forall\; i. \tag{7}$$

As demonstrated in Fig. 3 and Fig. 4, this method effectively replicates the complex behaviors and knotting observed in thin elastic rods.

We begin by decomposing the quasi-static orientation solve into successive local optimizations leveraging the new auxiliary

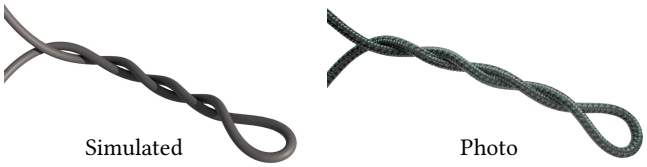

Fig. 3. *Our method is able to reproduce many of the effects of bending and twisting with self contacts. For example, our results closely match the twirls found in the real life cable on the right.*

Fig. 4. *Our method forms natural knots under tension.*

variable λ (Sec. 3.1). Then, we derive approximate solutions for λ (Sec. 3.2) and prove that the approximate solution is also well-behaved (Sec. 3.3). Finally, the exact solution can then be optionally computed using a novel fixed-point iteration for traditional Cosserat rods (Sec. 3.4).

3.1 Quasi-static Orientation Solve

To begin, quasi-static orientation equilibrium implies finding $\mathbf{q}$ to satisfy Eq. 7. Since the variable $\mathbf{q}$ represents unit quaternions, it is essential to enforce the unit norm constraint, such that $|\mathbf{q}_i| = 1$. Directly addressing this would involve solving a large global constrained optimization problem, which can be expensive. To address this, we adopt a local relaxation approach, iterating over each $\mathbf{q}_i$ while treating all other degrees of freedom as fixed. This results in the local optimization problem

$$\mathbf{q}_i = \arg\min_{\mathbf{q}_i} E_i^{\text{ss}} + E_{i-1}^{\text{bt}} + E_i^{\text{bt}} \quad \text{s.t. } |\mathbf{q}_i| = 1, \tag{8}$$

whose optimality condition implies that the net torque $\boldsymbol{\tau}^{\text{net}}$ acting on $\mathbf{q}_i$ must be 0. Adding a Lagrange multiplier λ to our equilibrium condition for the constraint $|\mathbf{q}_i| = 1$, Eq. 8 can be converted to

$$\boldsymbol{\tau}_i^{\text{net}} - \lambda \mathbf{q}_i = 0 \qquad \text{with} \qquad \boldsymbol{\tau}_i^{\text{net}} = \boldsymbol{\tau}_i^{\text{ss}} + \boldsymbol{\tau}_{i-1,i}^{\text{bt}} + \boldsymbol{\tau}_{i,i}^{\text{bt}} \tag{9}$$

where $\boldsymbol{\tau}_{i,i}^{\text{ss}}$ and $\boldsymbol{\tau}_{i,j}^{\text{bt}}$ denote the torques generated by E_i^{ss} on $\mathbf{q}_i$ and E_i^{bt} on $\mathbf{q}_j$, respectively. Since we assume all other segments are fixed, we only have to compute the torque generated from E_i^{ss}, E_{i-1}^{bt}, and E_i^{bt} on $\mathbf{q}_i$, which are obtained as the negative potential gradients

$$\boldsymbol{\tau}_{i,i}^{\text{ss}} = -\nabla_{\mathbf{q}_i} E_i^{ss} = -2k_i^{\text{ss}} \left(\frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{l_i} - \mathbf{d}_{i,3} \right) \mathbf{q}_i \mathbf{e}_3, \tag{10}$$

$$\boldsymbol{\tau}_{i,i}^{\text{bt}} = -\nabla_{\mathbf{q}_i} E_i^{\text{bt}} = -k_i^{\text{bt}} \left(\mathbf{q}_i - \phi_i \mathbf{q}_{i+1} \bar{\mathbf{q}}_i^0 \right), \tag{11}$$

$$\boldsymbol{\tau}_{i-1,i}^{\text{bt}} = -\nabla_{\mathbf{q}_i} E_{i-1}^{\text{bt}} = -k_{i-1}^{\text{bt}} \left(\mathbf{q}_i - \phi_{i-1} \mathbf{q}_{i-1} \mathbf{q}_{i-1}^0 \right). \tag{12}$$

These can then be substituted into the static-equilibrium condition (Eq. 9) and simplified to obtain

$$\begin{aligned} -2k_i^{\text{ss}} l_i^{-1} (\mathbf{x}_{i+1} - \mathbf{x}_i) \mathbf{q}_i \mathbf{e}_3 + k_{i-1}^{\text{bt}} \phi_{i-1} \mathbf{q}_{i-1} \mathbf{q}_{i-1}^0 \\ + k_i^{\text{bt}} \phi_i \mathbf{q}_{i+1} \bar{\mathbf{q}}_i^0 - \lambda \mathbf{q}_i = 0. \end{aligned} \tag{13}$$

As pointed out by Hsu et al. [2024], any torque acting along $\mathbf{q}_i$ will be canceled out by the unit quaternion constraint. We can rewrite Eq. 13 into a simplified form as

$$\mathbf{v} \mathbf{q}_i \mathbf{e}_3 + \mathbf{b} - \lambda \mathbf{q}_i = 0, \tag{14}$$

in which we group the effects of stretching into $\mathbf{v}$ and the effects of bending into $\mathbf{b}$ as

$$\mathbf{v} = -2k_i^{\text{ss}} l_i^{-1} (\mathbf{x}_{i+1} - \mathbf{x}_i), \tag{15}$$

$$\mathbf{b} = k_{i-1}^{\text{bt}} \phi_{i-1} \mathbf{q}_{i-1} \mathbf{q}_{i-1}^0 + k_i^{\text{bt}} \phi_i \mathbf{q}_{i+1} \bar{\mathbf{q}}_i^0. \tag{16}$$

This grouping conveniently allows for the consideration of an arbitrary number of edge connections such that $\mathbf{b}$ becomes the sum of all bending constraint contributions.

$$\mathbf{b}_i = \sum_{C_i^{\text{bt}}} \begin{cases} k_{ji}^{\text{bt}} \phi_{ji} \mathbf{q}_j \mathbf{q}_{ji}^0 & \text{for } C_{ji}^{\text{bt}} \\ k_{ij}^{\text{bt}} \phi_{ij} \mathbf{q}_j \bar{\mathbf{q}}_{ij}^0 & \text{for } C_{ij}^{\text{bt}} \end{cases} \tag{17}$$

Note the difference in the vertex ordering ij and ji. This means our method naturally supports the simulation of graphs. Returning to the solution with some substitutions from Eq. 14, we arrive at

$$\mathbf{q}_i(\lambda) = \frac{\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda \mathbf{b}}{\lambda^2 - |\mathbf{v}|^2}. \tag{18}$$

This leaves λ as the only unknown. Thus, efficiently solving for λ, as we describe below, results in an efficient and stable method for Cosserat rods.

3.2 Approximate Solution to λ

Recall that the unit quaternion constraint requires $|\mathbf{q}_i| = 1$. As the resulting polynomial is quartic, it can be difficult to solve for λ efficiently

$$\left| \frac{\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda \mathbf{b}}{\lambda^2 - |\mathbf{v}|^2} \right| = 1 \tag{19}$$

However, an exact solution to λ may not always be necessary. In many cases, an approximate solution, combined with a simple normalization step such as $\mathbf{q}_i = \mathbf{q}_i(\lambda)/|\mathbf{q}_i(\lambda)|$, can be sufficient, particularly when strict adherence to the prescribed Cosserat rod material model is not required. At the minimum, an approximate solution can also serve as an effective initial guess for any iterative solvers that refine λ.

To find an approximate solution, we first observe that, at most, four λ satisfy the condition $|\mathbf{q}_i| = 1$. Among these, we conjecture that the most positive root corresponds to the most physically stable configuration for three reasons. First, in Eq. 17, $\mathbf{b}$ represents the configuration without stretching or shearing. By picking the largest λ, we allow $\mathbf{b}$ to dominate in Eq. 14. Second, since a local minimum must exist, the fact that the two roots closest to 0 can be imaginary suggests that they are edge cases. Third, when $\lambda > |\mathbf{v}|$, the inner product between $\mathbf{b}$ and $\mathbf{q}(\lambda)$ will always be positive, minimizing the chances of ϕ flipping. This leaves us with the largest root which we found to always minimize the energy in practice.

Applying the triangle inequality to Eq. 19, we can see that

$$1 = \left| \frac{\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda \mathbf{b}}{\lambda^2 - |\mathbf{v}|^2} \right| \leq \frac{|\mathbf{v}||\mathbf{b}| + \lambda |\mathbf{b}|}{\left| \lambda^2 - |\mathbf{v}|^2 \right|}. \tag{20}$$

Solving this quadratic inequality gives an upper bound of $|\mathbf{v}|+|\mathbf{b}|$ for the largest positive λ leading to $|\mathbf{q}_i(\lambda)| < 1$. For a lower bound on λ, we can see in Eq. 20 that $|\mathbf{q}_i(\lambda)|$ contains an asymptotic discontinuity at $\lambda = |\mathbf{v}|$. Since $|\mathbf{q}_i(\lambda)|$ goes to infinity at $\lambda = |\mathbf{v}|$ but is less than 1 at $\lambda = |\mathbf{v}| + |\mathbf{b}|$, the largest root of λ must satisfy that

$$|\mathbf{v}| < \lambda \leq |\mathbf{v}| + |\mathbf{b}|. \tag{21}$$

In our implementation, we use the upper bound of the largest positive value to approximate λ as

$$\lambda \approx |\mathbf{v}| + |\mathbf{b}| \tag{22}$$

since this coincides with the exact Cosserat rod solution under zero strain (i.e. $\lambda = |\mathbf{v}| + |\mathbf{b}|$ when $|C^{\text{ss}}| = |C^{\text{bt}}| = 0$). We found this to approximate the measured λ well in our experiments, which is important to guarantee convergence. Since each update minimizes the global energy with respect to the orientation being updated, the global energy always monotonically decrease for a sufficiently accu-

rate λ. With this in mind, we dub our scheme Stable Cosserat Rods for its stability and further show our well-definedness in Sec. 3.3.

3.3 Well-Definedness of $\mathbf{q}(\lambda)$

In this section, we show that our $\mathbf{q}(\lambda)$ and approximate solution $\lambda = |\mathbf{v}| + |\mathbf{b}|$ can be used as a new and full-fledged material model by itself.

First, our material is invariant under vertex ordering. This is easy to see in Eq. 1 and Eq. 18 as swapping vertices only flips the signs of $\mathbf{v}$ and $\mathbf{b}$.

Second, our material is invariant under translation and rotation (material frame indifference). Rotating $\mathbf{q}_i(\lambda)$ by any arbitrary quaternion, $\mathbf{R}$, also naturally rotates $\mathbf{v}$ and $\mathbf{b}$ by $\mathbf{R}$.

$$\mathbf{R}\mathbf{q}_i(\lambda) = \frac{\mathbf{R}\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda\mathbf{R}\mathbf{b}}{\lambda^2 - |\mathbf{v}|^2} = \frac{(\mathbf{R}\mathbf{v}\bar{\mathbf{R}})\mathbf{R}\mathbf{b}\mathbf{e}_3 + \lambda\mathbf{R}\mathbf{b}}{\lambda^2 - |\mathbf{v}|^2}, \tag{23}$$

since $|\mathbf{v}| = |\mathbf{R}\mathbf{v}\bar{\mathbf{R}}|$, $|\mathbf{b}| = |\mathbf{R}\mathbf{b}|$, and $\lambda = |\mathbf{v}| + |\mathbf{b}|$ itself is rotationally invariant. Hence, $\mathbf{q}_i(\lambda)$ remains rotationally invariant.

Third, our material introduces no artificial strain. This can be verified by looking at $\mathbf{q}_i$ under $|C^{\text{ss}}| = |C^{\text{bt}}| = 0$ or when there is 0 strain. In Eq. 15, $|C^{\text{ss}}| = 0$ implies that $\mathbf{v} = -|\mathbf{v}|\mathbf{q}_i\mathbf{e}_3\bar{\mathbf{q}}_i$. Similarly, in Eq. 16, $|C^{\text{bt}}| = 0$ implies that $\mathbf{b} = |\mathbf{b}|\mathbf{q}_i$. Plugging these into the solution for $\mathbf{q}_i$ reveals that

$$\mathbf{q}_i(\lambda) = \frac{-|\mathbf{v}|\mathbf{q}_i\mathbf{e}_3\bar{\mathbf{q}}_i|\mathbf{b}|\mathbf{q}_i\mathbf{e}_3+\lambda|\mathbf{b}|\mathbf{q}_i}{\lambda^2-|\mathbf{v}|^2} = \frac{|\mathbf{v}||\mathbf{b}|\mathbf{q}_i+\lambda|\mathbf{b}|\mathbf{q}_i}{\lambda^2-|\mathbf{v}|^2} = \frac{|\mathbf{b}|(|\mathbf{v}|+\lambda)}{\lambda^2-|\mathbf{v}|^2}\mathbf{q}_i,$$

which normalizes back into $\mathbf{q}_i$. This means that $\mathbf{q}_i$ is itself a valid equilibrium solution, preserving the rest shape. Finally, as an aside, this form also shows that Eq. 22 leads to $|\mathbf{q}_i| = 1$ under 0 strain.

As such, we arrive at the conclusion that our $\mathbf{q}(\lambda)$ is well defined even in the approximate case. This makes our exact local closed-form solution highly stable and usable as a full-fledged alternative to traditional Cosserat rods.

3.4 Exact Solution to λ

In applications where the traditional Cosserat energy must be strictly adhered to, our λ iterant can also be highly effective as the initial guess. As it is known that an asymptote exists near our desired solution, fixed-point iterations can be well suited for this refinement of λ without reintroducing instability.

Fixed-point iteration involves the construction of a "fixed point" computed as $\lambda_{fp} = f(\lambda)$. By repeatedly applying the fixed-point operator $f(\lambda)$, λ can rapidly converge to a point where it becomes fixed (i.e. a fixed point). Critically, as long as we construct $f(\lambda) = \lambda$ to be equivalent to $|\mathbf{q}(\lambda)| = 1$, this fixed point will coincide with the solution for λ. As is typical with fixed point iterations, this can be done by carefully substituting some terms of λ with λ_{fp}.

$$|\mathbf{q}(\lambda)| = \left|\frac{\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda\mathbf{b}}{\lambda_{fp}^2 - |\mathbf{v}|^2}\right| = 1 \tag{24}$$

Finally, we invert and solve for λ_{fp} to arrive at

$$\lambda_{fp} = f(\lambda) = \sqrt{|\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda\mathbf{b}| + |\mathbf{v}|^2} \tag{25}$$

By repeatedly applying $f(\lambda)$, λ quickly converges to a fixed-point which satisfies that $|\mathbf{q}_i(\lambda)| = 1$. This is true for infinitely many combinations of substitutions. However, we found Eq. 25 to be the most stable of the ones we tested due to the Banach fixed-point theorem. Excluding the substitution choices that lead to discontinuous fixed-point operators, Eq. 25 proves to have the smallest gradient, $|\nabla_\lambda f(\lambda)| < 1$, in our domain. This makes our specific fix-point operator converge the fastest out of all the ones we obtained.

Algorithm 1 demonstrates our iterative solution for λ. To refine our initial guess of λ until matching traditional Cosserat formulations, we store λ_i on each segment as an auxiliary degree of freedom. We then update λ_i by applying $f(\lambda_i)$ once for each iteration of the orientation solve. Since our orientation solve is interleaved with position solves, we ensure that λ_i remains within our bound of $|\mathbf{v}| < \lambda_i \leq |\mathbf{v}| + |\mathbf{b}|$ by storing λ_i on each segment as γ_i such that

$$\lambda_i = |\mathbf{v}_i| + \gamma_i|\mathbf{b}_i|, \tag{26}$$

$$\gamma_i = (\lambda_i - |\mathbf{v}_i|)|\mathbf{b}_i|^{-1}. \tag{27}$$

This linearly maps the last λ from the previous bounds (Eq. 21) into the current one, leading to an efficient refinement for λ which can then be safely interleaved with position solves. We refer to this variant as the exact solution.

Algorithm 1: Iterative solution for traditional Cosserat rod

Function `iterateLambda(`$i, \mathbf{v}, \mathbf{b}$`):`
- $\lambda \leftarrow |\mathbf{v}| + \text{clamp}(\gamma_i, 1\text{e}{-3}, 1)|\mathbf{b}|$ `// Calculate λ from γ_i`
- $\lambda \leftarrow \sqrt{|\mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda\mathbf{b}| + |\mathbf{v}|^2}$ `// Eq. 25`
- $\gamma_i \leftarrow (\lambda_i - |\mathbf{v}|)|\mathbf{b}_i|^{-1}$ `// Update γ_i`
- **return** λ

4 IMPLEMENTATION

We outline our algorithm in Algorithm 2, which alternates between orientation and position updates for each time step with a fixed number of iterations. During the orientation update, we process each segment individually while keeping the vertex positions fixed. For each segment, we compute the contribution from stretching constraints as $\mathbf{v}$ and the sum of all contributions from connected bending constraints as $\mathbf{b}$. With $\mathbf{v}$ and $\mathbf{b}$, the orientation is updated to match the local quasi-static solution either approximately using Eq. 22 or exactly using Algorithm 1. Both are guaranteed to reduce global error without any stability issues. In practice, we found that our approximation is more than sufficient in both stability and accuracy for all our test cases. Then, we fix the updated quaternions and proceed to the position update, employing second-order information to minimize the implicit Euler variational energy. Specifically, we use vertex block descent (VBD) [Chen et al. 2024] with the collision handling solution of incremental potential contact (IPC) [Li et al. 2021].

Forces and Hessians on Positions. For our position update, we can compute the forces $\mathbf{f}_{i,i}$ as the negative gradient of E_i^{ss} on $\mathbf{x}_i$.

$$\mathbf{f}_{i,i}^{\text{ss}} = -\nabla_{\mathbf{x}_i} E_i^{ss} = \frac{k_i^{\text{ss}}}{l_i}\left(\frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{l_i} - \mathbf{d}_{i,3}\right). \tag{28}$$

Algorithm 2: One time-step of our simulation

Data: Time step size h, number of iterations N

// Initialize positions

$\mathbf{x} \leftarrow \mathbf{x}^t + h\mathbf{v}^t + h^2\tilde{a}$ // [Chen et al. 2024]

// Iterative solve with N iterations

for $n \leftarrow 1$ ***to*** N **do**

  // Position update via VBD

  foreach ***vertex*** i **do**

    $\mathbf{x} \leftarrow$ position update // Eq. 6

  end

  // Our orientation update

  foreach ***segment*** i **do**

    $\mathbf{v} \leftarrow -2k_i^{\text{ss}} l_i^{-1}(\mathbf{x}_{i+1} - \mathbf{x}_i)$ // Eq. 15

    $\mathbf{b} \leftarrow k_{i-1}^{\text{bt}} \phi_{i-1} \mathbf{q}_{i-1} \mathbf{q}_{i-1}^0 + k_i^{\text{bt}} \phi_i \mathbf{q}_{i+1} \overline{\mathbf{q}}_i^0$ // Eq. 17

    $\lambda \leftarrow |\mathbf{v}| + |\mathbf{b}|$ // Eq. 22 **or** Algorithm 1

    $\mathbf{q}_i \leftarrow \mathbf{v}\mathbf{b}\mathbf{e}_3 + \lambda \mathbf{b}$ // Eq. 18

    $\mathbf{q}_i \leftarrow \mathbf{q}_i / |\mathbf{q}_i|$

  end

end

$\mathbf{v} \leftarrow (\mathbf{x} - \mathbf{x}^t)/h$ // Wrap up VBD solve

As the energy is reduced to be quadratic without the orientation component, the Hessian becomes the simple diagonal matrix

$$\nabla_{\mathbf{x}_i} \mathbf{f}_{i,i} = \frac{k_i^{\text{ss}}}{l_i^2} \mathbf{I}. \tag{29}$$

These forces and Hessians can then be used directly in the position update via VBD.

Contacts. We handle contacts with an IPC-style barrier energy with thickness r [Li et al. 2020]. For each pair of contacting segments a and b, we first compute the closest points $\mathbf{x}_i$ and $\mathbf{x}_b$ with contact normal $\mathbf{n} = (\mathbf{x}_b - \mathbf{x}_a)/|\mathbf{x}_b - \mathbf{x}_a|$. Then the contact energy $E^{\text{collision}}$ is defined through the proximity function $d = (\mathbf{x}_b - \mathbf{x}_a) \cdot \mathbf{n}/r$

$$E^{\text{collision}}(d) = \begin{cases} -k^{\text{collision}}(d-1)^2 \ln(d) & \text{if } d \le 1 \\ 0 & \text{otherwise .} \end{cases} \tag{30}$$

We additionally use a conservative vertex bound [Wu et al. 2020] to prevent the phase-through of rod segments.

Friction. We first compute the difference between the current position $\mathbf{x}^t$ and position at the previous time step $\mathbf{x}^{t-1}$ as

$$\delta\mathbf{x} = (\mathbf{x}_a^t - \mathbf{x}_a^{t-1}) - (\mathbf{x}_b^t - \mathbf{x}_b^{t-1}) \tag{31}$$

Given the collision force $\mathbf{f}^{\text{collision}}$, the friction force is modeled as a simple spring with a carefully defined stiffness $\tilde{k}^{\text{friction}}$ to replicate the Coulomb friction as

$$\mathbf{f}_i^{\text{friction}} = -\tilde{k}^{\text{friction}} (\nabla_{\mathbf{x}_i} \delta\mathbf{x}) \delta\mathbf{x}^{\perp} \tag{32}$$

where $\delta\mathbf{x}^{\perp} = (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\delta\mathbf{x}$ for the position difference along the normal direction and $\tilde{k}^{\text{friction}} = \min(k^{\text{friction}}, \ \mu|\delta\mathbf{x}^{\perp}|^{-1}\mathbf{n} \cdot \mathbf{f}^{\text{collision}})$. Finally, the friction Hessian can be computed as

$$\nabla_{\mathbf{x}_i} f_i^{\text{friction}} \approx -\tilde{k}^{\text{friction}} (\nabla_{\mathbf{x}_i} \delta\mathbf{x})(\mathbf{I} - \mathbf{n}\mathbf{n}^T)(\nabla_{\mathbf{x}_i} \delta\mathbf{x}). \tag{33}$$

As suggested by Macklin et al. [2020], we approximate the friction hessian by not differentiating through $\tilde{k}^{\text{friction}}$ and $\mathbf{n}$ for stability.

5 RESULTS

We implement our method in both C++ and CUDA on an AMD 9950x 16-core CPU with an NVIDIA RTX 3090. We conduct comparisons on the CPU with large-scale examples implemented on the GPU. We report all timings as measured per 30 fps frame. We make our source code available at jerryhsu.io/projects/StableCosseratRods

5.1 Comparison to Other Integration Methods

Fig. 5 illustrates the convergence behavior of different methods applied to a cantilevered rod. For a given stiffness, both our approximated and exact solutions achieve convergence within just 4 iterations per time step. In contrast, XPBD requires over a thousand iterations to reach a similar level of convergence. As anticipated, VBD fails to converge to the correct result even with 8 iterations, as it cannot enforce the unit quaternion constraint. We further test VBD with a linearized unit quaternion constrain (VBD w/ LC). While this modification allows convergence, it introduces numerical instability that requires double precision to achieve stable results.

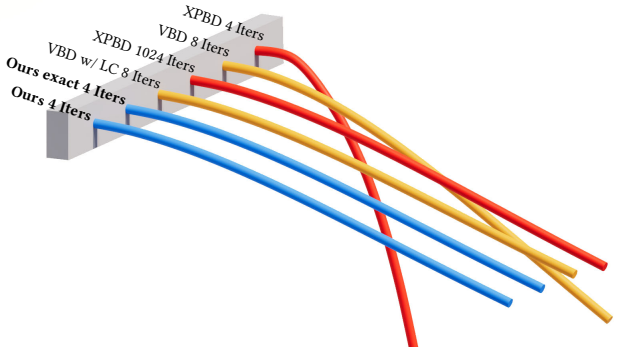

Fig. 5. *Our method converges to the target results with fewer iterations on this cantilevered rod example.*

The convergence issues are further exacerbated when simulating a much stiffer tree in Fig. 6. VBD fails to maintain the desired stiffness, while XPBD entirely fails to converge even with a step size reduced by 20×. Interestingly, our approximate solution exhibits an almost identical behavior to our exact variant. Quantitatively, our λ approximation achieves a mean squared error (MSE) of 5e−12 on the unit quaternion constraint. While our exact λ variant achieves an MSE of 8e−14, both of our methods border on the limits of single precision accuracy. Additionally, the average converged γ is 0.99999, remarkably close to our approximation of 1. These results indicate that our approximate λ is extremely close to its exact counterpart.

Fig. 7 provides an additional example of three trees waving in the wind with 9,948 vertices in total. In this example, our approximate solution takes 7.3 ms/frame, offering a slight performance advantage without compromising stability or accuracy over our exact solution at 7.5 ms/frame. For comparison, previous work using Projective Dynamics [Soler et al. 2018] and Sequential Quadratic Programming [Zhao et al. 2022] report similar computation times

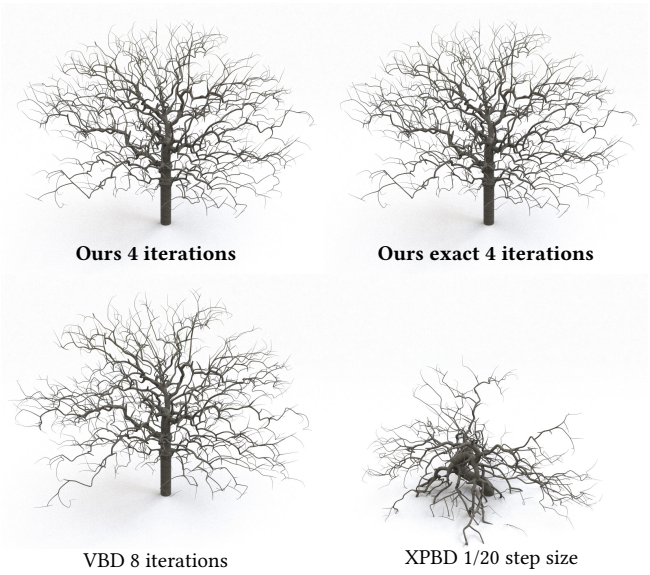

Fig. 6. *Even at 1/20th the time step, XPBD is unable to converge while VBD converges to an incorrect stiffness. In comparison, our method is far more stable and accurate with fewer iterations and larger time steps. Even our approximate solution is accurate enough to be indistinguishable from our exact formulation.*

Fig. 7. *Our method naturally supports arbitrary connections, such as three branching trees waving in the wind. Computation time: 7.3 ms (approximate) and 7.5 ms (exact) per frame for 9,948 vertices in total.*

(on different hardware) for simulations with 2 orders of magnitude fewer vertices.

5.2 Combining Different Materials

Our method can stably support a wide range of material stiffnesses. Fig. 8 illustrates a slingshot consisting of soft wrinkly elastic rubber tied to a rigid wooden handle. Despite the high k^{ss}, k^{bt}, and mass ratios between the bands and handle of 7542×, 790219×, and 46×, respectively, our method is able to complete the animation stably.

This is due in part to the accuracy of our λ approximation. When only considering the stiff handle, our approximation achieves a unit quaternion constraint MSE of 1e−11 or an average error of 0.001%.

Fig. 8. *Our method is stable across a mix of various material parameters. Here, the rubber bands simulated using our λ approximation are over one thousand times more elastic than the handle.*

For the softer rubber, our approximate λ yields an MSE of 5e−4 (1% error) compared to 1e−12 using our exact λ. In general, the stiffer the material, the more accurate our approximation. However, since the target value of the quaternion norm constraint is 1, even an MSE of 5e−4 remains functionally indistinguishable in practice, especially when the material is soft. In contrast, even VBD with linearized constraints struggles to handle this case robustly.

5.3 Comparison with Discrete Elastic Rods (DER)

We compare our method with Discrete Elastic Rods using the open source implementation provided by Fei et al. [2017]. Since DER employs a different potential, we tuned our Cosserat parameters to align the results as closely as possible for a fair comparison. Fig. 9 illustrates a slinky in the silhouette of a teapot, comprising a single chain of 2,880 segments. Since Gauss-Seidel-style methods propagate information one element at a time, this scenario represents a worst-case for methods like ours, while heavily favoring DER, which relies on a global integrator. Furthermore, instead of letting DER slowly iterate until convergence, we also use a fixed 4 iteration per 1/240s timestep with a parallelized CPU implementation for a fair comparison. Despite this, DER requires 1,052 ms per frame,

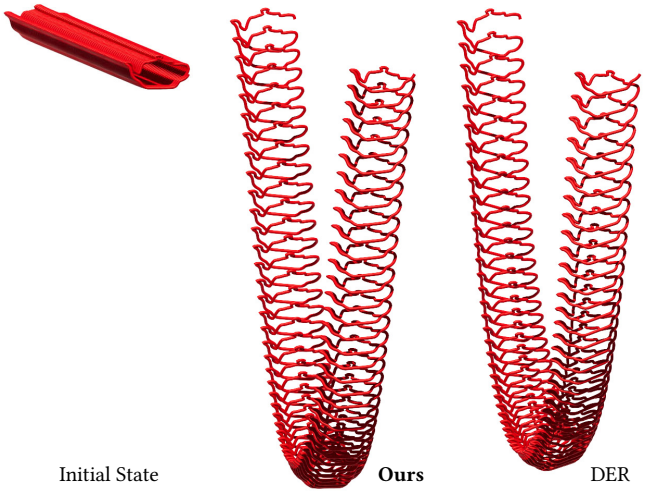

Fig. 9. *Although the Cosserat and DER energies do not match exactly, our method is able to produce similar motions as DER in this challenging example while remaining 46 times faster.*

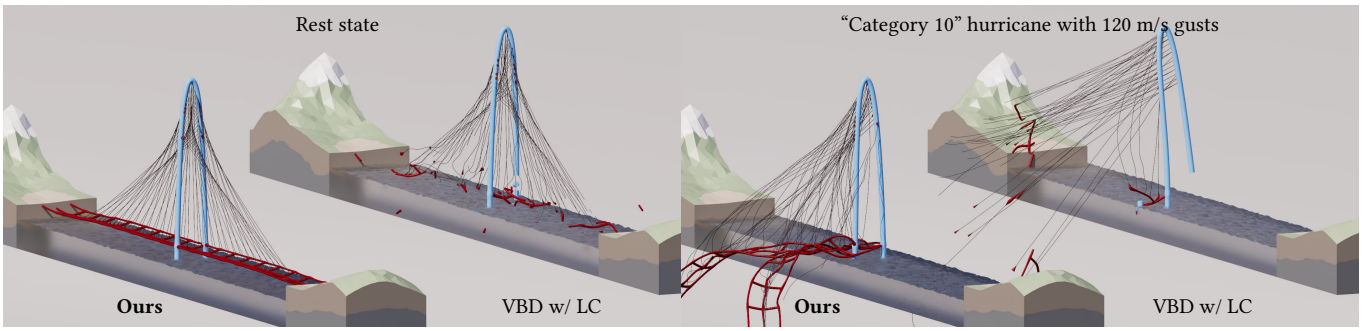

Fig. 10. *Hessian-based second-order methods like VBD can suffer from instability when Hessian becomes indefinite. Our method avoids this issue by using local analytic solutions.*

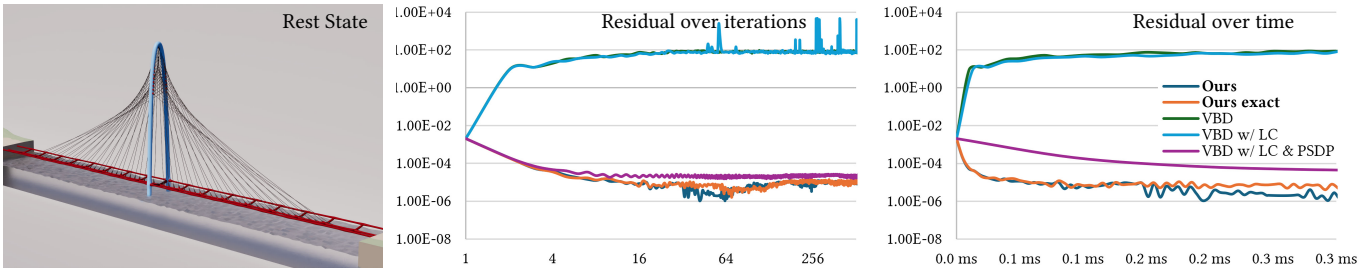

Fig. 11. *We measure the incremental potential residual for a representative frame taken from our bridge example. By the iteration, VBD requires double precision, linearized constraint solves, and PSD projections to remain stable but slow down near the solution. By computational time, our method converges the fastest by far due to the efficiency of our scheme.*

whereas our method only requires 22.8 ms per frame, which is over 46× faster than DER on the CPU. This difference in performance is due to the line-searched Newton solve used by DER to update its material frames (segment orientations).

5.4 Comparison with VBD-based methods

The energy Hessian of Cosserat rods is not only expensive to evaluate, but can also be highly unstable under specific conditions. For segment i with N bending constraints connected to it, we found the local Hessian to be positive-definite only when the following condition is satisfied:

$$\|\mathbf{x}_{i+1} - \mathbf{x}_i\| - l_0 < 2N\tilde{k}^{\text{bt}}/\tilde{k}^{\text{ss}} l_0. \tag{34}$$

Otherwise, the Hessian may become semi-definite. This instability poses significant challenges for any second-order method when segments have large $\tilde{k}^{\text{ss}}$, small $\tilde{k}^{\text{bt}}$, large strains, or large rest lengths.

Fig. 10 illustrates this with a model of the Margaret Hunt Hill Bridge consisting of 2208 vertices. This example includes a failure mechanism where steel members break if their strain exceeds 0.5%. This limit poses significant challenges for stability and accuracy. Our method successfully simulates the bridge under extreme wind conditions of 120 m/s gusts, achieving stable results at a performance rate of 2 ms per frame. In comparison, VBD with linearized constraints is unable to maintain stability, leading to immediate implosion. To address these challenges, we extended VBD (denoted as VBD w/ LC & PSDP) with double-precision floating-point arithmetic, linearized constraints, and Positive-Semi-Definite Projection (PSDP).

We examine the convergence of our methods and the VBD variants in Fig. 11. We select a representative frame from our bridge example and measure the incremental potential residual under both equal iteration count and computational time. We utilize the orientation of the previous frame as the initial guess. When compared under the same number of iterations, our method can converge to machine accuracy far faster than VBD. Both our approximate and exact variants performed nearly identically. In contrast, VBD and VBD with linearized constraints do not converge, while VBD with linearized constraints and PSD projection slow down significantly near the solution. When compared under equal computational time, the difference is further magnified. Our method converges after 0.1 ms, while VBD has barely moved at best. This is due to the extra costs of the Hessian formation, SVD decomposition, and solve. Per iteration, we measure VBD to be 109% slower, VBD with linearized constraints to be 168% slower, and VBD with all the augmentations to be over 18× slower per iteration.

In general, we found that the optimization often converges sufficiently after 4 iterations per timestep. In this example, our method's incremental potential residual converges from 2e-3 to 3e-5 after 4 iterations (vs. machine accuracy of 2e-6 after 512 iterations). As such, we use 4 iterations per timestep for most of our examples.

5.5 Knits

Fig. 12 further demonstrates the robustness our method on a knitted patch with complex yarn-yarn collisions. We clamp both ends of this cloth and twist it 5 full rotations. Despite the high yarn stiffness,

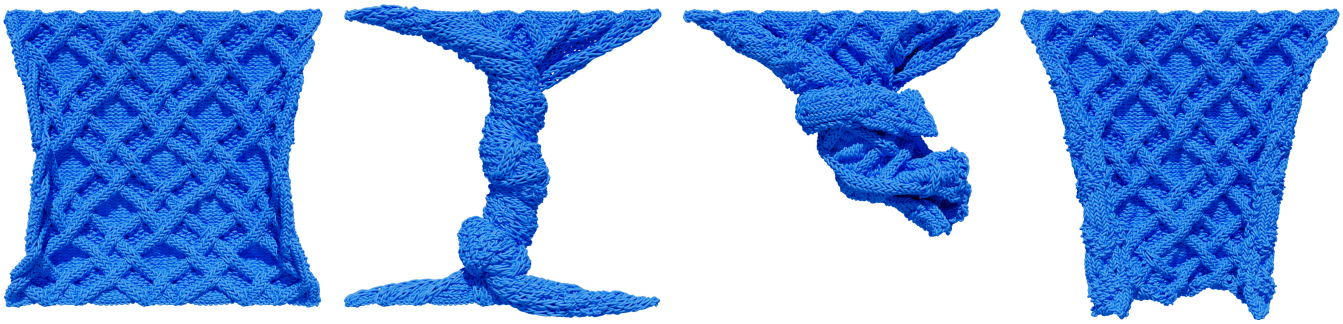

Fig. 12. *Our method is robust under complex collisions. Even after 5 full rotations, the distinctive yarn pattern is retained after release.*

Table 1. *Performance Table. Timings measure scenario 30 fps frame averages on an AMD 9950x CPU. * are measured on an NVIDIA RTX 3090 GPU. † examples include collisions. For each example, we additionally report the maximum and minimum stiffnesses and vertex masses.*

		Vertices / Segments		k^{ss}_{max} / k^{ss}_{min}	k^{bt}_{max} / k^{bt}_{min}	$\mathbf{M}_{max}$ / $\mathbf{M}_{min}$	# Itr	Step Size h	Steps/Frame	Ours Approx.	Ours Exact
Slingshot	(Fig. 8)	44 /	46	3.0e5 / 4.0e1	1.6e6 / 2.0e0	1.2e0 / 2.6e-2	4	1.0 ms	34	0.2 ms	0.2 ms
Bridge	(Fig. 10)	2,208 /	2,298	4.2e4 / 1.4e3	5.9e5 / 2.0e1	1.5e0 / 6.8e-3	4	1.0 ms	34	2.0 ms	2.1 ms
Slinky	(Fig. 9)	2,881 /	2,880	6.0e1 / 2.6e1	1.6e6 / 2.0e0	9.0e-3 / 3.9e-3	8	0.3 ms	112	22.8 ms	23.1 ms
Trees	(Fig. 7)	9,948 /	9,945	1.3e7 / 4.0e3	4.2e8 / 4.0e2	2.0e3 / 1.8e-1	4	1.0 ms	34	7.3 ms	7.5 ms
Afro*	(Fig. 13)	1,464,704 /	1,418,932	9.7e0 / 3.1e-1	1.5e3 / 6.2e1	1.0e0 / 1.0e0	8	1.0 ms	34	7.0 ms	-
Yarn Twist*†	(Fig. 12)	65,065 /	65,061	3.0e-4 / 3.0e-4	2.7e-4 / 2.7e-4	3.0e-6 / 1.5e-6	4	0.4 ms	84	32.0 ms	-
Yarn Letters*†	(Fig. 1)	255,607 /	255,593	2.1e-4 / 1.9e-4	4.1e-4 / 3.8e-4	2.1e-6 / 9.9e-7	4	0.4 ms	84	107.0 ms	-

small mass, and immense strain from these complex collisions, our method remains robust. Although some position buckling artifacts can be seen due to our piece-wise linear representation combined with the extreme stress, upon release, the yarn retains its distinctive pattern without any penetrations.

Additionally, the embarrassingly parallel nature of our method also allows for a significant performance uplift. On an NVIDIA RTX 3090 GPU, the prior knitwear simulator [Yuan et al. 2024] requires 4,480 ms per frame to twist a mesh-based yarn cloth 900 degrees. When using the same GPU, our method successfully twists a yarn-based yarn cloth 1800 degrees in just 32 ms per frame, demonstrating significant improvements in performance.

5.6 Hairs

We use our method to simulate each strand of hair in an Afro hairstyle, as shown in Fig. 13. With over 1.4 million vertices, our method only requires 7 ms per frame. Our integrator is particularly well-suited for hair simulations since each strand consists of just 32 vertices, which fits efficiently into shared memory, enabling efficient inter-thread communication in Jacobi style solves on the GPU.

5.7 Performance

Table 1 lists the performance of our method, highlighting its efficiency and scalability. One advantage of our method is the ease of parallelization. Without the requirement for large global solvers, parallelization on the GPU becomes trivial. Additionally, we found our method to work well on the GPU with simple Jacobi updates provided that position and orientation updates are handled in separate passes. We demonstrate this capability by implementing our method on an NVIDIA RTX 3090 using CUDA. In Fig. 1, this GPU-accelerated implementation enables us to simulate fully knitted

Fig. 13. *Thanks to our parallelizability, our method can simulate hair strands with over 1.4 million vertices at only 7.0 ms per frame.*

letters with 255,607 vertices in only 107 ms per frame. We can also hang the letters by selectively pulling out individual yarns, showcasing the robustness of our technique.

6 CONCLUSION

We have introduced a novel framework for simulating thin elastic rods with a focus on stability, efficiency, and simplicity. Our approach avoids reliance on complex implicit solvers to enforce orientation constraints by leveraging a split position and rotation optimization scheme. The cornerstone of our method is a closed-form Gauss-Seidel quasi-static orientation update, which is both robust and computationally efficient. Extensive validation against existing solvers demonstrates ours method's superiority, achieving improvements in performance by multiple orders of magnitude. Fi-

nally, the implementation is straightforward and requires minimal modifications beyond traditional position update schemes.

Limitations and Future work. Although we did not find any notable drawbacks in our evaluation, our method has some theoretical limitations. For example, it is unclear how stiffness parameters should be normalized under an approximate λ. In our evaluation, we simply used the same stiffness parameters for both method variants. Our approximate model may also require future work for compatibility with Sag-free initialization methods like Hsu et al. [2023]. As we make the quasi-static assumption like DER, it is also unclear how motor-driven motions can be incorporated. Finally, we tailor our method for rods with scaler stiffnesses following [Kugelstadt and Schömer 2016]. While we conjecture that our method can be extended to anisotropic stiffnesses, we leave this to future work.

ACKNOWLEDGMENTS

This project was supported in part by NSF grant #1956085.

REFERENCES

Baptiste Angles, Daniel Rebain, Miles Macklin, Brian Wyvill, Loic Barthe, Jp Lewis, Javier Von Der Pahlen, Shahram Izadi, Julien Valentin, Sofien Bouaziz, and Andrea Tagliasacchi. 2019. VIPER: Volume Invariant Position-based Elastic Rods. *Proc. ACM Comput. Graph. Interact. Tech.* 2, 2, Article 19 (July 2019), 26 pages.

Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. 2010. Discrete viscous threads. *ACM Trans. Graph.* 29, 4, Article 116 (July 2010), 10 pages.

Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. 2008. Discrete Elastic Rods. In *ACM SIGGRAPH 2008 Papers* (Los Angeles, California) *(SIGGRAPH '08)*. Association for Computing Machinery, New York, NY, USA, Article 63, 12 pages.

Florence Bertails. 2009. Linear Time Super-Helices. *Computer Graphics Forum* 28, 2 (2009), 417–426.

Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. 2006. Super-Helices for Predicting the Dynamics of Natural Hair. *ACM Trans. Graph.* 25, 3 (jul 2006), 1180–1187.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph.* 33, 4, Article 154 (July 2014), 11 pages.

Romain Casati and Florence Bertails-Descoubes. 2013. Super space clothoids. *ACM Trans. Graph.* 32, 4, Article 48 (July 2013), 12 pages.

Anka He Chen, Ziheng Liu, Yin Yang, and Cem Yuksel. 2024. Vertex Block Descent. *ACM Trans. Graph.* 43, 4, Article 116 (July 2024), 16 pages.

Gilles Daviet. 2023. Interactive Hair Simulation on the GPU using ADMM. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 24, 11 pages.

Crispin Deul, Tassilo Kugelstadt, Marcel Weiler, and Jan Bender. 2018. Direct Position-Based Solver for Stiff Rods. *Computer Graphics Forum* 37 (03 2018).

Yun (Raymond) Fei, Henrique Teles Maia, Christopher Batty, Changxi Zheng, and Eitan Grinspun. 2017. A multi-scale model for simulating liquid-hair interactions. *ACM Trans. Graph.* 36, 4, Article 56 (July 2017), 17 pages.

Xuchen Han, Theodore F. Gast, Qi Guo, Stephanie Wang, Chenfanfu Jiang, and Joseph Teran. 2019. A Hybrid Material Point Method for Frictional Contact with Diverse Materials. *Proc. ACM Comput. Graph. Interact. Tech.* 2, 2, Article 17 (July 2019), 24 pages.

Jerry Hsu, Tongtong Wang, Zherong Pan, Xifeng Gao, Cem Yuksel, and Kui Wu. 2023. Sag-Free Initialization for Strand-Based Hybrid Hair Simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2023)* 42, 4 (07 2023), 14 pages.

Jerry Hsu, Tongtong Wang, Zherong Pan, Xifeng Gao, Cem Yuksel, and Kui Wu. 2024. Real-Time Physically Guided Hair Interpolation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2024)* 43, 4, Article 95 (07 2024), 11 pages.

Li Huang, Fan Yang, Chendi Wei, Yu Ju (Edwin) Chen, Chun Yuan, and Ming Gao. 2023. Towards Realtime: A Hybrid Physics-based Method for Hair Animation on GPU. *Proc. ACM Comput. Graph. Interact. Tech.* 6, 3, Article 43 (Aug. 2023), 18 pages.

Hayley Iben, Mark Meyer, Lena Petrovic, Olivier Soares, John Anderson, and Andrew Witkin. 2013. Artistic simulation of curly hair. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Anaheim, California) *(SCA '13)*. Association for Computing Machinery, New York, NY, USA, 63–71.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2008. Simulating knitted cloth at the yarn level. In *ACM SIGGRAPH 2008 Papers* (Los Angeles, California) *(SIGGRAPH '08)*. Association for Computing Machinery, New York, NY, USA, Article 65, 9 pages.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2010. Efficient yarn-based cloth with adaptive contact linearization. *ACM Trans. Graph.* 29, 4, Article 105 (July 2010), 10 pages.

T. Kugelstadt and E. Schömer. 2016. Position and Orientation Based Cosserat Rods. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Zurich, Switzerland) *(SCA '16)*. Eurographics Association, Goslar, DEU, 169–178.

Jonathan Leaf, Rundong Wu, Eston Schweickart, Doug L. James, and Steve Marschner. 2018. Interactive design of periodic yarn-level cloth patterns. *ACM Trans. Graph.* 37, 6, Article 202 (Dec. 2018), 15 pages.

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental potential contact: intersection-and inversion-free, large-deformation dynamics. *ACM Trans. Graph.* 39, 4, Article 49 (Aug. 2020), 20 pages.

Minchen Li, Danny M. Kaufman, and Chenfanfu Jiang. 2021. Codimensional incremental potential contact. *ACM Trans. Graph.* 40, 4, Article 170 (July 2021), 24 pages.

M. Macklin, K. Erleben, M. Müller, N. Chentanez, S. Jeschke, and T. Y. Kim. 2020. Primal/dual descent methods for dynamics. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Virtual Event, Canada) *(SCA '20)*. Eurographics Association, Goslar, DEU, Article 9, 12 pages.

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. 2016. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In *Proceedings of the 9th International Conference on Motion in Games* (Burlingame, California) *(MIG '16)*. ACM, New York, NY, USA, 49–54.

Dinesh K. Pai. 2002. STRANDS: Interactive Simulation of Thin Solids using Cosserat Models. *Computer Graphics Forum* 21, 3 (2002), 347–352.

Sören Pirk, Michał Jarząbek, Torsten Hädrich, Dominik L. Michels, and Wojciech Palubicki. 2017. Interactive wood combustion for botanical tree models. *ACM Trans. Graph.* 36, 6, Article 197 (Nov. 2017), 12 pages.

Andrew Selle, Michael Lentine, and Ronald Fedkiw. 2008. A mass spring model for hair simulation. *ACM Trans. Graph.* 27, 3 (aug 2008), 1–11.

Carlota Soler, Tobias Martin, and Olga Sorkine-Hornung. 2018. Cosserat Rods with Projective Dynamics. *Computer Graphics Forum* 37, 8 (2018), 137–147.

J. Spillmann and M. Teschner. 2007. CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '07)*. Eurographics Association, Goslar, DEU, 63–72.

Nobuyuki Umetani, Ryan Schmidt, and Jos Stam. 2015. Position-Based Elastic Rods. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Copenhagen, Denmark) *(SCA '14)*. Eurographics Association, Goslar, DEU, 21–30.

Jiahao Wen, Jiong Chen, Nobuyuki Umetani, Hujun Bao, and Jin Huang. 2020. Cosserat Rod with rh-Adaptive Discretization. *Computer Graphics Forum* 39, 7 (2020), 143–154.

Longhua Wu, Botao Wu, Yin Yang, and Huamin Wang. 2020. A Safe and Fast Repulsion Method for GPU-based Cloth Self Collisions. *ACM Trans. Graph.* 40, 1, Article 5 (Dec. 2020), 18 pages.

Chun Yuan, Haoyang Shi, Lei Lan, Yuxing Qiu, Cem Yuksel, Huamin Wang, Chenfanfu Jiang, Kui Wu, and Yin Yang. 2024. Volumetric Homogenization for Knitwear Simulation. *ACM Trans. Graph.* 43, 6, Article 207 (Nov. 2024), 19 pages.

Chongyao Zhao, Jinkeng Lin, Tianyu Wang, Hujun Bao, and Jin Huang. 2022. Efficient and Stable Simulation of Inextensible Cosserat Rods by a Compact Representation. *Computer Graphics Forum* 41, 7 (2022), 567–578.