Neural-Assisted Homogenization of Yarn-Level Cloth

Xudong Feng*
xudongfeng18@gmail.com
State Key Laboratory of CAD&CG, Zhejiang University
Hangzhou, China

Huamin Wang
wanghmin@gmail.com
Style3D Research
Hangzhou, China

Yin Yang
yin.yang@utah.edu
The University of Utah
USA
Style3D Research
Hangzhou, China

Weiwei Xu†
xww@cad.zju.edu.cn
State Key Laboratory of CAD&CG, Zhejiang University
Hangzhou, China

Figure 1: An avatar outfitted in a knitted T-shirt with 14K vertices is doing Karate. We've simulated the garment using our homogenized yarn-level constitutive model, achieving 15FPS on a standard CPU. A key distinction of our model lies in its numerical stability, even with large time steps (up to 1/30 seconds). This enables our model to maintain yarn-level cloth behaviors, such as curliness, without sacrificing stability for accuracy. Most notably, our model boosts simulation efficiency, reducing computational time by at least two orders of magnitude compared to other homogenized models.

ABSTRACT

Real-world fabrics, composed of threads and yarns, often display complex stress-strain relationships, making their homogenization a challenging task for fast simulation by continuum-based models. Consequently, existing homogenized yarn-level models frequently struggle with numerical stability without line search at large time steps, forcing a trade-off between model accuracy and stability. In this paper, we propose a neural-assisted homogenized constitutive model for simulating yarn-level cloth. Unlike analytic models, a neural model is advantageous in adapting to complex dynamic behaviors, and its inherent smoothness naturally mitigates stability issues. We also introduce a sector-based warm-start strategy to accelerate the data collection process in homogenization. This model is trained using collected strain energy datasets and its accuracy is validated through both qualitative and quantitative experiments. Thanks to our model's stability, our simulator can now achieve two-orders-of-magnitude speedups with large time steps compared to previous models.

*The author was also partly affiliated with Style3D Research during this work.
†Corresponding author.

SIGGRAPH Conference Papers '24, July 27-August 1, 2024, Denver, CO, USA

ACM ISBN 979-8-4007-0525-0/24/07...$15.00
https://doi.org/10.1145/3641519.3657411

CCS CONCEPTS

• **Computing methodologies → Physical simulation**.

KEYWORDS

Yarn-level Cloth Simulation, Homogenization, Constitutive Model, Neural Networks

ACM Reference Format:

Xudong Feng, Huamin Wang, Yin Yang, and Weiwei Xu. 2024. Neural-Assisted Homogenization of Yarn-Level Cloth. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24 (SIGGRAPH Conference Papers '24), July 27-August 1, 2024, Denver, CO, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3641519.3657411

1 INTRODUCTION

Yarn-level cloth simulation, a field pioneered by Kaldor et al. [2008; 2010], significantly enhances the visual detail of knitted fabrics, showcasing features like curling effects. Subsequent researchers have introduced innovations such as persistent contact modeling [Cirio et al. 2016] and the integration of triangles with yarns in simulators [Casafranca et al. 2020] to increase simulation speed. However, despite these advancements, simulating yarn-level cloth remains computationally demanding with modern graphics hardware.

To enhance efficiency, Sperl et al. [2020] introduced a numerical homogenization method for yarn-level cloth (HYLC) simulation. This method relies on providing the yarn's physical properties, such as Young's modulus, and twisting and bending moduli, as well as the local geometric yarn pattern. A homogenization procedure is then developed to approximate the strain energy density function through yarn pattern simulation. It then defines the strain energy as a function of the combination of the first and second fundamental forms which determine the deformation of the local planar patch. We refer the fundamental forms space as HYLC strain space. HYLC employs Hermite interpolation over the strain energy density values derived from yarn pattern simulation at node points sampled in the HYLC strain space. This process forms a constitutive model, enabling highly realistic simulations in continuum-based cloth simulators.

However, even with the implementation of positive definiteness correction [Kim 2020; Kim et al. 2019; Teran et al. 2003; Wu and Kim 2023] to eliminate negative eigenvalues in the Hessian matrix, the HYLC method's simulation time step remains restricted to around 10^{-4} s in a Newton-type solver without line search, posing challenges for its application in interactive environments. This limitation is partly due to the discontinuity of second-order derivatives at the interpolated node points of the strain energy density function. While Hermite interpolation ensures gradient continuity at these node points, it does not address discontinuity in second-order derivatives.

Inspired by recent advancements in AI for science [Wang et al. 2023] and in the design of neural material models [Li et al. 2023a], we introduce a neural-assisted homogenization method for large time-step simulations of yarn-level clothing. The key insight of our approach is leveraging neural networks to allow greater flexibility in material model design. A network-based representation eliminates the need for meticulously choosing functions to describe nonlinear material behavior and overcomes the limitations of traditional spline interpolation, such as restricted-order derivative continuity at node points. Building on this concept, our method involves training neural networks with synthetic strain energy density data. We utilize the network's capacity to incorporate smooth activation functions in neurons, thereby enabling the creation of a neural network with smooth derivatives for representing the hyperelastic constitutive model. Specifically, we employ the sigmoid activation function and introduce a regularization strategy. This strategy involves penalizing the magnitude of third-order derivatives during training, which reduces oscillations in the second-order Hessian of the neural constitutive model. Such an approach significantly improves the performance of Newton-type solvers, which are prevalent in implicit simulators [Baraff and Witkin 1998].

While penalizing the third-order derivatives of Hermite interpolating functions, as used in [Sperl et al. 2020], is possible, directly optimizing the numerous coefficients of Hermite basis functions presents challenges. This difficulty is due to the high-dimensional parameter space inherent in the HYLC strain space, making the process both complex and time-consuming. From this perspective, our neural constitutive model offers a more compact and efficient representation of the high-dimensional Hermite interpolating functions within HYLC. It also addresses the issue of discontinuity in second-order derivatives at node points. Once trained, we convert our neural constitutive model back into analytic basis functions, thereby bypassing the computational overhead associated with derivative calculation through the neural network's computational graph. Furthermore, acknowledging that the behavior of yarn fluctuation in yarn-level simulation remains relatively stable across the HYLC strain space, we have developed a sector-based warm-start procedure. This approach significantly accelerates the data collection process. Our contributions are summarized below.

- *A neural constitutive model* in Section 5. We present a neural constitutive model, designed to deliver stable and realistic results in continuum-based simulations. We define the stability as simulation stability without line search. The key factor contributing to this model's stability is the smoothness of the second-order derivatives of the model represented by neural networks. Additionally, we propose an efficient, parallelized *baked* implementation, enabling seamless integration of our model into a continuum-based cloth simulation.
- *Sector-based warm-start for yarn pattern simulation* in Section 4. The essence of numerical homogenization is the derivation of the strain-energy density function from synthetic data generated by yarn pattern simulation. To fulfill this task, we introduce a sector-based warm-start strategy that significantly reduces simulation costs. This strategy leverages the deformation history of the yarn structure, leading to one order-of-magnitude speedup compared to simulation from scratch.
- *Safeguard strategy for constitutive model* in Section 6. To enhance simulation stability under significant deformations, we devised a safeguard-based strategy for the constitutive model. This method uses a near-quadratic expansion technique to extend the neural constitutive model beyond its

trained domain, thus improving the stability of the continuum-based simulator when deformations are out of trained region.

Our experiments demonstrate that a continuum-based cloth simulator with our model can achieve 15FPS simulation for 14K vertices on a desktop PC with an Intel i9-10850K CPU, as Fig. 1 shows.

2 RELATED WORK

2.1 Yarn-Level Cloth Simulation

Yarn-level simulators, as shown by Kaldor et al. [2008], offer highly realistic cloth simulations by modeling individual yarn dynamics and inter-yarn contacts. However, they require smaller time steps than continuum-based simulators due to the nonlinearity of yarn dynamics and the challenge of managing numerous contacts. Pizana et al. [2020] proposed a stable bending model for yarn dynamics to enhance stability, while incorporating dissipation energy is another effective approach [Sánchez-Banderas and Otaduy 2017, 2018].

The complexity introduced by the fine-grained modeling in yarn-level simulators is significant. The adaptive contact approach [Kaldor et al. 2010] was proposed to reduce the computational cost by reusing contact information. Subsequent works [Cirio et al. 2014, 2016] and [Sánchez-Banderas et al. 2020] adopted persistent contacts between yarns to improve performance, but this simplification limits the generality of yarn-level simulators. As an alternative, Casafranca et al. [2020] proposed a method that combines continuum-based modeling with yarn-level modeling. This hybrid approach employs yarn modeling specifically in critical regions while employing continuum-based modeling in less significant areas, offering a more flexible and efficient solution for applying yarn-level simulations.

2.2 Continuum-based Simulation and Homogenization

Since the seminal work of Baraff et al.[1998], there has been a surge of research aimed at enhancing the efficiency and realism of continuum-based cloth simulators. For example, the (extended) position-based dynamics framework [Macklin et al. 2016; Müller et al. 2007] innovates by substituting traditional cloth dynamics with positional constraints, thereby achieving stability even in the presence of high stiffness. Additionally, the projective dynamics framework [Bouaziz et al. 2014; Overby et al. 2017] integrates a local projection step with a global solving step, constituting a single iteration of a rapid solver. This framework has been further extended for parallel implementations [Li et al. 2023b; Wang and Yang 2016].

The realism of continuum-based cloth simulation depends on physical parameters. Both optimization-based methods [Bickel et al. 2009; Miguel et al. 2013; Wang et al. 2011] and learning-based methods [Feng et al. 2022; Yang et al. 2017] can be used to measure physical parameters of fabrics. Simulation realism can also be enhanced through numerical homogenization techniques derived from yarn-level simulations. Numerical homogenization involves learning the macroscale constitutive model from microscale simulations [Guedes and Kikuchi 1990]. Reviews of numerical homogenization can be found in [Geers et al. 2010; Matouš et al. 2017]. Recently, numerical homogenization [Chan-Lock et al. 2022; Fei et al. 2018; Montazeri et al. 2021; Zhang et al. 2023] has gained popularity in the graphics community. Sperl et al. [2020] proposed homogenization of yarn-level cloth with large strains, enabling the simulation of characteristic features of yarn-level cloth in continuum-based simulators.

2.3 Deep Neural Network Constitutive Model

In material science and computational physics, researchers started exploring the idea of incorporating neural networks into constitutive models, since the early work by Ghaboussi and Ellis [1992; 1991]. Unlike analytic constitutive models, neural networks, as universal approximators [Hornik et al. 1989], can represent complex functions through a few layers [Lefik and Schrefler 2003]. This capability lends them excellent realism when used in simulators based on an accurate fit for experimental data, as shown in [Li et al. 2023a]. Compared to other data-driven methods, such as piecewise linear functions [Huang et al. 2019], support vector machines, or radial basis functions, neural networks provide smoother results while maintaining simplicity in design, implementation, and control.

Providing the external boundary condition and corresponding macroscale deformation in a global setting, the neural constitutive model can be trained with a differentiable simulator [Huang et al. 2019; Xu et al. 2021]. However, this method is expensive since each network weight update necessitates a scene simulation. A more direct way is to train the neural constitutive model with sampled strain-stress or strain-energy data, for elasticity [Shen et al. 2005], elasto-plasticity [Lefik and Schrefler 2003], steels with hysteresis [Wang et al. 2022], and laminated fabrics [Gao et al. 2022]. Colasante et al. [2016] proposed a method to build a network-based constitutive model for the in-plane deformation of fabrics. Other works train the network on homogenized data, including both elasticity [Le et al. 2015] and in-elasticity [Logarzo et al. 2021].

Constitutive models trained using strain-stress or displacement-nodal force frameworks risk violating conservation laws when the stiffness matrix lacks symmetry. To address this, our approach focuses on learning strain energy density functions. This choice ensures the preservation of stiffness matrix symmetry in our method, as in [Li et al. 2023a].

3 BACKGROUND

Since our method employs the yarn pattern simulation in HYLC [Sperl et al. 2020], to prepare training data and develop neural networks to represent the macroscale strain energy density function defined in it, we briefly introduce the formulations of these two components for the purpose of clarity.

Notations. The macroscale deformation of the mid-surface, i.e., the local planar patch to which the yarn pattern is attached, is determined by the *macroscale strain* $\mathbf{s}$:

$$\mathbf{s} = \begin{bmatrix} \sqrt{I_0} - 1 & \frac{I_1}{\sqrt{I_0 I_2}} & \sqrt{I_2} - 1 & \lambda_1 & \lambda_2 & c^2 \end{bmatrix}, \quad \mathbf{I} = \begin{bmatrix} I_0 & I_1 \\ I_1 & I_2 \end{bmatrix}, \quad (1)$$

where $\mathbf{I}$ is the first fundamental form of the mid-surface, and λ_1 and λ_2 are the maximum and minimum eigenvalues of the second fundamental form, respectively. The last entry, c^2, signifies the squared cosine of the angle between the eigenvector corresponding to λ_1 and the x-axis. We denote the i-th component of $\mathbf{s}$ as s_i. For example, $s_0 = \sqrt{I_0} - 1$.

Yarn pattern simulation. It is designed to minimize the deformation energy model in the Discrete Elastic Rod (DER) [Bergou et al. 2008] method to replicate the stretch, bending, and twisting dynamics of yarns, where the contacts between yarns are resolved using Kaldor's [2008] repulsion formulation. Specifically, given a node point $\mathbf{s}$ in HYLC strain space, yarn pattern simulation can be formulated as a constrained optimization problem:

$$\varphi^*_{\mathbf{s}} = \min_{\mathbf{u}} \varphi_{\text{pat}}(\mathbf{u}), \qquad \text{s.t. } C(\mathbf{u}, \mathbf{s}) = 0, \tag{2}$$

where the yarn fluctuation vector $\mathbf{u} = [\mathbf{u}^0, \tau^0, \mathbf{u}^1, \tau^1, \ldots, \mathbf{u}^{n-1}]$, $\mathbf{u}^i$ is the Cartesian displacement of the i-th vertex with respect to the initial position defined by the deformed mid-surface, and τ^i is the twist displacement of the i-th edge. We integrate the homogenization energy, denoted as φ_{pat}, and the constraint function $C(\mathbf{u}, \mathbf{s})$, as established in [Sperl et al. 2020]. The term φ_{pat} encompasses the yarn's elastic energy and the contact potential between yarn segments, effectively representing these two energies. Meanwhile, the constraint $C(\mathbf{u}, \mathbf{s})$ serves to regulate the periodicity and the fluctuation within the yarn patterns.

Macroscale strain energy function. To circumvent the curse of dimensionality of s in the HYLC strain space, we adopt the simplification approach utilized by [Sperl et al. 2020] and [Miguel et al. 2016] , which decomposes the six-dimensional energy density function into a combination of a constant component as well as one- and two-dimensional functions. For any node point $\mathbf{s}$, we define

$$\Psi^{\text{stretch}}(\mathbf{s}) = \sum_{i=0}^{2} \Psi_{1D,i}(s_i) + \sum_{\substack{\{i,j\} \in \{\{0,1\},\\ \{0,2\},\{1,2\}\}}} \Psi_{2D,i,j}(s_i, s_j), \tag{3}$$

for

$$\begin{aligned}\Psi^{\text{bend}}_{1D}(\mathbf{s}) = c^2 \left[\Psi_{1D,3}(s_3) + \Psi_{1D,4}(s_4)\right] \\ + (1 - c^2) \left[\Psi_{1D,3}(s_4) + \Psi_{1D,4}(s_3)\right],\end{aligned} \tag{4}$$

and

$$\begin{aligned}\Psi^{\text{bend}}_{2D}(\mathbf{s}) = \sum_{i=0}^{2} \left[c^2\Psi_{i,3}(s_i, s_3) + (1 - c^2)\Psi_{i,3}(s_i, s_4)\right] \\ + \sum_{i=0}^{2} \left[c^2\Psi_{i,4}(s_i, s_4) + (1 - c^2)\Psi_{i,4}(s_i, s_3)\right],\end{aligned} \tag{5}$$

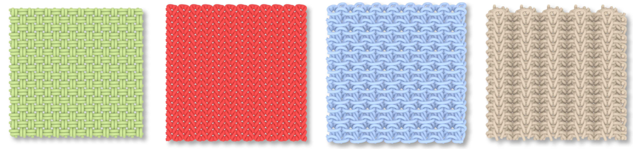
where we simplify $\Psi_{2D,ij}$ to Ψ_{ij}. The strain energy density function $\Psi(\mathbf{s})$ is defined as:

$$\Psi(\mathbf{s}) = \Psi_0 + \Psi^{\text{stretch}}(\mathbf{s}) + \Psi^{\text{bend}}_{1D}(\mathbf{s}) + \Psi^{\text{bend}}_{2D}(\mathbf{s}), \tag{6}$$

where Ψ_0 is the constant strain energy density value for the yarn pattern when $\mathbf{s} = \mathbf{0}$, i.e., no deformation is applied to the mid-surface.

4 DATA COLLECTION WITH WARM-START

We gather training data, comprising pairs of macroscale strain from the mid-surface and the corresponding strain energy density values, via yarn pattern simulation for four patterns: basket, stockinette, honeycomb, and cartridge belt rib, as illustrated in Fig. 2.



(a) Basket (b) Stockinette (c) Honeycomb (d) Cartridge

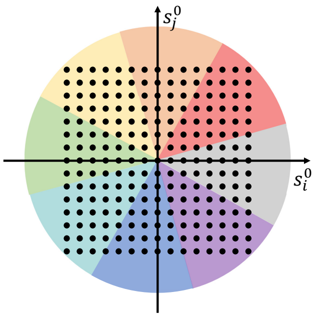
Figure 2: Four periodic yarn patterns used for evaluation purposes in our experiments. For the sake of simplicity, we abbreviated *cartridge belt rib* as *cartridge* and *slip stitch honeycomb* as *honeycomb* throughout the remainder of this paper.

The primary objective of our warm-start strategy is to expedite the pattern simulation process. This is achieved by initializing the solution at a given node point in the strain space using the solution from a neighboring point. [Zhang et al. 2023] explores a BFS-based warm start strategy for homogenizing planar flexible structures. Despite its benefits, it struggles with control and scalability due to limited parallelism. Conversely, we suggest a sector-based strategy, improving concurrency and addressing these issues. We elaborate on our method in this section.

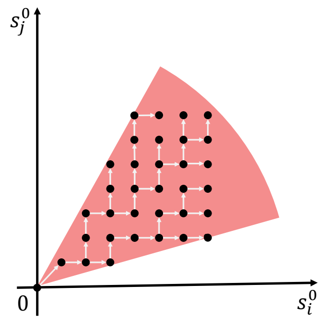
To enhance the parallelism of the warm-start procedure, we employ sectors to group node points. While a simple warm-start strategy in the HYLC strain space is to propagate the solution at a node point to its neighboring points, it is not friendly to parallel implementation. To address this issue, we refine our warm-start strategy by categorizing node points into different sectors according to their polar coordinates, as illustrated in Fig. 3 for the 2D case. In practice, we use 64 sectors in 2D and two sectors in 1D. The solution propagation can then be done with sector-level parallelism. We warm start a node point by using results from a point within the same sector to which it belongs. Within each sector, we first build an edge for a node point s_i by selecting a node j satisfying $\operatorname{argmin}_j \left\{\|s_i - s_j\| : \|s_j\| < \|s_i\|, j \in [0, N-1], j \neq i\right\}$. We use breadth-first search to traverse the node points in the sector and select the node point closest to the origin to serve as the root. This way, we can propagate the solution from node points with small strains to those with large strains.

Given the warm-start strategy for yarn pattern simulation, we must determine the sampling range and sampled node points next. We choose different sampling ranges for different components in the macroscale strain. For in-plane strain components, we set the sampling ranges as follows: $s_0, s_2 \in [-0.5, 0.8]$, and $s_1 \in [-0.5, 0.5]$. For out-of-plane strain components, we choose a sampling range of $s_3, s_4 \in [-250, 250]$. The sector-based warm-start strategy is then applied to accelerate pattern simulation by breaking the 1D sample ranges into sub-intervals and 2D sample ranges into sectors. Consequently, we obtain five 1D strain energy datasets $\Psi_{1D,i}$ for $i = \{0, 1, 2, 3, 4\}$ and nine 2D strain energy datasets $\Psi_{2D,ij}$ for network training in Sec. 5. We also calculate the derivatives $\nabla_i \Psi_{1D,i}$ for the 1D dataset and $\nabla_i \Psi_{2D,ij}$, $\nabla_j \Psi_{2D,ij}$ for the 2D dataset using finite difference methods for network training later.

Our warm-start strategy accelerates the homogenization process, yielding a tenfold increase in speed. Additionally, this strategy is versatile, extendable to any dimensional space by defining



(a) Eight sectors of strain samples.



(b) The simulation sequence (in white)

Figure 3: Strain sample sectors. As shown in (a), we categorize 2D strain samples into multiple sectors. Within each sector, we gather training data by executing yarn pattern simulation. This process involves progressively increasing strains in a specific pre-defined sequence, as shown in (b).

an N-sphere. Please refer to the supplementary document for more details about yarn pattern simulation and data collection.

5 NEURAL CONSTITUTIVE MODEL TRAINING

We choose to represent each 1D and 2D function in Eq. 5 with separate neural networks. This choice is due to their compactness compared to other bases like Hermite interpolation. For example, in the case of 2D functions with 101x101 sampling points, a neural network with 1,137 parameters can represent the relationships while we need 16,000 coefficients for bicubic Hermite interpolation. Specifically, we use five neural networks $\hat{\Psi}_i(s_i)$ for 1D functions and nine neural networks $\hat{\Psi}_{ij}(s_i, s_j)$ for 2D functions. Specifically, there are a total of five neural networks $\hat{\Psi}_i(s_i)$ used to represent 1D functions and nine neural networks $\hat{\Psi}_{ij}(s_i, s_j)$ used to represent 2D functions. These trained networks are summed together to form the neural constitutive model used in our work. The network architecture is simply a three-layer MLP with a sigmoid activation function at each neuron. We will use $\hat{\Psi}_I(\mathbf{s}_I)$ to denote both 1D and 2D functions when the context is clear. A sample point in the dataset is denoted as $\mathbf{s}_I^d$ correspondingly.

The total loss function involved in network training is a weighted combination of four loss terms:

$$L_I^{\text{final}}(\mathbf{s}_I;\epsilon) = w^Z L_I^Z(\mathbf{s}_I) + w^F L_I^F(\mathbf{s}_I;\epsilon) + w^C L_I^C(\mathbf{s}_I;\epsilon) + w^S L_I^S(\mathbf{s}_I). \tag{7}$$

Next, we provide the purpose and the definition of each term. We justify the network architecture and offer training details in supplementary materials.

5.1 Zero-Order Prediction Loss

This loss term is designed to prompt the networks to reproduce the values of the strain energy density function at the points sampled in the training dataset, resulting in:

$$L_I^Z(\mathbf{s}_I) = \frac{1}{M}\sum_{d=0}^{M}\left(\Psi_I^d - \hat{\Psi}_I(\mathbf{s}_I^d)\right)^2, \tag{8}$$

where M is the number of samples, and d indices through the samples in the training data.

5.2 First-Order Prediction Loss

This loss term penalizes the deviation of the first-order derivatives of $\hat{\Psi}_I$ with respect to the derivatives calculated for points in the training data. It is important for the approximation accuracy of the trained networks. To ease the implementation, we leverage finite differences to approximate the first-order derivatives of neural networks and the sampled points. Consequently, for the 1D neural constitutive model $\hat{\Psi}_i$, we have $G(\hat{\Psi}_i, s_i;\epsilon) = (\hat{\Psi}_i(s_i+\epsilon) - \hat{\Psi}_i(s_i))/\epsilon$. For 2D functions $\hat{\Psi}_{ij}$, we approximate their derivatives as follows: $G_i(\hat{\Psi}_{ij}, s_i, s_j;\epsilon) = (\hat{\Psi}_{ij}(s_i+\epsilon, s_j) - \hat{\Psi}_{ij}(s_i, s_j))/\epsilon$ and $G_j(\hat{\Psi}_{ij}, s_i, s_j;\epsilon) = (\hat{\Psi}_{ij}(s_i, s_j+\epsilon) - \hat{\Psi}_{ij}(s_i, s_j))/\epsilon$. The small interval ϵ is set to 10^{-3} by default. The first-order prediction loss L_I^F for networks $\hat{\Psi}_I$ is formulated as:

$$L_I^F(\mathbf{s}_I;\epsilon) = \frac{1}{M}\begin{cases} \sum_{d=0}^{M}(G(\hat{\Psi}_{ij}, s_i^d;\epsilon) - \nabla_i \Psi_{1D,i}^d)^2, & \text{if } I = i, \\ \sum_{d=0}^{M}\Big((G_i(\hat{\Psi}_{ij}, s_i^d, s_j^d;\epsilon) - \nabla_i \Psi_{2D,ij}^d)^2 & \text{if } I = \{i,j\}, \\ \quad + (G_j(\hat{\Psi}_{ij}, s_i^d, s_j^d;\epsilon) - \nabla_j \Psi_{2D,ij}^d)^2\Big), & \end{cases} \tag{9}$$

where $\nabla_i \Psi_{1D,i}$ and $\nabla_j \Psi_{2D,ij}$ denote the first-order derivatives computed for sample points.

5.3 Third-Order Derivative Loss

This loss is employed to penalize the third-order derivatives of networks. By minimizing the magnitude of third-order derivatives, it can make the quadratic approximation of its function in the local region much more accurate. Thus, it is critical for the stability of the simulator in a large time step without line search. Empirically, we observe that applying the third-order derivative loss effectively suppresses the oscillation of second-order derivatives in elastic energy. For these networks trained on the 1D strain dataset, it is easy to estimate and penalize their third-order derivatives:

$$L_i^C(\mathbf{s};\epsilon) = \frac{1}{2\epsilon^3}\left|\hat{\Psi}_i(s_i+2\epsilon) - 2\hat{\Psi}_i(s_i+\epsilon) + 2\hat{\Psi}_i(s_i-\epsilon) - \hat{\Psi}_i(s_i-2\epsilon)\right|^2. \tag{10}$$

For constitutive networks trained on 2D strains, there are four unique third-order derivatives of $\hat{\Psi}_{ij}$, which can be selected as $\partial^3\hat{\Psi}_{ij}/\partial s_i^3$, $\partial^3\hat{\Psi}_{ij}/\partial s_i^2\partial s_j$, $\partial^3\hat{\Psi}_{ij}/\partial s_i\partial s_j^2$, $\partial^3\hat{\Psi}_{ij}/\partial s_j^3$. For instance,

$$\frac{\partial^3\hat{\Psi}_{ij}}{\partial s_i^3} \approx \frac{1}{2\epsilon^3}\Big(\hat{\Psi}_{ij}(s_i+2\epsilon, s_j) - 2\hat{\Psi}_{ij}(s_i+\epsilon, s_j) + 2\hat{\Psi}_{ij}(s_i-\epsilon, s_j) - \hat{\Psi}_{ij}(s_i-2\epsilon, s_j)\Big), \tag{11}$$

and the third-order derivative loss for neural networks that represent 2D functions is formulated as follows:

$$L_{ij}^C(s_i, s_j;\epsilon) = \left(\frac{\partial^3\hat{\Psi}_{ij}}{\partial {s_i}^3}\right)^2 + \left(\frac{\partial^3\hat{\Psi}_{ij}}{\partial {s_i}^2\partial s_j}\right)^2 + \left(\frac{\partial^3\hat{\Psi}_{ij}}{\partial s_i\partial {s_j}^2}\right)^2 + \left(\frac{\partial^3\hat{\Psi}_{ij}}{\partial {s_j}^3}\right)^2. \tag{12}$$

A general form of third-order derivative loss can be expressed as

$$L_I^C(\mathbf{s}_I;\epsilon) = \frac{1}{M}\begin{cases}\sum_{d=0}^{M} L_i^C(s_i^d;\epsilon), & \text{if } I = i,\\ \sum_{d=0}^{M} L_{ij}^C(s_i^d, s_j^d;\epsilon), & \text{if } I = \{i,j\}.\end{cases} \tag{13}$$

5.4 Strain Concentration Loss

The primary objective of this loss is to regulate the network's behavior for points that lie outside the effective area covered by the training data. In the absence of ground-truth values for the strain energy density function at these external points, inspired by the fitting strategy in [Sperl et al. 2020], we impose a different constraint: the negative gradients at these points should approximately direct towards the point $\mathbf{s} = \mathbf{0}$. This approach ensures that the strain values do not increase outside the sampled region, but rather concentrate around the central point $\mathbf{s} = \mathbf{0}$. This concentration is achieved as the deformation energy is minimized, adhering to the gradient constraints. The formulation of this loss is:

$$L_I^S(\mathbf{s}_I) = \frac{1}{M}\sum_{d=0}^{M}\begin{cases} D(G_i(\hat{\Psi}_i, s_i^d;\epsilon), s_i^d) & \text{if } I = i\\ D(G_i(\hat{\Psi}_{ij}, s_i^d, s_j^d;\epsilon), s_i^d) & \\ \quad + D(G_j(\hat{\Psi}_{ij}, s_i^d, s_j^d;\epsilon), s_j^d) & \text{if } I = \{i,j\},\end{cases} \tag{14}$$

where $D(x,t) = (1-\delta_{\mathsf{Sign}(x),\mathsf{Sign}(t)})|x - \mathsf{Sign}(t)|^2$. G denotes the function used to compute derivatives, as in the first-order prediction loss.

6 NETWORK BAKING AND SAFEGUARDING

Next we discuss practical issues involved in the use of our neural model, including network baking for fast runtime performance, and safeguarding when deformations are beyond the sampled region.

6.1 Network Baking

Conducting neural network inferences on the fly at each time step would be too expensive in continuum-based cloth simulation. The goal of network baking is to avoid runtime inferencing by converting the network defined in HYLC strain space back to Hermite interpolating functions, which can significantly boost the simulation performance. In the Karate demo in Fig. 7, the simulation cost is reduced from 510 ms (without baking) to 65ms (with baking) per frame. Note that baking back to Hermite interpolating functions does not lead to large second-order discontinuities. Since the network is trained with a third-order derivative loss, the function that the network represents in each sub-region can be well approximated by quadratic functions realized by Hermite interpolating functions.

The baking can be achieved by evaluating the function values and first-order derivatives at sampled points for 1D or 2D stretching and bending functions, and piecewise Hermite interpolating functions are constructed for each sub-interval in 1D or sub-squared region in 2D. These functions can be directly located according to the vector $\mathbf{s}$ and calculated in the continuum-based simulator.

In our experiments, we use 200 cubic Hermite functions for 1D and 100×100 bi-cubic Hermite functions for 2D.

6.2 Safeguarding Strategy

When applying the learned strain energy function to continuum-based cloth simulation, the macroscale strain of a triangle might fall outside the region covered by the sampled node points. It can lead to unstable simulation results if not carefully handled. Therefore, we propose a safeguarding strategy to enforce the Hessian matrix for these strain points to be close to the points on the boundary of the sampled region. This is realized by constructing analytic quadratic functions that equate their function values and derivatives to the derivatives at the boundary points of the sampled region in a sector-based manner. Subsequently, these functions are used as the strain energy density functions for those points outside the sampled region. This strategy mitigates discrepancies in second-order derivatives of constitutive models within and outside the sampled region, thus improves the stability of the simulation.

6.2.1 One-dimensional case. Let $[s_i^{\min}, s_i^{\max}]$ be the sample range for a neural network $\hat{\Psi}_i$ that represents one of the 1D stretching and bending functions in Eq. 5. We first calculate the function values $(\hat{\Psi}_i^{\min}, \hat{\Psi}_i^{\max})$, first-order derivative $(g_i^{\min}, g_i^{\max})$, and second-order derivatives $(h_i^{\min}, h_i^{\max})$ at its endpoints. With these quantities, we then construct a quadratic function $S_{1\mathrm{D},i}$ as

$$S_{1D,i}(s_i) = \begin{cases} a_{\min}s_i^2 + b_{\min}s_i + c_{\min}, & \text{if } \; s_i < s_i^{\min},\\ a_{\max}s_i^2 + b_{\max}s_i + c_{\max}, & \text{if } \; s_i > s_i^{\max},\\ \hat{\Psi}_i(s_i) & \text{otherwise.}\end{cases} \tag{15}$$

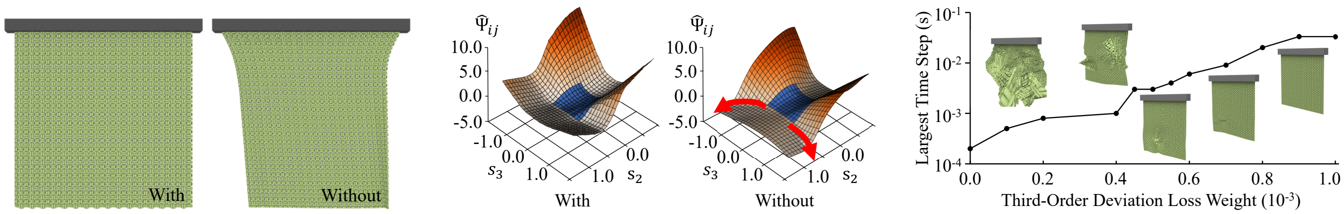
where the coefficients for $s_i < s_i^{\min}$ are computed by equating the function value and derivative of the quadratic function to these values of the network $\hat{\Psi}_{1\mathrm{D},i}$ at $s_i^{\min}$, which yields:

$$\begin{cases} a_{\min} = \frac{1}{2}h_i^{\min}, & b_{\min} = g_i^{\min} - 2a_{\min}s_i^{\min},\\ c_{\min} = f_i^{\min} - a_{\min}(s_i^{\min})^2 - b_{\min}s_i^{\min}. & \end{cases} \tag{16}$$

The coefficients, $a_{\max}$, $b_{\max}$, $c_{\max}$, can be computed in the same way. It can be verified that the above safeguard function maintains C^2 continuity at the endpoints of 1D sample ranges.

6.2.2 Two-dimensional case. Unlike the 1D case, there are a large number of boundary points in the 2D case, and we must determine the choice of boundary points before constructing the quadratic energy density function for a point $x = (s_i, s_j)$ outside the sampled region. Therefore, we choose to divide the 2D plane into N sectors, similar to the warm-starting approach, and construct a quadratic function at each edge between two neighboring sections. Once constructed, the quadratic function for any point x is computed through the linear blending of two functions constructed at the two boundary edges of the sector to which x belongs. This design transforms the 2D quadratic function construction problem into several 1D problems.

Suppose we divide the 2D sample region into N sectors uniformly around the origin. For the k-th sector, it contains all of the points with their polar angles between ϕ_k and ϕ_{k+1}, where $\phi_k = \frac{2\pi}{N}k$. If we have constructed 2D quadratic functions, G_{ij}^k and G_{ij}^{k+1}, at its left and right edges, respectively, the quadratic energy density function $S_{2\mathrm{D},ij}(s_i, s_j)$ for a point (s_i, s_j) in this sector but



(a) With and without the first-order prediction loss L^F.

(b) With and without the strain concentration loss L^B.

(c) The effect of the third-order derivation loss weight w^C.

Figure 4: Ablation studies conducted on the hanging simulation of a 20cm×20cm fabric sample. We demonstrate the crucial roles of three specific losses: the first-order prediction loss, the strain concentration loss, and the third-order derivative loss. Each of these losses contributes uniquely to the accuracy and stability of the simulation, highlighting their importance in our model.

outside the dataset's effective region Ω is

$$S_{2D,ij}(s_i, s_j) = \begin{cases} wG_{ij}^k(s_i, s_j) + (1-w)G_{ij}^k(s_i, s_j), & \text{if } (s_i, s_j) \notin \Omega \\ \hat{\Psi}_{ij}(s_i, s_j), & \text{otherwise} \end{cases} \tag{17}$$

We construct G_{ij}^k by equating the function value, gradient, and Hessian to the endpoint of the sector boundary edge, similar to the 1D case. w is the polar barycentric coordinate of the point in this sector. Given the function value f_k, the gradient $\mathbf{g}_k$, and the Hessian matrix $\mathbf{H}_k$ of $\hat{\Psi}_{ij}$, we calculate G_{ij}^k as

$$G_{ij}^k(s_i, s_j) = \begin{bmatrix} s_i & s_j \end{bmatrix} \mathbf{A}_k \begin{bmatrix} s_i \\ s_j \end{bmatrix} + \begin{bmatrix} s_i & s_j \end{bmatrix} \mathbf{b}_k + c_k, \tag{18}$$

in which

$$\begin{cases} \mathbf{A}_k = \frac{1}{2}\mathbf{H}_k, & \mathbf{b}_\mathbf{k} = \mathbf{g}_k - 2\mathbf{A}_k\mathbf{t}_k \\ c_k = f_k - \mathbf{t}_k^T\mathbf{A}_k\mathbf{t}_k - \mathbf{b}_k^T\mathbf{t}_k. \end{cases} \tag{19}$$

In the 2D case, the employed strategy generates a piecewise linear approximation along the boundary, which results in the discontinuity of the energy function at the boundary. Although this discontinuity cannot be eliminated, it can be mitigated by increasing the number of sectors. We find that the expansion approach successfully smooths out the differences in the second-order derivatives of constitutive models inside and outside the trained region, improving the simulation stability. In our implementation, we set the number of sectors to 128, which works well empirically.

Table 1: Quadratic expansion error E analysis for Stockinette pattern. This analysis shows our model has a lower quadratic expansion error, i.e., smaller magnitudes of third-order derivatives. For the definition of quadratic expansion error E and details on its computation, please refer to the supplementary material.

Type	Basket	Stockinette	Honeycomb	Cartridge
[Sperl et al. 2020]	1.31e-3	2.67e-3	5.37e-3	1.60e-3
Ours	**6.28e-4**	**4.05e-4**	**1.73e-3**	**1.26e-3**

7 RESULTS

(Please refer to the supplemental video and document for additional examples.) Our continuum-based simulator runs on CPUs and it utilizes implicit Euler time integration in conjunction with our neural constitutive model. It employs a hierarchical grid [Fan et al. 2011] for proximity search and handles contacts by impulse-based approaches [Bridson et al. 2002; Narain et al. 2012]. In this paper, we define stability as the implicit simulation stability without line search. By default, we use Newton's method to solve time integration with no backtracking line search. (The step size is fixed at one.)

In the simulation of a T-shirt with 14K vertices, as shown in Fig. 7, with a time step of $\Delta t = 1/30$s, the computational time for one timestep is divided as follows: 16ms for calculating the triangles' fundamental forms [Grinspun et al. 2006]; 10ms for accessing the constitutive model; 15ms for matrix assembly and solving linear systems; and 26ms for collision detection and handling.

7.1 Ablation Studies

In ablation studies, we focus on the hanging simulation of a square fabric sample. Figure 4(a) illustrates the significance of the first-order prediction loss, L^F, in preventing distortions in the reference configuration, which are typically caused by incorrect internal forces. Figure 4(b) highlights the necessity of the strain concentration loss, L^B, for maintaining a monotonically increasing strain energy density function outside of the sampled region (in blue). Absence of this loss leads to an incorrect decrease in energy as strain intensifies (in red arrows). Finally, Figure 4(c) demonstrates that increasing the weight of the third-order derivative loss, w^C, from 0 to 10^{-3} enhances the largest stable time step from 1/5000s to 1/30s.

7.2 Stability Evaluation

A key reason for the stability of our simulations is attributed to our model's smoother third-order derivatives, as shown in Fig. 5(a). This characteristic enables our model to provide an accurate quadratic expansion. To support this claim, we uniformly sampled 10K points from each 2D baked constitutive model, integrated their quadratic expansions, and conducted a comparative analysis. The results, presented in Table 1, reveal that the quadratic expansion error of our model is 20 to 80 percent lower than that of [Sperl et al. 2020].

Without backtracking line search, the stability issue in a constitutive model becomes apparent through its inability to perform simulations at large time steps. This is demonstrated in the two animated examples in Fig.6. In these examples, simulations using our model run robustly with a time step of $\Delta t = 1/30$s. In contrast, simulations employing the HYLC model, as proposed in [Sperl et al. 2020], fail when the time step is $\Delta t = 1/1000$s only. Additionally, Fig. 7 highlights our model's capability to simulate knitted garments on rapidly moving avatars, when using large time steps.

With backtracking line search, simulations should be able to run stably at any large time step. However, the challenge shifts to determining how small the step size should be to ensure stability. As depicted in Fig.5(b), our model maintains stability with $\Delta t = 1/30$s by simply setting the step size to one, as expected. In contrast, the HYLC model requires a significantly smaller minimal step size to achieve stability. This disparity is also observed when our simulator employs the gradient descent solver with Hessian preconditioning [Wang and Yang 2016]. This suggests that the stability issue is a universal concern, independent of the choice of solver.

7.3 Accuracy Evaluation

To assess the accuracy of our model within continuum-based simulators, we executed an experiment involving the simulation of stretching a stockinette fabric strip measuring 5cm by 12cm, in both the course and wale directions. For comparison, we use a ground truth generated by a DER-based yarn-level simulator [Bergou et al. 2008]. As depicted in Fig. 8(a) and 8(b), our model's simulation closely mirrors the real-world Poisson and curling effects observed in the middle of the fabric strip, with Hausdorff distances to the ground truth being 0.69cm and 0.72cm, respectively. This contrasts with the results from the HYLC model proposed in [Sperl et al. 2020], which deviates from the ground truth, with Hausdorff distances exceeding 1.0cm. Furthermore, Fig. 8(c) compares the relationship between force density and stretch ratio as predicted by our constitutive model and the HYLC model. This comparison reveals that our model's predictions align more closely with the ground truth.

8 CONCLUSIONS, LIMITATIONS AND FUTURE WORK

This study presents a novel neural-assisted homogenization method for yarn-level cloth, enhancing both efficiency and precision in continuum-based simulations. Using sector-based strategies and a neural constitutive model, our simulator exhibits remarkable stability with larger time steps while maintaining accuracy, as validated by qualitative experiments.

Our method uses synthetic data from yarn simulations, not real data, impacting the realism of our simulations. Our model omits higher-dimensional strain energy components, affecting accuracy. The considerable costs of data collection and network training, essential for fabric design, are noteworthy. We plan to address these limitations soon, particularly by considering real-world data collection. Our ultimate goal is a unified model for wider applications, expected to enhance our method's realism, accuracy, and usability.

REFERENCES

David Baraff and Andrew Witkin. 1998. Large Steps in Cloth Simulation. In *Proceedings of SIGGRAPH 98 (Computer Graphics Proceedings, Annual Conference Series)*, Eugene Fiume (Ed.). ACM, 43–54.

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)*. Article 63, 12 pages.

Bernd Bickel, Moritz Bächer, Miguel A. Otaduy, Wojciech Matusik, Hanspeter Pfister, and Markus Gross. 2009. Capture and Modeling of Non-linear Heterogeneous Soft Tissue. *ACM Trans. Graph. (SIGGRAPH)* 28, 3, Article 89 (July 2009), 9 pages.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans. Graph. (SIGGRAPH)* 33, 4, Article 154 (July 2014), 11 pages.

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Trans. Graph. (SIGGRAPH)* 21, 3 (July 2002), 594–603.

Juan J Casafranca, Gabriel Cirio, Alejandro Rodríguez, Eder Miguel, and Miguel A Otaduy. 2020. Mixing Yarns and Triangles in Cloth Simulation. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 101–110.

Antoine Chan-Lock, Jesús Pérez, and Miguel A Otaduy. 2022. High-Order Elasticity Interpolants for Microstructure Simulation. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 63–74.

Gabriel Cirio, Jorge Lopez-Moreno, David Miraut, and Miguel A. Otaduy. 2014. Yarn-Level Simulation of Woven Cloth. *ACM Trans. Graph. (SIGGRAPH Asia)* 33, 6, Article 207 (Nov. 2014), 11 pages.

Gabriel Cirio, Jorge Lopez-Moreno, and Miguel A Otaduy. 2016. Yarn-Level Cloth Simulation with Sliding Persistent Contacts. *IEEE transactions on visualization and computer graphics* 23, 2 (2016), 1152–1162.

G Colasante and PD Gosling. 2016. Including Shear in a Neural Network Constitutive Model for Architectural Textiles. *Procedia Engineering* 155 (2016), 103–112.

Glenn W Ellis, Chengwan Yao, and Rongda Zhao. 1992. Neural Network Modeling of the Mechanical Behavior of Sand. In *Engineering Mechanics*. ASCE, 421–424.

Wenshan Fan, Bin Wang, Jean-Claude Paul, and Jiaguang Sun. 2011. A Hierarchical Grid Based Framework for Fast Collision Detection. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1451–1459.

Yun Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2018. A Multi-Scale Model for Simulating Liquid-Fabric Interactions. *ACM Trans. Graph.* 37, 4 (2018), 1–16.

Xudong Feng, Wenchao Huang, Weiwei Xu, and Huamin Wang. 2022. Learning-Based Bending Stiffness Parameter Estimation by a Drape Tester. *ACM Trans. Graph.* 41, 6 (2022), 1–16.

Minjun Gao, Junhui Meng, Nuo Ma, Moning Li, and Li Liu. 2022. Artificial Neural Network–Based Constitutive Relation Modelling for the Laminated Fabric Used in Stratospheric Airship. *Composites and Advanced Materials* 31 (2022), 26349833211073146.

Marc G. D. Geers, Varvara G. Kouznetsova, and W. A. M. Brekelmans. 2010. Multi-Scale Computational Homogenization: Trends and Challenges. *J. Comput. Appl. Math.* 234, 7 (2010), 2175–2182.

Jamshid Ghaboussi, J. H. Garrett Jr, and Xiping Wu. 1991. Knowledge-Based Modeling of Material Behavior with Neural Networks. *Journal of Engineering Mechanics* 117, 1 (1991), 132–153.

Eitan Grinspun, Yotam Gingold, Jason Reisman, and Denis Zorin. 2006. Computing Discrete Shape Operators on General Meshes. In *Computer Graphics Forum*, Vol. 25. Wiley Online Library, 547–556.

JoséMiranda Guedes and Noboru Kikuchi. 1990. Preprocessing and Postprocessing for Materials Based on the Homogenization Method with Adaptive Finite Element Methods. *Computer Methods in Applied Mechanics and Engineering* 83, 2 (1990), 143–198.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks* 2, 5 (1989), 359–366.

Daniel Z Huang, Kailai Xu, Charbel Farhat, and Eric Darve. 2019. Predictive Modeling with Learned Constitutive Laws from Indirect Observations. *arXiv preprint arXiv:1905.12530* (2019).

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2008. Simulating Knitted Cloth at the Yarn Level. *ACM Trans. Graph. (SIGGRAPH)* 27, 3, Article 65 (Aug. 2008), 9 pages.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2010. Efficient Yarn-Based Cloth with Adaptive Contact Linearization. *ACM Trans. Graph. (SIGGRAPH)* 29, 4, Article 105 (July 2010), 10 pages.

Theodore Kim. 2020. A Finite Element Formulation of Baraff-Witkin Cloth. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 171–179.

Theodore Kim, Fernando De Goes, and Hayley Iben. 2019. Anisotropic Elasticity for Inversion-Safety and Element Rehabilitation. *ACM Trans. Graph.* 38, 4 (2019), 1–15.

BA Le, Julien Yvonnet, and Q-C He. 2015. Computational Homogenization of Nonlinear Elastic Materials Using Neural Networks. *Internat. J. Numer. Methods Engrg.* 104, 12 (2015), 1061–1084.

Marek Lefik and Bernhard A Schrefler. 2003. Artificial Neural Network as an Incremental Non-linear Constitutive Model for a Finite Element Code. *Computer Methods in Applied Mechanics and Engineering* 192, 28-30 (2003), 3265–3283.

Xuan Li, Yu Fang, Lei Lan, Huamin Wang, Yin Yang, Minchen Li, and Chenfanfu Jiang. 2023b. Subspace-Preconditioned GPU Projective Dynamics with Contact for Cloth Simulation. In *SIGGRAPH Asia 2023 Conference Papers*. Article 1, 12 pages.

Yue Li, Stelian Coros, and Bernhard Thomaszewski. 2023a. Neural Metamaterial Networks for Nonlinear Material Design. *ACM Trans. Graph.* 42, 6, Article 186 (dec 2023), 13 pages.

Hernan J Logarzo, German Capuano, and Julian J Rimoli. 2021. Smart Constitutive Laws: Inelastic Homogenization Through Machine Learning. *Computer Methods in Applied Mechanics and Engineering* 373 (2021), 113482.

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*. 49–54.

Karel Matouš, Marc GD Geers, Varvara G Kouznetsova, and Andrew Gillman. 2017. A Review of Predictive Nonlinear Theories for Multiscale Modeling of Heterogeneous Materials. *J. Comput. Phys.* 330 (2017), 192–220.

Eder Miguel, David Miraut, and Miguel A. Otaduy. 2016. Modeling and Estimation of Energy-Based Hyperelastic Objects. *Comput. Graph. Forum (Eurographics)* 35, 2 (May 2016), 385–396.

Eder Miguel, Rasmus Tamstorf, Derek Bradley, Sara C. Schvartzman, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Steve Marschner, and Miguel A. Otaduy. 2013. Modeling and Estimation of Internal Friction in Cloth. *ACM Trans. Graph. (SIGGRAPH Asia)* 32, 6, Article 212 (Nov. 2013), 10 pages.

Zahra Montazeri, Chang Xiao, Yun Fei, Changxi Zheng, and Shuang Zhao. 2021. Mechanics-Aware Modeling of Cloth Appearance. *IEEE Transactions on Visualization and Computer Graphics* 27, 1 (2021), 137–150. https://doi.org/10.1109/TVCG.2019.2937301

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position Based Dynamics. *Journal of Visual Communication and Image Representation* 18, 2 (2007), 109–118.

Rahul Narain, Armin Samii, and James F. O'Brien. 2012. Adaptive Anisotropic Remeshing for Cloth Simulation. *ACM Trans. Graph. (SIGGRAPH Asia)* 31, 6, Article 152 (Nov. 2012), 10 pages.

Matthew Overby, George E Brown, Jie Li, and Rahul Narain. 2017. ADMM $supseteq$ projective dynamics: Fast simulation of hyperelastic models with dynamic constraints. *IEEE transactions on visualization and computer graphics* 23, 10 (2017), 2222–2234.

José M Pizana, Alejandro Rodríguez, Gabriel Cirio, and Miguel A Otaduy. 2020. A Bending Model for Nodal Discretizations of Yarn-Level Cloth. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 181–189.

Rosa María Sánchez-Banderas and Miguel A Otaduy. 2017. Dissipation Potentials for Yarn-Level Cloth. In *Spanish Computer Graphics Conference (CEIG)*. 11–18.

Rosa Maria Sánchez-Banderas and Miguel A Otaduy. 2018. Strain Rate Dissipation for Elastic Deformations. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 161–170.

Rosa M Sánchez-Banderas, Alejandro Rodríguez, Héctor Barreiro, and Miguel A Otaduy. 2020. Robust Eulerian-on-Lagrangian Rods. *ACM Trans. Graph.* 39, 4 (2020), 59–1.

Yuelin Shen, K Chandrashekhara, WF Breig, and LR Oliver. 2005. Finite Element Analysis of V-Ribbed Belts Using Neural Network Based Hyperelastic Material Model. *International Journal of Non-Linear Mechanics* 40, 6 (2005), 875–890.

Georg Sperl, Rahul Narain, and Chris Wojtan. 2020. Homogenized Yarn-Level Cloth. *ACM Trans. Graph. (SIGGRAPH)* 39, 4, Article 48 (July 2020), 16 pages.

J. Teran, S. Blemker, V. Ng Thow Hing, and R. Fedkiw. 2003. Finite Volume Methods for the Simulation of Skeletal Muscle. In *Proceedings of SCA*. 68–74.

Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. 2011. Data-Driven Elastic Models for Cloth: Modeling and Measurement. *ACM Trans. Graph. (SIGGRAPH)* 30, 4, Article 71 (July 2011), 9 pages.

Hanchen Wang, Yuanqi Du Tianfan Fu, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, Anima Anandkumar, Karianne Bergen, Carla P. Gomes, Shirley Ho, Pushmeet Kohli, Joan Lasenby, Jure Leskovec, Tie-Yan Liu, Arjun Manrai, Debora Marks, Bharath Ramsundar, Le Song, Jimeng Sun, Jian Tang, Petar Veličković, Max Welling, Linfeng Zhang, Connor W. Coley, Yoshua Bengio, and Marinka Zitnik. 2023. Scientific Discovery in the Age of Artificial Intelligence. *Nature* 620 (2023), 47–60.

Huamin Wang and Yin Yang. 2016. Descent Methods for Elastic Body Simulation on the GPU. *ACM Trans. Graph. (SIGGRAPH Asia)* 35, 6, Article 212 (Nov. 2016), 10 pages.

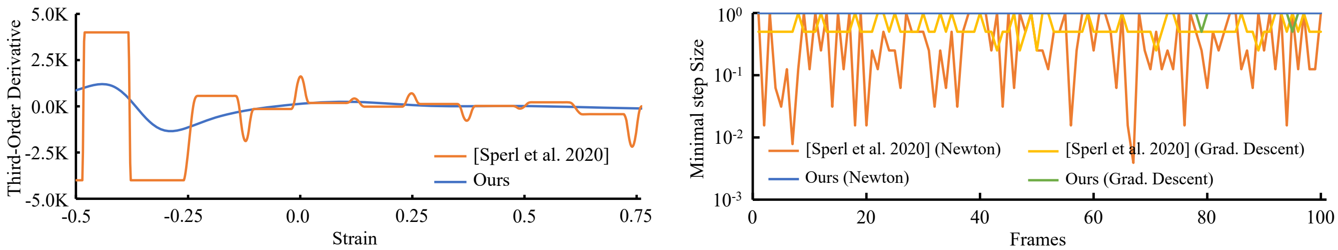
Jia-Ji Wang, Chen Wang, Jian-Sheng Fan, and YL Mo. 2022. A Deep Learning Framework for Constitutive Modeling Based on Temporal Convolutional Network. *J. Comput. Phys.* 449 (2022), 110784.

Haomiao Wu and Theodore Kim. 2023. An Eigenanalysis of Angle-Based Deformation Energies. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 6, 3 (2023), 1–19.

Kailai Xu, Daniel Z Huang, and Eric Darve. 2021. Learning Constitutive Relations using Symmetric Positive Definite Neural Networks. *J. Comput. Phys.* 428 (2021), 110072.

Shan Yang, Junbang Liang, and Ming C. Lin. 2017. Learning-Based Cloth Material Recovery from Video. In *IEEE International Conference on Computer Vision*.

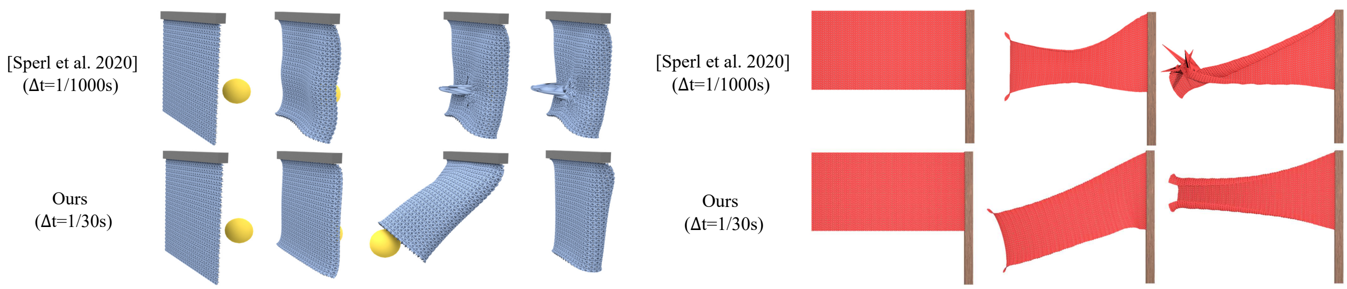
Zhan Zhang, Christopher Brandt, Jean Jouve, Yue Wang, Tian Chen, Mark Pauly, and Julian Panetta. 2023. Computational Design of Flexible Planar Microstructures. *ACM Trans. Graph.* 42, 6, Article 185 (dec 2023), 16 pages. https://doi.org/10.1145/3618396



(a) The third-order derivatives of strain energy density functions

(b) The minimal step sizes allowed during the simulation of each frame

Figure 5: Stability analysis based on third-order derivatives and minimal step sizes. As depicted in (a), our model exhibits smoother and smaller third-order derivatives compared to the HYLC model referenced in [Sperl et al. 2020]. This indicates that our model is suitable for stable simulations and can accommodate greater minimal step sizes in both Newton's method and the preconditioned gradient descent method as shown in (b).



(a) A 50cm×50cm fabric sample hit by a sphere

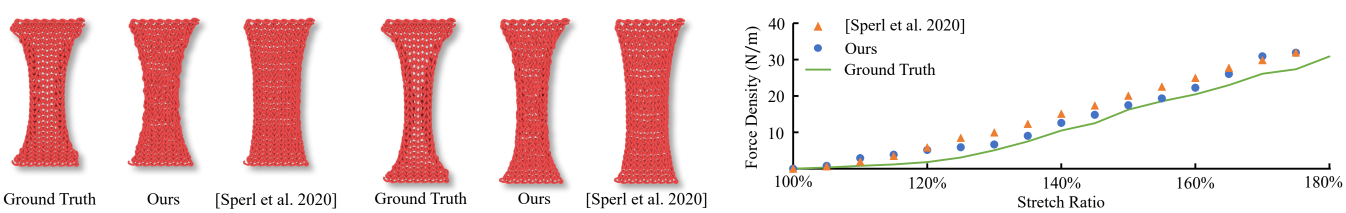
(b) A 20cm×50cm flag under strong wind

Figure 6: Animation examples simulated with our model and the HYLC model, as referenced in [Sperl et al. 2020]. While the simulations with our model run stably at $\Delta t = 1/30$s in both examples, the simulations with the HYLC model fails even when $\Delta t = 1/1000$s.

(a) A dancing avatar wearing a dress

(b) A Karate avatar wearing a T-shirt

Figure 7: Knitted garments simulated with our model on rapidly moving avatars. Thanks to the stability of our model, the continuum-based simulator can robustly simulate these examples at $\Delta t = 1/30$s, without backtracking line search. (The step size is fixed at one.)



(a) Stretch Ratio 130%

(b) Stretch Ratio 150%

(c) The relation between force density and stretch ratio

Figure 8: A uni-axial stretching experiment involving a stockinette fabric strip. Our model demonstrated its accuracy in continuum-based simulation when compared with the ground truth. This accuracy is evident both qualitatively, as seen in the Poisson and curling effects in the middle of the strip depicted in (a) and (b), and quantitatively, as illustrated by the correlation between force density and stretch ratio shown in (c). The ground truth is simulated using a yarn-level simulator. The results of our method and [Sperl et al. 2020] are obtained using a continuum-based simulator but are rendered with yarn mapping for enhanced visual effects.