UNIVERSITY OF LEEDS

An Improvement on Physics-based Cloth Simulation Realism.

Deshan Gong

Submitted in accordance with the requirements for the degree of PhD. Computer Science

The University of Leeds

Faculty of Engineering

School of Computing

September 2023

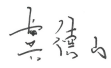
Abstract

Nowadays, cloth simulation is widely applied in 3D entertainment and fashion design. It also has been popular in computer graphics for decades and various simulation methods have been proposed. In both applications and research, cloth simulation realism is an important concern. Among the existing cloth simulation methods, physics-based cloth simulation methods integrate physical laws to explicitly model cloths' physics and govern cloths' mechanical behaviors. Compared with the other existing methods, they can simulate visually more pleasing and mechanically more accurate cloths. Moreover, they are capable of synthesizing various cloths made from different materials by using different cloth physical parameters. However, a physics-based cloth simulator's simulation realism may be restricted by its coarse cloth physical model and inaccurate cloth physical parameters. To pursue higher simulation realism, this research introduces more fine-grained cloth physical models and accurate cloth physical parameter estimation methods. In our first and second works, we propose two differentiable cloth simulators with more fine-grained cloth physical models that can capture cloth yarn-level mechanical behaviors and model cloth material heterogeneity. By using our novel fully-differentiable physical models, they can leverage gradient-based optimization methods to accurately estimate cloth physical parameters and then reproduce the observed cloths. In our third work, we introduce a cloth simulator that can simulate cloth time-dependent persistent wrinkles by integrating our novel time-dependent friction and plastic model. Through exhaustive experiments, we demonstrate our differentiable cloth simulators are accurate in estimating cloth physical parameters and reproducing the observed cloths. We also demonstrate our third simulator can plausibly simulate cloth time-dependent persistent wrinkles. Through these three works, we gain a significant improvement in cloth simulation realism.

Intellectual Property

The candidate confirms that the work submitted is his/her own and that appropriate credit has been given where reference has been made to the work of others.

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

Signed 

Acknowledgements

This research has been carried out by a team which has included:

- Prof. Andy Bulpitt
- Prof. He Wang
- Prof. Ningtao Mao
- Prof. Tianjia Shao
- Prof. Yin Yang
- Prof. Zhanxing Zhu

My own contributions include the methodologies, code implementations, experiments, and writing. The other members of the group and their contributions have been as follows:

- Prof. He Wang, as my main supervisor, guided my research during my entire Ph.D. study.
- Prof. Zhanxing Zhu gave me suggestions for the writing and experiments of the first work.
- Prof. Andy Bulpitt proofread the first work and gave suggestions for the writing and video.
- Prof. Ningtao Mao provided me access to his textile lab for the Cusick drape testing, guided me using Cusick drape meter, offered same fabric samples, gave suggestions for the tested fabric selection the second work.
- Prof. Yin Yang and Prof. Tianjia Shao provided suggestions for the writing and the needed experiments of the third work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Narrowing the real-virtual gap is a persistent goal that stimulates computer graphics research to pursue a more realistic simulation (Marschner and Shirley 2015). Cloths are so indispensable in our daily lives that can be seen in various scenarios for different usages, e.g., garments, furniture, buildings. Likewise, in animations and games, cloths are also ubiquitous and their simulation fidelity can usually determine the overall simulation plausibility. However, simulating cloth is challenging because real cloths have complex physical properties, such as nonlinearity and heterogeneity, and tend to exhibit more complicated mechanical behaviors, e.g., folds and wrinkles. Thanks to the admirable efforts that have been made on cloth simulation, modern cloth simulators can produce visually pleasing virtual cloths and, nowadays, are widely applied in 3D entertainment and fashion design (Stuyck 2022). The development process of cloth simulation during the last three decades essentially witnesses the history of improving the simulation realism. Along this way, various simulation methods have been proposed.

Conventionally, the cloth simulation methods can be classified into three groups: geometry-based method, physics-based method, and data-driven method (Choi and Ko 2005). As the simplest approach, the geometry-based method is the first one adopted by computer graphics to do cloth simulation. It shapes cloth static geometries or drives its dynamic motions according to pre-defined rules which are usually specified by mathematical equations (Weil 1986). Despite its simplicity, designing the rules to achieve a realistic simulation is difficult. Conversely, due to the lack of physics, the cloths simulated by this method usually look unnatural and counterintuitive. To address this shortcoming, the physics-based method was introduced and it substitutes physical laws for the pre-defined rules. Not only does this method avoid labori-

ously defining the rules, but it also shows a superiority in the simulation realism and is good at handling extremely complex simulation scenarios, e.g., cloths in intensive collisions. Since the introduction of the first physics-based cloth simulator, this method has been dominating in the applications requiring high-fidelity cloth simulations. Nevertheless, to further improve the simulation realism, more fine-grained physical laws usually need to be integrated, which inevitably complicates and slows down simulators. Some research leverages the CPU and GPU parallel computing to mitigate this problem (Tang et al. 2013; Wang 2021). As another way to speed up the simulation, the data-driven method simulates cloths by replaying cloth motions that are learned from the given training data. Without solving the physical equations at the run time, the data-driven cloth simulators are intrinsically faster and can even achieve interactive simulation speed. Thus, they are more feasible in efficiency-demanding applications like virtual try-on (Santesteban et al. 2019; Vidaurre et al. 2020). Nevertheless, compared with the physics-based method, it sacrifices flexibility and generalization ability. For instance, the simulation fidelity tends to dramatically drop when the simulation settings are out of the training data (Wang 2021). Hence, each of these three methods has its advantages and disadvantages, and which method is more feasible should be determined by the simulation requirements and goals. When the priority is positioned on the simulation realism, it is widely acknowledged that the physics-based method is the best (Volino and Magnenat-Thalmann 2000; Stuyck 2022). Admittedly, the state-of-the-art physics-based cloth simulators can produce impressive simulation results, but there is still room for further improvement due to coarse cloth physical models and inaccurate cloth physical parameters.

1.1 Motivations

To a physics-based cloth simulator, two of the major factors that jointly determine the simulation realism are: *the cloth physical model* and *the simulation parameters*. The cloth physical model defines the relationship between the cloth deformation and response force. For a physics-based cloth simulator, the integrated cloth physical model decides the cloth's mechanical properties can be embedded and the cloth behaviors can be simulated. A coarse cloth physical model usually overlooks many mechanical properties and behaviors that exist in real cloths and finally makes the simulated cloth look unnatural. Real cloths usually exhibit complex mechanical behaviors and properties due to their mechanical structures at the micro-level. For

example, woven and knitted cloths are common in our daily lives and, at the micro-level, they are made from interwoven yarns each of which is a bundle of fibers (or filaments). Due to this structure, cloth deformation essentially consists of two parts: the relative motion between yarns (or fibers) and the deformation of yarns (Lin et al. 2012). Thus, there are two kinds of internal forces which respectively derive from the relative motion between yarns and the deformation of yarns. Either kind of force is indispensable to cloth mechanical behaviors. For instance, yarns resist elongation when being stretched such that a piece of cloth can keep its rest length when being used. In addition, cloths' internal friction prohibits the relative sliding motion between the contacting yarns. On the one hand, the friction between the contact yarns prevents a cloth from unraveling. On the other hand, it is one of the important reasons for cloth hysteresis and persistent deformations (Ngo and Boivin 2004). Omitting the internal friction will miss these cloth mechanical behaviors in the simulated cloth and restrict the simulation realism. Conversely, modeling cloth micro-mechanics enables cloth simulators to simulate the cloth mechanical behaviors at the yarn-level and to produce high-fidelity simulations (Cirio et al. 2014; Cirio et al. 2015; Cirio et al. 2017). In addition, a cloth's physical properties are jointly decided by every yarn and the interactions between yarns. For instance, real cloth physical properties exhibit random variations across its geometry because a cloth micro-level (yarn-level or fiber-level) structure may randomly change in production and use. Consequently, two same-size cloths that are made from the same material are likely to behave differently even in the same motion because cloth physical properties usually vary randomly across its geometry. Thus, real cloths are more like heterogeneous materials rather than homogeneous materials (Wang et al. 2008). Unfortunately, even in state-of-the-art cloth simulators, cloth micro-level structure and stochastic physical properties are usually overlooked. Consequently, the coarse cloth physical models restrict the simulation realism.

Apart from the cloth physical model, cloth physical parameters also determine the simulation realism. To simulate a specific kind of cloth in a physics-based cloth simulator, the conventional way requires numerous times of trial-and-errors to tune the cloth's physical parameters, which is prohibitively time- and labor-consuming. To avoid this, many studies leverage the numerical optimization and machine (or deep) learning methods to estimate these parameters (Wang et al. 2011; Ju and Choi 2020). However, the former requires a very long optimization time and the latter relies on a huge amount of training data. Recently, differentiable physics has been thriving in objects' physical parameter estimation due to its outstanding convergent speed, sample

efficiency, and parameter estimation accuracy (Belbute-Peres et al. 2018; Hu et al. 2020; Hu et al. 2019). Its application in cloth physical parameter estimation also exhibits a superior performance than the other methods (Liang et al. 2019; Li et al. 2022a). However, cloth physical models and simulation parameters jointly determine the simulation realism because the underlying cloth physical models of differentiable cloth simulators decide what cloth physical parameters can be estimated. To our best knowledge, the available differentiable cloth simulators are built on coarse sheet-level cloth physical models which ignore cloths' micro-mechanical behaviors and heterogeneous material properties. A differentiable cloth simulator built on a coarse cloth physical model can hardly reproduce an observed cloth no matter how it tries to optimize the cloth physical parameters. Consequently, the simulation realism is still restricted by the underlying coarse cloth physical models. Thus, to improve cloth simulation realism, using more fine-grained cloth physical models to embed real cloth mechanical properties and accurately measuring cloth physical parameters must be carried out at the same time.

This research project aims at improving physics-based cloth simulation realism by modelling more fine-scaled physics in real cloths and more accurately estimate cloth physical parameters. To achieve this objective, we propose two differentiable cloth simulators based on more fine-grained differentiable cloth physical models for accurate parameter estimation and a physics-based cloth simulator for realistically simulating cloth persistent wrinkles due to cloth micro-mechanics.

1.2 Contributions

The first work introduces a fine-grained differentiable cloth simulator for estimating cloth physical parameters. By modeling cloths as interlaced yarns, the underlying cloth physical model can capture the cloth micro-mechanical interactions between yarns, easily encode different woven patterns, and simulate blend woven by mixing different yarns. The simulator is equipped with our novel differentiable physical models to simulate various yarn-level internal forces such that it can conduct gradient-based optimization to estimate cloth physical parameters. Compared with a previous differentiable cloth simulator which models cloths as continuum sheets (Liang et al. 2019), our fine-grained differentiable cloth simulator can more accurately fit the observed cloth dynamics and the estimated cloth physical parameters are more explainable.

The second work introduces a Bayesian differentiable cloth simulator for estimating cloth phys-

ical parameters from the images captured by a textile standard testing method: the Cusick drape testing (Cusick 1961). The machine learning methods tend to adopt more flexible ways to estimate cloth physical properties in a casual experiment setting such that can avoid using expensive apparatuses and rigid standards (Bhat et al. 2003; Yang et al. 2017). Even if these methods can work decently, they sacrifice the parameter estimation accuracy and the inaccurate parameters would in turn degrade the simulation realism. By contrast, the Cusick testing is required to be conducted in a rigidly controlled environment while following the widely acknowledged standards in the textile community. In this way, it can eliminate the noise that may affect parameter estimation accuracy. To this end, this work proposes a new Cusick drape dataset. In addition, the cloths cut from the same fabric usually show randomly varied drape shapes because of the random variation of cloth physical properties across its geometry at the micro-level. To model this randomness, this work proposes a Bayesian differentiable cloth simulator that models cloths as stochastic heterogeneous materials. Instead of using a uniform physical parameter globally, it randomly samples cloth's local physical properties from probability distributions. It is also fully differentiable such that can adopt gradient-based optimization methods to estimate cloth physical parameters. Through experiments, we demonstrate that: (1) compared with the homogeneous differentiable cloth simulator, our Bayesian differentiable cloth simulator can more closely fit the observations; (2) compared with the deterministic model, our Bayesian differentiable cloth simulator model can account for and learn the randomness observed in the Cusick testing.

The third work proposes a physics-based cloth simulator that is integrated with fine-grained cloth physical models for simulating cloth with persistent wrinkles. Persistent wrinkles commonly appear on clothes (e.g., the knee area on the trousers). Compared with cloth overall dynamics, the wrinkles are secondary cloth mechanical behaviors. The formation of persistent wrinkles is an important feature that makes cloth distinctive from elastic membranes. It is a deformation history-dependent behavior that is determined by the deformations that a cloth has experienced. Moreover, these wrinkles on garments can imply the habits of the person who wears them and greatly affect the simulation realism (Bridson et al. 2003). At the micro-level, the persistent wrinkles are a joint effect of the cloth yarn relative sliding motions and yarn persistent deformation (Prevorsek et al. 1975). However, the cloth simulators in computer graphics have only used either internal friction or plastic deformation to simulate the persistent wrinkles, but have never combined them together (Narain et al. 2013; Miguel et al. 2013; Wong et al.

2013). In addition, it has been observed that cloth persistent wrinkles usually exhibit a time-dependence: the longer the deformations that cause the persistent wrinkles are kept, the more obvious the wrinkles would look (Levison et al. 1962). However, to our best knowledge, very few cloth simulators in computer graphics can simulate this phenomenon. To gain a more realistic simulation, we propose a novel time-dependent friction model and plastic model where the former is responsible for simulating the wrinkles caused by the yarn relative sliding and the latter is designated to simulate yarn permanent deformations. Through experiments, we demonstrate that both our friction model and plastic model can simulate time-dependent wrinkles. Meanwhile, by combining them, our cloth simulator can naturally simulate the persistent wrinkles caused by different deformations. In addition, compared with the previous friction model and plastic model, only our model can additionally simulate the time-dependent wrinkles.

In short, our research makes contributions to improving cloth simulation realism by introducing more fine-grained cloth physical models and more accurate physical parameters estimation methods in three sub-works:

- A fully differentiable yarn-level cloth simulator that models cloth micro-level inter and intra-yarn mechanics, and can estimate cloth physical parameters to accurately reproduce the observed cloth dynamics.
- A Bayesian differentiable cloth simulator for learning cloth stochastic and heterogeneous physical parameters from images that are captured by a strictly controlled textile testing protocol: the Cusick Drape testing.
- A novel time-dependent friction model and plastic model that can naturally simulate cloth persistent wrinkles which are affected by deformation duration.

1.3 Overview

This writing showcases our works with the following arrangement.

- **Chapter 2** gives an inclusive literature review about cloth simulation and cloth simulation parameter estimation. It provides a relevant background and introduces the problems that are studied in these fields.
- **Chapter 3** introduces our fine-grained yarn-level differentiable cloth simulator and demon-

strates the improvement in simulation realism by estimating cloth simulation parameters with modeling cloth micro-mechanics.

- **Chapter 4** shows our Bayesian differentiable cloth simulator for estimating cloth physical parameter probability distributions with a textile standard testing protocol: the Cusick drape testing. The results demonstrate that modeling cloth material heterogeneity and stochasticity improves the differentiable cloth simulator's ability to learn real cloth random mechanical behaviors and, in turn, increases the simulation realism.

- **Chapter 5** presents our cloth persistent wrinkles simulation method which is realized by using our time-dependent friction and plastic model. This work shows that, on the one hand, combining the cloth internal friction and material plasticity makes the simulated wrinkles look more natural. On the other hand, including time-dependence makes the formation of wrinkles follow our daily observations.

- **Chapter 6** summarizes the thesis with a discussion about the works' limitations and the plan for future research.

In addition, since cloth is employed as an application in this thesis, we use the terms "cloth" and "fabric" interchangeably.

Chapter 2

Related Work

This chapter starts with Section 2.1 for reviewing cloth simulation methods in computer graphics which include geometry-based methods, physics-based methods, and data-driven methods. With a broad review of the development of cloth simulation, this section will illustrate the path along which people keep pursuing a higher simulation realism and highlight the advantages of physics-based methods in high-fidelity cloth simulation. By the end of this section, it will be clear that cloth micro-mechanics is important to cloth mechanical properties and behaviors, and realistic cloth simulation. Apart from cloth physical models, cloth physical parameters also determine the simulation realism of physics-based cloth simulation methods. The following section summarizes the works about cloth parameter estimation, and compares the advantages and disadvantages of different estimation methods. It will highlight the advantages of differentiable cloth simulation in learning accuracy, sample efficiency, and convergent speed.

2.1 Cloth Simulation

The cloth simulation methods can be broadly grouped into three categories: geometry-based method, physics-based method, and data-driven method (Ng and Grimsdale 1996; Choi and Ko 2005; Hu et al. 2009; Santesteban et al. 2019). The geometry-based method is the earliest approach adopted by computer graphics to do cloth simulation. It is straightforward and relies on hard-coded rules to determine cloth geometry, so the simulated realism is almost fully decided by the rationality of the rules. It is almost impossible to handle various simulation scenarios with one group of rigid rules, so its flexibility is low. In addition, as this method embeds no physics,

it may simulate the cloths that are impossible in the real world. To alleviate this problem, the first physics-based cloth simulator was introduced sooner after that (Feynman 1986). By integrating physics laws, it achieves a higher simulation realism and avoids the tedious labor work of defining the rules. However, a realistic physics-based cloth simulator usually needs to integrate complex physical laws and use more memory to encode geometrical information, e.g., using a cloth mesh with a higher resolution. This inevitably slows down the simulation speed, increases memory consumption, and makes the physics-based method infeasible in speed-demanding applications, e.g., real-time simulation and virtual try-on. The data-driven method provides a solution to this problem by replaying the cloth motions learned from data. In this way, it boosts the simulation efficiency to a very high level with sacrificing some flexibility, i.e., generalize poorly to the simulation scenarios out of train data. All of these three methods have unique advantages and shortcomings, and are playing irreplaceable roles in different application scenarios. Section 2.1.1, Section 2.1.2, and Section 2.1.3 will summarize the academic works that study these methods, respectively. Moreover, we refer the reader to the review papers (Ng and Grimsdale 1996; Choi and Ko 2005; Volino et al. 2005; Nealen et al. 2006; Thomaszewski et al. 2007; Hu et al. 2009) and books (Volino and Magnenat-Thalmann 2000; Stuyck 2022) for more in-depth introductions of the cloth simulation methods and their implementations.

2.1.1 Geometry-based Method

Geometry-based methods use manually defined rules to determine cloth static geometries or dynamic motions. Weil (1986) proposes a geometry-based cloth simulator (which is usually taken as the first cloth simulator in graphics) for simulating the static shape of a piece of rectangular cloth hung by constraint points. It models a cloth as a 2D grid with 3D geometry and introduces a two-stage simulation strategy (an approximation stage followed by a relaxation stage) to generate the cloth's surface. In the approximation stage, it approximates the cloth surface by the curves that are defined by all the pairs of the constraint points. It proposes a catenary equation for defining a curve's geometry. In the relaxation stage, it searches for a compromise cloth geometry that can satisfy all the curves by using iterative optimization. This simulator can reflect the simulated cloth's material by adjusting the coefficients of the catenary equation. However, those coefficients encode no physical information and adjusting them is equivalent to directly manipulating the simulated cloth's shape. As a result, it is usually difficult to tune the coefficients to simulate an expected cloth. Additionally, only simulating a

hung cloth is not useful in practice and, as a geometry-based method, it lacks flexibility and can only handle this single simulation scenario. By contrast, Hinds and McCartney (1990) develop a more practical cloth simulator for garment design which can simulate a garment that fits the top part of a mannequin. It can assist tailors in designing the sewing patterns before making a physical garment. This simulator adopts the real garment making routine and models a garment by sewing multiple cloth panels at their edges. Given the position of the panels' edges, the simulator positions the panels around the mannequin by using an interpolation method that can fit the garment to the geometry of the mannequin and also ensure a natural smoothness on the sewing edges. In addition, to simulate a garment's drapes and folds, the application adopts a harmonic function to generate a fine-scaled geometrical offset which is superimposed on the garment. However, for simplicity, the garments simulated by this simulator do not include sleeves. By contrast, Ng (1995) focus on simulating the folds on the cylindrical cloths that cover a cylindrical object, e.g., a sleeve on an arm. It assumes that the cloth and the underlying flesh are represented by the meshes with the same topology such that their vertices can be mapped one by one. It also assumes that the folds tend to appear on the slack area of cloth which are selected by measuring the distance between the mapped vertices. To simulate fine-scaled geometries on the sleeve, it uses the quadratic *sinc* function and parabola function to model folds.

In general, geometry-based methods are simple, but they lack the flexibility for handling various simulation scenarios and requires sophisticated designing experience to define the rules. Since the introduction of physics-based methods and data-driven methods, they gradually play a secondary role in modern cloth simulators. For instance, geometry-based methods are more commonly used in combination with these methods for simulating fine-scaled geometries. Kang et al. (2001) use geometry-based simulation to add wrinkles on the cloth simulated by a low-resolution mass-spring physical model. Concretely, when the two adjacent mass points are closer than a threshold, the simulator uses *sin* function to generate waves which are superimposed onto the cloth for introducing fine-scaled wrinkles on the coarse mesh. Apart from simulating wrinkles, the geometry-based method is also used to synthesis a cloth's yarn-level geometry. Admittedly, simulating the cloth mechanical behaviors at the yarn-level purely by physics-based methods can bring about a higher simulation realism, but it is also computationally expensive. Instead, Sperl et al. (2021) use the geometry-based method to add yarn-level geometrical details on a mesh simulated by the sheet-level physics-based cloth simulator. It first allocates the yarn-

level geometry to the triangles in the mesh when the cloth is in its rest shape. When the simulation starts, the cloth dynamics are governed by a sheet-level cloth simulator. The yarn-level geometries are updated according to the mesh deformation during the simulation. In this way, this simulator is as efficient as a coarse sheet-level cloth simulator and, meanwhile, can generate fine-scaled yarn-level geometrical details.

2.1.2 Physics-based Method

Physics-based cloth simulators use physical laws to model cloth mechanical behaviors. Owing to this, they are flexible and can handle diverse simulation scenarios. In addition, through modeling cloth physics, they can naturally simulate various cloths made from different materials by adjusting cloth physical parameters. Furthermore, the simulated cloths can maintain physical constancy and correctness no matter how complex the involved mechanical behaviors are. These advantages collectively determine their dominating role in modern cloth simulators for producing realistic simulations. In a physics-based cloth simulator, the underlying cloth physical model determines the physical laws that can be integrated, and the physical laws decide the mechanical properties and behaviors that can be embedded in the simulated cloth. Therefore, the cloth physical model and the integrated physical laws have a great influence on simulation realism. Since the introduction of the first physics-based cloth simulator in computer graphics, various cloth physical models and laws have been proposed to pursue a higher simulation realism by more accurately simulating cloth mechanical behaviors. Among them, yarn-level cloth simulators model a cloth as interlaced yarns and can achieve the state-of-the-art simulation realism in computer graphics. They can naturally integrate various physical laws to realistically simulate real cloth mechanical properties due to cloth yarn-level mechanics (e.g., nonlinear elasticity, plasticity).

Cloth Physical Models

Generally, according to the cloth physical model, physics-based cloth simulation methods can be further classified into *mass-spring methods*, *particle-based methods*, and *shell (or plate) methods*. The physical models determine what physical laws can be integrated to simulate cloth mechanical behaviors. *Mass-spring methods* model a piece of cloth as the material points which are connected by massless springs where the material points' spatial positions collectively define cloth geometry. The topology of the spring connections is determined by the embedded cloth

internal forces. Hooke's law is commonly adopted to model elastic cloths where the internal forces derive from the elongation or compression of the springs and tend to keep cloths in their rest shapes, i.e., all springs are in their rest length. A cloth's physical properties are defined by the material points' weight and springs' stiffness parameters (Provot 1995; Choi and Ko 2002; Liu et al. 2013). This modeling method is easy to implement and enables fast simulation thanks to the simple physics, but springs are too coarse to integrate real cloths' physical properties for accurately simulating cloth mechanical behaviors (e.g., nonlinearity, viscosity). In addition, it is difficult to arrange springs to model real cloth mechanics. By contrast, the *particle-based methods* model cloth as a collection of particles which define the spatial positions of the cloth cross over between yarns. A cross over refers to the intersection between two contact yarns. To model cloth internal forces, they define various potential energy functions of adjacent particles' position and velocity where they can embed cloth nonlinear elastic material properties (Breen et al. 1992; Breen et al. 1994a; Breen et al. 1994b). However, the cross overs are dense in real cloth. As a result, compared with the mass-spring methods, simulating cloths by particle-based methods need more computational resources and is usually much slower. Mass-spring methods and particle-based methods are usually referred to as discrete methods as they both treat the simulated cloths as a collection of discrete elements, i.e., material points or particles. By contrast, *shell methods*, which derives from continuum mechanics, model the simulated cloths as a continuum objects such that the mass and internal energy of a piece of simulated cloth are evenly distributed across its geometry. In continuum mechanics, an object's mechanical behaviors are defined by constitutive laws which specify the strain-stress, i.e., average deformation and force, relationship (Chaves 2013). To conduct simulation on computers, these methods needs to do spatial discretization (by finite element or finite difference method) such that the continuous geometry can be approximated by many smaller simple elements. Compared with *mass-spring methods* and *particle-based methods*, they can adopt various sophisticated constitutive laws to accurate model the mechanical behaviors observed in real cloths, e.g., nonlinearity, viscosity, and plasticity (Volino and Magnenat-Thalmann 2000), and can achieve a superior simulation realism. However, they are more complicated to implement because strain and stress have more complex definitions. In addition, the extra spatial discretization needs to careful implementations.

Apart from these three conventional methods, some recently proposed cloth physical simulators model simulate at the yarn-level. They model a piece of simulated cloth as interlaced inex-

tensible or elastic splines for representing woven or knitted cloth's yarns. In this way, they can explicitly simulate the mechanical behaviors of each individual yarn (Kaldor et al. 2008; Cirio et al. 2014; Cirio et al. 2017). In the simulated cloths, the internal forces consist of the intra- and inter-yarn forces where the former are defined by the deformation of a spline and the latter are defined by the interactions between the splines. In this way, these cloth models can naturally simulate cloth micro-mechanical behaviors which decide cloth nonlinear or non-elastic mechanical properties, such as viscosity and hysteresis. The hysteresis refers to the energy consumption in the cloth deformation process commonly observed in real cloths and is usually attributed to cloth internal frictions (Wong et al. 2013). By contrast, although the particle-based methods (Breen et al. 1992; Breen et al. 1994a; Breen et al. 1994b) model cloths' cross overs, they do not model the frictional contact between the yarns and cannot simulate hysteresis. Therefore, yarn-level cloth simulators can achieve a far higher simulation realism than the others.

Cloth Physical Laws

In a physics-based cloth simulator, the integrated cloth physical laws determine the mechanical properties and behaviors that can be seen in the simulated cloth. As real cloths can exhibit complicated mechanical behaviors (e.g., nonlinear elastic, non-elastic, plasticity, hysteresis) other than simple elastic sheets, various cloth physical laws have proposed to reproduce real cloths' mechanical behaviors and pursue a higher simulation realism.

Linear Elastic Mechanics The early cloth simulators usually model cloths as linear elastic materials. Feynman (1986) proposes the first physics-based cloth simulator for simulating cloth static drapes. It adopts the shell method to model a simulated cloth as an elastic plane which is discretizated into a grid of points. In addition, it decouples a cloth's deformation into in-plane deformation (i.e., tensile or stretching) and out-of-plane deformation (i.e., bending). It defines a linear elastic stretching and bending energy to model the cloth's internal forces: stretching force and bending force, which tend to keep the simulated cloth in its rest shape. To simulate a cloth's static drape, this simulator searches for the cloth shape with minimum total energy which is the sum of the cloth's internal stretching and bending energy, and external gravity potential energy. However, it is unable to simulate cloth motions and has a very restricted utility. By contrast, to simulate cloth dynamics, Terzopoulos et al. (1987) adopt the Lagrange equation to build the

cloth motion equation which is an ordinary differentiable equation (ODE). To solve the ODE, it uses the numerical integration method which is equivalent to approximating the ODE by multiple linear equations (usually referred to as time discretization in cloth simulation). To model cloth physics, it takes a piece of cloth as a continuum elastic 2D surface and defines the surface's deformation energy density functions to model the cloth's internal forces. To do spatial discretization, it adopts the finite difference method to approximate the continuous surface by a grid of points. Whereas this simulator can only simulate simple cloths, like tablecloths or flags. In our daily lives, cloths also commonly appear as on-body garments. Simulating a garment that can move with the underlying body is a practical application for 3D animations and fashion designs. When making a real garment, a tailor needs to sew multiple cloth panels together. The garment simulator proposed by Carignan et al. (1992) mimics this real garment workflow in simulation. It also sews multiple virtual cloth panels together and each cloth panel's physics is governed by the cloth physical model proposed by Terzopoulos et al. (1987). Considering the numerous collisions between the body and the on-body garment, it adopts the conservation of momentum law to define the motions of the elements in the collision. In this way, it avoids the bouncing artifacts that would appear when using the conventional potential energy method (i.e., introducing a repulsive force to push elements in contact apart). Even if simply modeling cloths as linear elastic material can simulate visually pleasing results, real cloths usually exhibit nonlinear elastic or non-elastic mechanical behaviors. Since seminal studies introduced above, many works have been proposed to improve the cloth simulation realism by more accurately modeling real cloth mechanical behaviors.

Biphasic Mechanics by Strain Limiting When modeling cloths an elastic materials, the simulated cloths tend to exhibit the *super-elasticity* artifacts where the simulated cloths' stretching deformation concentrates on the areas that are under high stresses. For instance, a piece of simulated hung cloth would exhibit a local over-elongation around its two top corners for pinning the cloth. However, real cloths rarely have large local stretching deformations and, consequently, the *super-elasticity* artifacts make the simulated cloths look more like rubber (Provot 1995). A straightforward approach to this problem is using high-stiffness parameters. However, when adopting this method, the simulation time step size has to be reduced to guarantee the stability of numerical time integration, which will degrade the simulation efficiency (Hauth et al. 2003). By contrast, Provot (1995) introduces the first strain limiting method to address

the *super-elasticity* artifacts. Concretely, the work integrates the strain limiting method into a mass-spring cloth simulator to restrict the maximum local elongation and compression of the springs by hard-coded rules (only 10% elongation and compression are allowed). In this way, the simulator can effectively eliminate the *super-elasticity* artifacts and, instead, the stretching deformation can naturally propagate to the entire cloth without altering cloth stiffness parameters. This simple and effective strain limiting approach is widely adopted in the later works (Desbrun et al. 1999; Meyer et al. 2001). Essentially, it simply manipulates cloth geometry by post-processing. Therefore, before correcting the artifacts, this method first needs to allow the over-stretching to occur and then adjusts cloth geometry to enforce the restrictions. By contrast, Vassilev et al. (2001) enforce the restrictions by limiting cloth deformation variation rate (i.e., material points' velocities in a mass-spring model). In this way, the over-stretching can never occur in every simulation step. However, directly correcting cloth geometry may cause self-intersection artifacts and greatly degrade the simulation realism. To resolve this problem, Bridson et al. (2002) introduce a robust collision handling method that adopts the continuous collision detection (CCD), and combines the impulse method and geometry method to do collision response. This collision-handling method can cooperate with strain limiting to avoid the *super-elasticity* without suffering from self-intersection artifacts. Additionally, strain limiting method in their work also simultaneously enforces restrictions on the deformation and deformation rate.

The strain limiting methods introduced above are *local* strain limiting methods because the deformation restrictions are only enforced onto the elements that exceed the given threshold. In the implementation, the local restrictions are usually enforced in an iterative manner: first running forward simulation and then correcting the local deformations to enforce the restrictions. When correcting cloth deformations, the simulated cloth physics is ignored. Therefore, the simulated cloth will not be stable until all elements satisfy the restrictions after a forward simulation. Consequently, it may need prohibitively many iterations to converge and even fail to converge especially when simulating an inextensible cloth in high deformation scenarios, e.g., tightly stretching all the boundaries of a cloth. Instead, Goldenthal et al. (2007) adopt the augmented Lagrangian equation to model a cloth's constrained motion equations where the constraints embed the strain limits. As such, in every simulation step, the simulator can take into account the simulated cloth global physics and enforces the deformation restrictions at the same time. Consequently, it achieves a higher convergence performance than the iterative

methods. In addition, to accelerate the optimization speed of the augmented Lagrangian equation, it introduces a step and projection method that starts with a forward simulation without considering the restrictions to compute the cloth next state. (In a cloth simulator, a cloth's state refers to the cloth deformation and deformation rate or a cloth's mesh vertices' position and velocity after discretization.) In the next simulation step, it projects the cloth new state on the manifold defined by the deformation restrictions. Although this method is efficient, it tends to cause numerical damping artifacts because the projection operation will delete the normal component of cloth velocity and cause energy dissipation. To alleviate this problem, English and Bridson (2008) use the second-order backward Euler method (Choi and Ko 2002) to do numerical integration rather than the commonly used first-order backward Euler method (Baraff and Witkin 1998). In this way, more cloth dynamics information is used in simulation and therefore more energy can be kept after the projecting operation.

Due to the cloths' structure, they behave like anisotropic materials that are stiffer along the directions of the yarns, e.g., the perpendicular warp and weft directions in woven cloths, and are relatively softer along the other directions. Thus, an accurately simulated cloth should reflect this anisotropy. To embed this anisotropy by strain limiting, the works introduced above need to model a piece of cloth as a perpendicular mass-spring grid or quadrilateral mesh where the springs or mesh edges are along the same directions as the yarns. In this way, they can embed cloth anisotropy by limiting the springs or mesh edges elongation. Compared with shell methods, mass-spring methods are coarser due to the simpler physical models. In addition, arranging the springs to accurately model cloth internal forces is difficult. When adopting shell methods, quadrilateral meshes are infeasible when modeling cloth with non-straight boundaries or using adaptive meshes. To simulate anisotropic cloths without restricting cloth mesh topologies, Thomaszewski et al. (2009) introduce the Continuum-based Strain Limiting (CSL) method that can enforce deformation restrictions onto the principle components of the strain and strain rate tensors which represent cloth deformation and deformation rate along the warp and weft directions. In this way, it does not have to use quadrilateral mesh for embedding anisotropy and can be easily integrated into shell cloth simulators. In shell cloth simulators, the mesh resolution of a simulated cloth also determines the simulation realism. For instance, only high-resolution meshes can embed cloth fine-scaled geometrical details, e.g., wrinkles (Selle et al. 2008). However, when using high-resolution cloth meshes, the dimension of motion equation would be so high that could require a prohibitive longer time to converge with enforcing

deformation restrictions. To alleviate this problem, Wang et al. (2010b) propose a multiple-resolution approach for accelerating convergence when using high-resolution meshes. It realizes this by introducing a downsampling and upsampling function. Concretely, it first downsamples the original mesh resolution to a low-resolution coarse mesh and enforces the restrictions on the coarse mesh through an iterative method. Then, the corrected coarse mesh is gradually upsampled with enforcing the restrictions onto the finer mesh. This upsampling is repeated until the mesh returns to its original resolution. In this way, compared with directly enforcing restrictions on the high-resolution mesh, it achieves a faster convergence speed. In addition, Narain et al. (2012) observe that using an iterative manner to implement the strain limiting is infeasible when using irregular mesh that consists of faces in different size. For instance, manipulating a mesh face's vertices positions to enforce the deformation restrictions may overly change the geometry of the adjacent smaller faces, which makes convergence difficult. Therefore, to cooperate with their adaptive remeshing method for cloth simulation, they adopt the Augmented Lagrangian method to enforce deformation restrictions (Nocedal and Wright 2006) because it considers a cloth's global geometry in optimization.

Nonlinear Elastic Mechanics Essentially, cloths' inextensibility is a natural result of the cloth nonlinear mechanical property where a piece of cloth is usually soft in small deformation and becomes dramatically stiffer as the deformation increases. The strain limiting method models cloth as a biphasic material that becomes infinitely stiff when the deformation exceeds the threshold and this essentially is a simple nonlinear mechanical model. However, it has been observed that the nonlinearity exhibited in cloth mechanical behaviors is far more complex than a biphasic material. In general, a cloth's force-deformation process can be roughly separated into three regions: a region of initial resistance to deformation, a region of low deformation, and a region of high deformation (Breen et al. 1994b). In the region of low deformation, cloths' mechanical behaviors are close to a linear elastic material. In the other two regions, the nonlinearity becomes obvious. Therefore, simply modeling cloth as an elastic material or biphasic material usually does not cause obvious artifacts when only simulating limited kinds of cloth within the small deformations. However, this coarse modeling method cannot guarantee simulation realism when simulating a cloth that is under large deformation or simulating cloths that can exhibit more complex nonlinear mechanical behaviors.

To pursue a higher simulation realism, cloth simulators introduce mechanically more accurate

physical laws to embed cloth nonlinear mechanical properties. Three well-known methods to model cloth nonlinear mechanics are using piecewise linear model, introducing cloth internal friction, and using hyper-elastic constitutive laws. The piecewise linear model is a numerical fitting method that approximates a complicated nonlinear force-deformation curve by multiple linear (or simple nonlinear) functions. For instance, Wang et al. (2011) use a piecewise linear model to approximate cloth nonlinear (and anisotropic) stretching and bending mechanical behaviors. The model consists of 24 stretching stiffness parameters and 15 bending stiffness parameters which split the cloth nonlinear stretching and bending load-deformation curves into multiple linear pieces. To reproduce real cloth mechanical behaviors, it adopts numerical optimization to estimate these parameters by minimizing the geometrical difference between observed real cloths and simulations. In the simulation stage, every cloth mesh element's stretching and bending stiffness parameters can vary and are interpolated according to the element's deformation such that the simulated cloth can exhibit real cloth nonlinear mechanical behaviors. By contrast, to reproduce real cloth static drape, Breen et al. (1994b) use piecewise linear and quadratic functions to fit the nonlinear force-deformation curves of real cloth measured by the Kawabata Evaluation System (Kawabata 1980). In this way, it can realistically simulate cloth static drape shapes which can reflect the visually distinctive features that can indicate the cloth material. Eberhardt et al. (1996) further improve this work by allowing cloth dynamics simulation, introducing air resistance, and additionally considering the hysteresis effect.

In the textile, the hysteresis effect is commonly attributed to cloth internal friction which refers to the contact friction between yarns and fibers. Cloth internal friction is also a reason for the cloth nonlinear mechanical properties. For example, Ngo and Boivin (2004) integrates an enhanced version of Dahl's friction model (additionally embeds the Stribeck effect) to their cloth simulator to simulate cloth nonlinear stretching, shearing, and bending behaviors. By using numerical optimization, their simulator can estimate the cloth physical parameters to fit the force-deformation curves measured in the Kawabata Evaluation System (KES-F). With the estimated parameters, it accurately reflects the nonlinear properties of the real cloth in simulation. In KES-F force-deformation curves, a cloth's reaction force usually dramatically increases as deformation becomes large. One plausible reason for this phenomenon is that, as a cloth gradually deforms, the inter-yarn friction forces increase because the normal contact forces between the yarns increase. However, Dahl's friction model adopted by Ngo and Boivin (2004) cannot simulate this phenomenon. Miguel et al. (2013) introduce a strain-dependent

Dahl's friction model where the friction force is a function of strain such that the friction force strength can increase with deformation. After integrating this friction model, their simulator can more accurately simulate cloth non-linear mechanical behaviors.

In continuum mechanics, nonlinear mechanics has been studied for a long time and there are various hyper-elastic constitutive laws to account for the nonlinear mechanical behaviors observed in different real-world materials. Eischen et al. (1996) use the nonlinear shell theory (Libai and Simmonds 1983) to simulate cloth which is accurate, but is computationally too expensive. The Saint-Venant-Kirchhoff (StVK) model is one of the constitutive laws for modeling nonlinear elastic materials. To embed cloth mechanical nonlinearity with acceptable simulation efficiency, Volino et al. (2009) introduce a simplified (anisotropic) StVK model to a shell cloth simulator. Through experiments, this work demonstrates that this model can simulate a force-displacement curve similar to the real measurements. Moreover, this model's simulation accuracy is further demonstrated in a later work (Miguel et al. 2012) that uses the StVK model to fit the force-displacement curves measured on real fabrics. In addition, Clyde et al. (2017) adopt the Kirchhoff-Love thin shell model from the continuum mechanics to simulate cloth physics and introduces a hyper-elastic constitutive law, which is similar to the Ogden model (Ogden and Hill 1972), to simulate cloth nonlinear elastic mechanical behaviors.

Non-elastic Mechanics Apart from the nonlinear elasticity, real cloths also usually exhibit non-elastic mechanical behaviors. For example, cloths have history-dependent mechanical behaviors where a cloth cannot return to its original rest shape after experiencing an extremely large deformation, such as permanent elongations and persistent wrinkles. This mechanical characteristic is indispensable to real cloth, and must to considered in cloth production and fashion design. For instance, on the one hand, it can be leveraged to shape cloths for design; on the other hand, this property is usually undesired in the high quality and durable cloths. In material engineering, this irreversible deformations are usually referred to as plastic deformations and this material property is usually referred to as plasticity. In textile, cloth plasticity is taken as the combinational effect of cloth internal friction and the persistent deformations of yarns and fibers (Prevorsek et al. 1975). To model cloth plasticity, a cloth's deformation should be decoupled into an elastic part and plastic part. Cloth (either linear or nonlinear) elastic mechanical behavior is only determined by the elastic deformation. The plastic deformation varies only when the elastic deformation exceeds a predefined threshold, usually referred to as

yield strain. In computer graphics, O'Brien et al. (2002) adopt the perfect plastic model to simulate plastic deformation which assumes that the deformation exceeding the yield strain is immediately converted to plastic deformation. However, the perfect plasticity is uncommon in real materials. In practice, plastic deformation usually involves hardening (or aging) effects where, after experiencing plastic deformations, the material becomes difficult to have more plastic deformations. Therefore, Grinspun (2008) adopts the hardening plastic model to achieve a higher simulation realism. This hardening plastic model is then integrated into the simulator proposed by Narain et al. (2013) to simulate the persistent wrinkles of cloths and papers.

In addition, cloths also exhibit deformation rate dependent mechanical properties, i.e., viscosity (Sun and Hu 2020), where cloths' reaction forces are proportional to cloths deformation rate. Essentially, viscosity is widely used in cloth simulators for stabilizing simulation and has another more popular name, i.e., damping (Volino and Magnenat-Thalmann 2005). Terzopoulos and Fleischer (1988) extend their previous elastic model (Terzopoulos et al. 1987) by integrating a plastic model and introducing a viscosity model which consists of a sequentially connected Maxwell model and Voigt model to simulate nonelastic mechanical behaviors in soft bodies (including cloths). In a more recent research, Jung et al. (2016) analyze the cloth non-elastic mechanical behaviors in cloth stretching deformations. Based on the observations, they decouple cloth non-elastic stretching deformations into three components: immediate elastic deformation, viscoelastic deformation, and permanent deformation. The formation process of a cloth's persistent elongation involves both viscoelastic deformation and permanent deformation such that cloth stretching plastic deformation is time-dependent, i.e., creeping effect (the cloth can constantly extend when applying a constant stretching force). It adopts the Kelvin model to simulate this creeping behavior. Through experiments, it demonstrates their simulator can realistically simulate real cloth creeping mechanical behaviors. In addition, cloth hysteresis is also an important history-dependent deformation. As mentioned above, it is usually modeled by cloth internal friction which can also cause cloth plastic deformations and play the same role as damping in a cloth simulator. For example, Miguel et al. (2013) demonstrate their Dahl's friction model can also simulate cloth persistent deformations. Wong et al. (2013) introduce a cloth bending force model, which embeds cloth internal friction, can simulate cloth permanent wrinkles and stable the simulated cloth motions as the damping force.

Yarn-level Cloth Simulation

Cloth macro-level nonlinear elastic and non-elastic properties are the natural results of cloth micro-mechanics. However, mass-spring and shell cloth simulation methods only focus on cloth macro-level behaviors. As such, these models can only mimic the nonlinear elastic and non-elastic mechanical behaviors exhibited at the macro-level by using numerical methods (e.g., piecewise linear) or more feasible constitutive laws. The approximated cloth physics inevitably overlooks real cloth fine-grained mechanical behaviors and restricts the simulation realism. Breen et al. (1992), Breen et al. (1994a), and Breen et al. (1994b) start the early works that consider cloth micro-mechanics in simulation and introduce a particle-based cloth simulator which uses particles to denote the cross overs between the contacted yarns. Nevertheless, this cloth physical model cannot embed woven or knitted patterns (yarns' arrangement). In addition, these simulators are only designed for simulating woven cloth. However, the widely used real cloths are usually classified into woven cloth, knitted cloth, and non-woven cloth. They usually exhibit very distinctive mechanical characteristics due to their different yarn arrangement. For example, woven cloths (are usually used to make tablecloths and curtains) are inextensible. By contrast, owing to the looser yarn arrangement, knitted cloths (usually used to make t-shirts and socks) can be stretched very much and exhibit more obvious mechanical nonlinearity while being stretched. For example, a piece of knitted cloth can stretched easily at the beginning where the major deformations are caused by yarns' relative movement. As the cloth being stretched, it will be more difficult when yarns tightly contact with each other and the major deformations derives from yarns' deformations. In addition, compared with woven cloths, knitted cloths usually exhibit a more obvious Poisson effect: stretching one direction affects the orthogonal direction. However, most of the cloth simulators are designed for simulating inextensible woven cloths. Consequently, when simulating knitted cloths, these simulator tend to produce unrealistic simulation results.

Kaldor et al. (2008) introduce the first yarn-level cloth simulator for simulating knitted cloths. It models cloths as interlocked yarns each of which is modeled as (inextensible) B-splines and introduces a penalty energy to push the contact yarns toward opposite sides for handling the yarn contacts at cross overs. This work points out that the knitted cloth's nonlinear mechanical behaviors exhibited when being stretched are jointly caused by the inter-yarn structure variation and intra-yarns deformation. It demonstrates this argument through experiments: by

simulating the interactions between yarns and each yarn's deformation at the micro-level, the simulator can naturally simulate cloth macro-level nonlinear mechanical behaviors. In addition, by modeling the loose knitted patterns (the yarns arrangement in knitted cloths), it can realistically reproduce real knitted cloths' mechanical characteristics: more stretchy and obvious Poisson effect. However, handling yarn contacts is the main efficiency bottleneck of this simulator. In every simulation step, the simulator needs to iterate through all the cross overs to calculate the penalty energy (by integrating its energy density function over the length of yarn) and then compute the energy's derivatives (w.r.t, cross overs' spatial position) to acquire the contact force. Although this work tries to reduce the computational consumption by narrowing the integration interval (integrate over a short segment around the cross over rather than the entire yarn) when computing the energy, the simulator still would become prohibitively slow when simulating a large knitted cloth. Therefore, its subsequent work proposes a contact force approximation algorithm (Kaldor et al. 2010) which can speed up the yarn-yarn contact handling by 7-9 times. Consequently, it becomes capable of simulating much larger cloths, e.g., on-body garments. Generally, the algorithm calculates contact forces in one simulation step with saving the cloth state as a reference state. In the subsequent simulation steps, instead of integrating the penalty energy and calculating contact forces again (which is computationally expensive), it uses a linear function to approximate the contact forces according to the saved reference state and the cloth's current state. As long as the difference between the cloth current state and the reference state is greater than a given threshold during the simulation, the contact force will be calculated again and also the reference state will be updated. Although this method can effectively accelerate the yarn-yarn contact handling, explicitly simulating the collisions between yarns is still too expensive.

Instead, Cirio et al. (2014) assume the yarns are constantly in contact with each other at cross overs such that it can avoid explicitly handling the yarn-yarn collisions and gain a higher simulation efficiency. To simulate the constant contact yarns, it models yarns as highly constrained strands that can only slide through the spatial positions of the cross overs. To model a yarn's constrained motion, it adopts the Lagrangian-on-Eulerian method (Sueda et al. 2011) to define cross overs' spatial positions in the (3-dimensional) Lagrangian coordinate system and simulate contact yarns relative sliding motions in the (2-dimensional) Eulerian coordinate system. The cross overs' 3D spatial positions encode the simulated cloth geometry. By contrast, the 2D Eulerian coordinate system is used for modeling yarns' sliding motion. This work is designed

for simulating woven cloths. It discretizes the piece of simulated cloth's perpendicular warp and weft yarns into segments, and each segment is delimited by two adjacent cross overs in the same yarn. To further accelerate the simulation speed, this simulator leverages GPU parallel computing and finally can simulate a cloth that consists of over 2K yarns at about 2 minutes per frame. In addition, by explicitly modeling cloth micro-mechanics at the yarn-level, this simulator gains a higher simulation realism, and can naturally simulate woven cloth nonlinear elastic and non-elastic mechanical behaviors. Concretely, by modeling shearing lock (King et al. 2005) and shearing friction, this simulator can simulate cloth nonlinear elastic shear behaviors and shear wrinkles which are commonly observed in real cloths. Cirio et al. (2015) further apply this constant contact assumption to knitted cloth simulation. Moreover, it decouples the knitted cloth elastic energy into the yarn bending energy and wrapping energy where the former tends to keep each yarn in its rest shape (i.e., keep a yarn straight) and the latter is used for keeping the angle between the two wrapping yarns at the rest angle (i.e., keep a cloth flat). In addition, it also demonstrates that simulating cloth at the yarn-level can naturally simulate cloth nonlinear mechanical behaviors and modeling the sliding friction between yarns can simulate cloth persistent plastic deformations. However, this simulator only supports a few kinds of knitted cloths with simple stitches (i.e., knit and purl) that only have two contact points between every two contact yarns. Its subsequent work (Cirio et al. 2017) enriches the supported stitch types and can model the stitches consisting of multiple yarns such that can simulate more diverse knitted cloths.

In addition, based on the work proposed by Kaldor et al. (2008), Leaf et al. (2018) introduce a yarn-level woven (or knitted) pattern design tool that allows users to view a cloth's relaxation at the yarn-level on the fly with adjusting the woven patterns at the same time. Compared with sheet-level cloth simulators (which adopt the mass-spring or shell simulation methods), yarn-level cloth simulators can explicitly encode cloth woven patterns by allocating the yarn's relative position in the cloth. As woven patterns have a significant influence on cloth mechanical properties and appearance, designing woven patterns is important in textile engineering and fabric design. To achieve real-time simulation speeds, this tool represents yarn-level cloth as repetitive and periodic tiles which connected on their edges to compose an entire cloth. It assumes the mechanical behaviors (e.g., internal energies and inter-yarn collision) of all the tiles are identical such that it only needs to calculate the physics of a partition of the cloth. To ensure the geometry consistency between the adjacent tiles, it enforces a boundary condition to

connect the tiles' edges. In addition, with a GPU implementation for parallel computing, the tool finally can simulate yarn-level cloth relaxation at an interactive rate.

Since the introduction of these efficient yarn-level cloth simulators, many works have been proposed to optimize the physical model for pursuing a higher simulation realism. Sánchez-Banderas and Otaduy (2017) introduce a velocity-based dissipation potential energy that can be integrated into the yarn-level woven cloth simulator proposed by Cirio et al. (2014). In this way, the simulator not only can damp cloth motions to stabilize the simulation but also can simulate cloth viscosity mechanical behaviors to increase simulation realism.

In addition, Pizana et al. (2020) optimize the bending model used in the simulator proposed by Cirio et al. (2014) and resolves the undefined bending force direction problem. In the simulator proposed by Cirio et al. (2014), the bending energy is proportional to the bending angle between two neighbor yarn segments. To calculate the bending force, the simulator needs to compute the derivatives of the bending energy w.r.t. cross overs' spatial positions. However, when the bending angle is equal or close to zero, the bending model suffers from undefined derivatives: the derivative of the zero bending angle is zero and therefore the bending force direction is undefined. Owing to this flaw, this bending model may incur flipped bending artifacts when simulating yarns with non-zero rest bending angles, which are usually used to simulate predefined persistent wrinkles. Pizana et al. (2020) resolve this problem by calculating and saving the normals (at the cross overs) which are perpendicular to the surface of the cloth at its rest shape. When a bending angle is equal or close to zero, the bending force direction is set to the pre-computed normal direction. In addition, this method can also decouple a bending angle into an out-of-plane bending angle and an in-plane bending angle where the former is along the pre-computed normal direction and the latter is along the bi-normal direction. In this way, the simulator is allowed to use different stiffness parameters for the in-plane and out-of-plane bending deformations such that can simulate yarns' anisotropic bending behaviors.

Apart from the bending model, the yarn stretching energy in the simulator proposed by Cirio et al. (2014) may become extremely large when the two end nodes of a yarn segment are very close and this would degrade the simulation stability. To alleviate this problem, Cirio et al. (2014) define a penalty energy to repulse the parallel yarns apart to avoid this situation. However, when simulating tightly woven fabrics where the distance between every two parallel yarns is small, the simulation stability still cannot be guaranteed by this method. Instead, Sánchez-Banderas et al.

(2020) solve this problem by adopting a more robust discretization method through introducing a new node into Eulerian-on-Lagrangian coordinate: *Eulerian with Interpolation Lagrangian node* (EIL node). Concretely, when the two end nodes of a segment are closer than a given threshold, one of the end nodes is converted to an EIL node. Its 3D spatial position in the Lagrangian coordinate is given up and instead is linearly interpolated by the opposite end node in this segment and the other end node of the adjacent segment along the same yarn. In every simulation step, the simulator checks all the segment's end nodes sequentially and the EIL node is ignored when computing the stretching energy such that the simulator can avoid the great stretching forces and improve the simulation stability.

Optimize Efficiency

Compared with geometry-based cloth simulators, physics-based cloth simulators are usually slower due to the complicated physical equations and numerical integration for dynamic simulation. Apart from simulation realism, simulation efficiency is also an important consideration in computer graphics applications. Even if physics-based cloth simulators gain a higher simulation realism, an overly slow simulation speed is still unacceptable. Therefore, speeding up physics-based cloth simulators is another important research direction. Many works have proposed and some representative efficiency optimization methods are commonly adopted in modern physics-based cloth simulators.

When using the explicit (forward) Euler method to do numerical integration, the simulation time step size must be carefully limited to avoid unstable simulation which especially tends to occur when an overly large force appears. However, this would slow down the simulation efficiency because, for example, simulating a 1-minute cloth animation may need thousands of forward simulation steps. To alleviate this problem, Baraff and Witkin (1998) introduce the implicit (backward) Euler method into the field of cloth simulation to allow large simulation step size without breaking the simulation stability. Moreover, it uses an adaptive time step size which reduces time step size as long as it detects overly large forces. When adopting the implicit Euler method, the cloth motion equations become a system of nonlinear equations that usually need to be solved by the Newton method. As the Newton method needs the second-order derivatives, i.e., Hessian matrices of the forces, which are expensive to compute, using the implicit Euler method still slows down simulation speed. Essentially, solving the implicit Euler cloth motion equations is equivalent to solving a nonlinear optimization problem for minimizing

cloth total potential energy (Martin et al. 2011). To speed up the optimization, Bouaziz et al. (2014) decouple the objective function (i.e., cloth total energy) into a linear part and nonlinear part, and proposes a local-global optimization algorithm, called Projective Dynamics (PD), to replace the Newton method. Concretely, it conducts optimization by projecting the cloth's current state on a constrained manifold where the cloth's total energy is zero. Therefore, the constrained manifold is designed for modeling the cloth's internal and external forces. In the local step, PD independently projects the cloth's current state onto every manifold defined by one force. In the global step, PD finds the compromised state that can to the greatest extent satisfy all the projected states. Compared with the Newton method, PD can run multiple local steps in parallel because projection operations are independent. In addition, PD avoids calculating the expensive Hessian Matrix. As a result, PD gains a significant improvement in the simulation efficiency with an acceptable sacrifice on the numerical accuracy. Later, this solver is adopted by Ly et al. (2020) who integrate the Signorini-Coulomb law (Daviet et al. 2011) into their PD-based cloth simulator to accurately simulate the cloth frictional contacts (including cloth self-contacts and the contacts with other objects). Apart from cloth physical models and physical laws, cloth mesh resolution also affects simulation realism because it determines the minimum details that can be encoded in cloth's geometry. In real cloths, dynamic wrinkles commonly appear and disappear with cloth motions. The quality of simulated cloth wrinkles dramatically affects the overall simulation plausibility. As wrinkles are fine-scaled geometries, a coarse-resolution mesh will restrict the simulated wrinkles' fidelity. However, globally increasing cloth mesh resolution will introduce extra memory consumption and slow down the simulation speed. By contrast, Narain et al. (2012) introduce an adaptive remeshing algorithm that can dynamically adjust cloth mesh topology according to the cloth deformation. Concretely, it reduces mesh resolution in the areas where the geometry is simple, i.e., flat and smooth, by merging the vertices and, conversely, increases mesh resolution in the areas that have complex geometry, e.g., wrinkles and folds, by splitting mesh edges. In this way, this simulator can keep fine geometrical details without dramatically slowing down the simulator.

Müller et al. (2007) introduce a new simulation paradigm referred to as position-based dynamics (PBD) which is not only efficient but also stable. Different from the classical force-based simulation methods that first calculate the forces impulsed on the simulated object's physical system and then evaluate positions through Newton's second law and numerical integration, PBD directly manipulates positions by solving a quasi-static problem in which the simulator

searches for an equilibrium state of the simulated object's that can satisfy the physically inspired geometric constraints (Bender et al. 2014b). Most PBD simulators adopt the Gausi-Seidel fashion to iteratively enforce constraints such that they are efficient. As the simulated object's state must satisfy the physically inspired constraints, the object's state must be physically valid in every simulation step and PBD is unconditionally stable. In addition, apart from efficiency and unconditional stability, PBD is also simple to implement and easy to control because it avoids the velocity and acceleration layer, and directly manipulates an object's geometry. Since the work of Müller et al. (2007), many improvements have been proposed. The later work of Müller (2008) introduces a hierarchy structure and solver which are based on the Multigrid method (McCormick 1987) so that it can define the object's state and constraints in different levels from coarse to fine. This work enables their PBD-based simulator to enforce the constraints in a coarse-to-fine manner and leverage the Multigrid method to improve the convergence performance. Müller et al. (2015) apply geometric constraints on the deformations defined by rotation invariant Green-St Venant strain, widely used in continuum mechanics, to easily simulate cloth anisotropic behaviors: large stretching stiffness in warp and weft direction, but low stretching stiffness in the diagonal direction, i.e., shearing stiffness. Moreover, Bender et al. (2014a) integrate continuum mechanics into PBS by using Venant-Kirchhof or Neo-Hookean constitutive models to define energy-based constraints. In this way, their PBS cloth simulator can handle complex isotropic and anisotropic elastic behaviors, embed elastoplastic material properties, and also simulate lateral contraction. Nowadays, PDB can simulate various materials including rigid bodies (Deul et al. 2016), elastic rods (Umetani et al. 2015), fluid (Macklin and Müller 2013), and the interactions between different materials (Macklin et al. 2014; Shao et al. 2018). It has been widely used in games, movies, virtual reality, and popular simulation engines, e.g. PhyX (Nvidia 2024), Bullet (Coumans and Bai 2016). However, PBD is a physically inaccurate simulation method because the user-defined material stiffnesses are affected by the time step size and number of solvers iterations. Concretely, as the time step size decreases or the number of iterations increases, the simulated material becomes stiffer (Bender et al. 2015). XPDB (Macklin et al. 2016) mitigates this problem by newly formulating the constraints to model material elastic potentials so that the constraints be enforced in a time step size and iteration count independent manner. Quan et al. (2020) further improve XPDB in cloth simulation and address the over-stretching problem by reducing time step size. Another long-standing problem in PDB is that the simulation does not converge to a certain solution with

mesh refinement and using adaptive mesh in PBD is still an open problem. In summary, PDB is especially suitable for interaction applications that need to produce visually plausible simulations at a real-time speed. Compared with the force-based simulation methods, PBD is less feasible for pursuing the high simulation realism.

Although the yarn-level cloth simulation brings about a higher simulation realism, it also reduces the simulation speed. No matter what optimization method is taken, compared with sheet-level cloth simulators, yarn-level cloth simulator inevitably need a much higher space and time complexity because of the more fine-grained geometrical discretization and more complex physical laws. However, the simulated cloth does not necessarily always exhibit very complicated mechanical behaviors that cannot be simulated by a sheet-level cloth simulator. Sometimes, simulating a cloth's yarns-level mechanical behaviors does not gain any simulation realism improvement, but wastes computational resources. Casafranca et al. (2020) propose a hybrid method which mixes sheet-level and yarn-level cloth models. It introduces a kinematic filter for designating the yarn-level cloth model to simulate the complex mechanics that cannot be handled by the sheet-level cloth model. However, the dramatic variation of the discretization resolution in the yarn-level cloth model and sheet-level cloth model (where the vertices in the former are far more sparse than those of the latter) tends to incur convergence problems and drastic simulation artifacts. To resolve these problems, it adopts the modified preconditioned conjugate gradient solver (Ascher and Boxerman 2003) and introduces a specially designed preconditioner for cooperating with their hybrid cloth simulation method to solve cloth motion equation. In this way, this simulation method gains a higher simulation efficiency by letting a faster sheet-level cloth model simulate the areas with simple mechanical behaviors. Compared with fully yarn-level cloth simulators, it gains a significant improvement in the simulation efficiency without sacrificing the simulation fidelity derived from cloth yarn-level mechanics (nonlinearity, internal friction, and plasticity).

This subsection reviews the development of physics-based cloth simulators that gradually improve the cloth simulation realism by accurately modeling real cloth mechanical behaviors. However, pursuing a higher simulation realism inevitably introduces more complex physical models and laws into cloth simulators. As a result, physics-based cloth simulators become slower and more computationally expensive. Thanks to the fast growth of parallel computing (especially GPU parallel computing), many cloth simulator can simulate high-fidelity clothes

with an acceptable speed (Tang et al. 2013; Li et al. 2020; Wang 2021). Section 2.1.3 will introduce another paradigm for increasing simulation efficiency: data-driven cloth simulation.

Collision Handling

Before introducing the data-driven method, this subsection ends with a short review of collision handling which is an indispensable component in physics-based cloth simulators, even if it is not closely related to the contributions of this work. Collision handling is responsible for simulating the interactions between the objects and avoiding penetration errors. Cloth is so soft and highly deformable that it tends to exhibit complex geometries in motions. As a result, compared with rigid bodies, cloths not only can contact other objects in the environment but also may collide with itself (Carignan et al. 1992). Collision handling is usually achieved by the cooperation of collision detection and collision response: the former is responsible for detecting the geometry in proximity; the latter is designated to rectify cloth geometry to eliminate penetrations (Moore and Wilhelms 1988). Continuous collision detection (CCD) is commonly adopted for robust collision checking (Provot 1997; Brochu et al. 2012; Wang 2014). To guarantee collision response produces a piece of collision-free cloth, the works proposed by Bridson et al. (2002) and Harmon et al. (2008) uses both repulsive impulse method (Baraff 1994; Sifakis et al. 2008) and geometric method (Volino et al. 1995). Moreover, collision handling is usually the simulation bottleneck. To alleviate this problem, the collision handling methods proposed by Zhang and Kim (2014), Tang et al. (2011), Tang et al. (2018b), and Tang et al. (2018a) leverage CPU and GPU parallel computing to boost efficiency.

2.1.3 Data-driven Method

Different from physics-based methods, data-driven methods do not rely on physical laws to determine cloth motions. Instead, they learn the natural cloth dynamics from real or simulated data such that they do not need to calculate forces, do numerical integration, and solve motion equations at the run time. Generally, the rationale behind data-driven methods is equivalent to selecting and playing back the cloth dynamics in the training dataset. Therefore, compared with physics-based methods, data-driven methods have intrinsic advantages in the simulation efficiency. They are usually preferred in applications that demand the simulation speed (e.g., gaming, real-time virtual try-on). Additionally, they are also widely used to add fine-scaled details on the coarse mesh such that physics-based cloth simulators can use low-resolution meshes

without losing the simulation fidelity. Generally, data-driven cloth simulation methods can classified into pre-computing methods, machine learning methods, and self-supervise learning methods.

Pre-computing Method

The early data-driven cloth simulators usually adopt the pre-computing paradigm which first builds a database that stores various cloth dynamics that are simulated in advance. Then, the simulators only need to make a selection in the database when running a simulation. For instance, James and Fatahalian (2003b) propose a data-driven simulator for simulating soft bodies. The pre-computed object dynamic database consists of an object's motions each of which is represented by a temporal sequence of the object's states. Each state sequence is denoted by a matrix and each column of this matrix encodes the object's mesh vertices positional displacement from vertices' initial positions. To control the storage consumption, it adopts the principle component analysis (PCA) method to compress the matrices in the database. In the simulation stage, the simulator only needs to play a cloth motion saved in the database or switch to another motion when an impulse force is exerted on the simulated object. There is no complex computation in the simulation stage, so it can achieve the real-time simulation speed. However, this model is only applied in three simple simulation scenarios: a dinosaur on a moving car dashboard, a plant in a moving pot, and a piece of cloth on a moving door, where the pre-computed motions have a very low degree of freedom (DoF) and the database is sparse. For example, the moving door only has three angular velocities: $\{\omega, 0, -\omega\}$. Consequently, it can only be applied in very restricted simulation scenarios. By contrast, Cordier and Magnenat-Thalmann (2005) introduce a more flexible data-driven cloth simulator for interactively simulating on-body garments. It is capable of simulating the garments made from different materials and the garment's geometry on the bodies with various poses. However, rather than simulating a piece of cloth's overall dynamics by data-driven methods, this simulator uses data-driven methods to assist a (mass-spring) physics-based cloth simulator which is only responsible for simulating cloth gross motions with very coarse mesh resolutions. Concretely, it separates a coarse garment mesh into tight regions and float regions according to the distance between the garment and the body. A tight region is close to the body, so its deformation is almost determined by the underlying body motion. The simulator uses data-driven methods to simulate tight regions by using optimization methods to search for the closest cloth geometry in

the pre-computed database. By contrast, a float region is relatively far from the body, so it is difficult to predict this region's deformation according to the body's motion. Therefore, the simulator uses physics-based methods to handle float regions. In addition, data-driven methods are also responsible for correcting and refining the geometries of the coarse meshes by weighted blending the pre-computed high-resolution garment geometries. The weights are the Euclidian distances between the garment's coarse mesh and pre-computed fine mesh. This simulator can also run at a real-time speed because most of the time-consuming fine-scaled simulations have been shifted to the pre-computing stage. Instead of assisting a physics-based simulator, Kim and Vendrovsky (2008) introduce a data-driven cloth simulator that can independently interpolate an on-body garment shape according to a given body pose. This simulator first selects multiple body-garment pairs in a pre-computed database according to the Euclidian distances between the given body pose and the body poses in the database. Then, it merges the corresponding garments of the selected body poses with the weights that are calculated by using a quadratic programming algorithm to solve an constrained optimization problem. The objective function in the optimization problem measures the Euclidian distance between the merged body poses and the given body pose. The constraints limit the range of the weights, i.e., within $[0, 1]$. This simulator was applied in the production of the 3D animation *Alvin and the Chipmunks* (Alter 2008).

Instead of using the naive Euclidian distance to determine the blending weights, Feng et al. (2010) build a database consisting of coarse-fine garment mesh pairs, and train two regression models to blend and add fine-scaled geometry. They notice that, after removing the fine-scaled details on a garment (e.g., wrinkles), the garment's deformations are coherent in the local regions such that its geometry can be approximated by low-resolution patches. Therefore, it uses a physics-based simulator to conduct simulations with a low-resolution garment mesh. In every simulation step, the simulated garment geometry from the physics-based simulator is corrected by one of the trained regression models to fit the pre-computed coarse mesh. Next, this corrected mesh is taken as input of the other regression model which is responsible for adding fine-scaled wrinkles from the pre-computed high-resolution garment database. Instead of adding fine-scaled wrinkles on an entire garment, Wang et al. (2010a) propose a real-time data-driven cloth simulator that only adds wrinkles around the joint areas of an on-body garment. This work assumes that the wrinkles on on-body garments mainly result from the motions of body joints and a joint's motions only locally determine the wrinkles on the garment's nearby

areas. Based on this assumption, in the pre-computing stage, it independently simulates the wrinkles around the nine selected joints where wrinkles tend to appear by sampling multiple joint rotation angles. In the simulation stage, the simulator interpolates the wrinkles for each joint according to the joint's rotation angle by sampling from the pre-computed database and then blending the wrinkles with a distance-based weighting function. Next, the merged wrinkles are transferred to the coarse mesh whose motion is simulated by a simple physics-based cloth simulator. Even though this simulator can produce visually pleasing garments with interactive simulation speed, it tends to exhibit artifacts when simulating loose garments where wrinkles may be affected by non-local body motions. Moreover, the pre-computed database in this work is too sparse because of the few sampling rotation angles (only uniformly samples 15 degrees for each joint's rotational DoF). Xu et al. (2014) resolve the first problem by using the sensitivity analysis-based optimization method, and simultaneously considering the influence of a body's local and remote bonds on the wrinkles. Sensitivity analysis is used for evaluating the output of a function with a perturbation of the input (Tortorelli and Michaleris 1994). This work adopts the sensitivity analysis to evaluate the garment geometry variation resulting from the small changes in the body pose (bonds position). Thus, through solving an optimization problem, it can determine the motions of the non-local bones that considerably affect the simulated garment's geometry. In addition, this work alleviates the second problem by adopting a stochastic (Monte Carlo Markov Chain) and greedy sampling strategy to build a much richer database. In this way, this simulator gains an improvement in the simulation realism. In addition, this work also demonstrates that the simulation realism and generalization ability of the pre-computing data-driven cloth simulators are decided by the diversity of samples in the database.

In general, simulation realism and generalization ability can be improved by introducing more diverse samples into the database. However, arbitrarily increasing samples will make the pre-computing stage prohibitively long and cause an intractably large storage consumption. Therefore, for controlling the database size, data-driven cloth simulators usually have to restrict the simulation scenarios. Furthermore, they usually ignore cloth dynamics, i.e., velocity and acceleration, and assume cloth deformations are repetitive, e.g., the wrinkles on a garment are always the same when they are on the bodies with the same pose. However, due to the body's motions, real garment deformations are not necessarily the same. To pursue a higher simulation realism, Kim et al. (2013) additionally consider garment's dynamics in their data-driven cloth simulator and studies cloth deformations' bifurcations, where two same garments can exhibit

different geometries when they are worn on the bodies with the same pose. Concretely, it uses two graphics, a primary graph and a secondary cloth motion graph, to denote the movements of the body and the garment, respectively. Each note in the graph denotes the state of the pose-garment and each edge represents a movement of the pose-garment in the database. To simulate cloth deformations' bifurcations, each node in the primary graph is allowed to have multiple corresponding nodes in the secondary cloth motion graph. The simulator selects the garment deformation node according to the body movement embedded in the primary graph (i.e., the path from the first node to the current node in the primary graph). To simulate as many as possible cloth deformations in a pose, it does exhaustively sampling and simulation in the pre-computing stage. Thanks to the dense samples in their database and the additional dynamic information, this simulator can almost reach a physics-based cloth simulator's simulation fidelity and realistically simulate cloth deformations' bifurcations.

Machine Learning Method

The pre-computing data-driven cloth simulation method must be bound to a database from which a simulator selects the needed cloth deformations in the simulation stage. To guarantee the simulation realism and generalization ability, the database needs to include many samples and, consequently, consume a considerable amount of storage. By contrast, learning-based methods only use a database for training machine learning models which no longer need a database in the simulation stage. Kavan et al. (2011) use a physics-based simulator to build a cloth database consisting of coarse-fine mesh pairs. It trains a linear regression model for upsampling a coarse mesh to a fine mesh. In the simulation stage, it uses the physics-based cloth simulator to simulate with a coarse cloth mesh which is then refined by the trained upsampling linear regression model. Moreover, it introduces a regularization method (referred to as *harmonic regularization*) to avoid overfitting and gain a decent generalization ability. However, geometrically, this regularization method plays a role like a filter which is responsible for smoothing the overly fine-scale geometries in the training samples. As a result, this regularization method tends to make the simulated cloths look smoother than expectations. This simulation flaw is especially obvious when simulating a cloth under constant forces, e.g., a cloth in the wind. To resolve this problem in this particular simulation scenario, it also introduces a wave propagation method for adding fine-scaled geometries to the upsampled fine meshes. In addition, De Aguiar et al. (2010) observe that the motions of the different facets of on-body garments

are not independent. Therefore, it is possible to compress the garment geometry information, encoded by all vertices' spatial positions, such that simulations can more efficiently run in a low-dimensional space. Therefore, this work adopts the PCA method to compress the garment and body geometrical information. Before compressing, the body and garment geometries are moved to a canonical space by rotation and translation through a pre-processing operation such that the bias caused by the global rigid rotation and translation can be eliminated. This can help the trained model gain a higher generalization ability. It trains a linear regression model to match the garment and body pairs in the low-dimensional space (after compressing by PCA). The linear regression model takes as input the body static geometry, garment dynamics information (garment geometry in the current and two previous simulation steps), and the global rigid movement (rigid rotation and translation) information (which has been removed in the pre-processing stage). The trained linear regression model outputs the garment's future geometry. Moreover, instead of using synthesis training data simulated by physics-based cloth simulators, it uses real-world captured on-body garment motions as training data such that its simulation realism is not restricted by the physics-based cloth simulator. Compared with the pre-computing data-driven method, apart from getting rid of the large databases, it shows a stronger generalization ability and performs well when simulating non-skintight cloths.

However, the works mentioned above only fit the simulated garments to different poses. A realistic on-body garment not only needs to deform with the wearer's motion but also needs to fit different body shapes. For example, a virtual try-on application should be able to wear the virtual garments on customers with various body shapes and also need to interactively deform the garments to follow the customers' motion. The previous works cannot be applied in these scenarios because they use the same body shape in their training database. However, a database that consists of both various motions and body shapes can be prohibitively large: the Cartesian product of the sampled motions and body shapes. Instead, Guan et al. (2012) decouple a garment's deformation into the deformations caused by the body shape and the deformations caused by the body motion. In this way, the training database needs much fewer samples: the summation of the sampled motions and body shapes. To learn the garment deformations caused by these two factors, this work trains two linear regression models for adjusting a garment geometry to motions and different body shapes respectively. Similar to (De Aguiar et al. 2010), it also adopts the PCA method to reduce the dimension of the cloth geometrical information and considers cloth dynamics to fit body motions (takes as input the garments geometry in two

previous steps). The two linear regression models take as input the body shape and motion information respectively. They output two deformation matrices which can be multiply with a vector that embeds a template garment geometry (a garment in its initial rest shape) to calculate the garment geometry that can fit a given body (with a specific pose and motion). As such, this data-driven cloth simulator can realistically simulate garments on the bodies with different poses (motions) and shapes with real-time simulation speed.

However, the linear regression model is too coarse to account for the body-garment relationship which usually exhibits a complicated nonlinearity and therefore tends to incur simulation artifacts. To model the nonlinearity, Santesteban et al. (2019) use a neural network (NN), which is nonlinear, to fit garments to various body shapes and motions. The same as the work proposed by Guan et al. (2012), this work also factorizes an on-body garment's deformations into body-shape-dependent deformations and body-motion-dependent deformations. It adopts a multilayer perceptron (MLP) NN to model the body-shape-dependent deformations and a Gated Recurrent Units (GRU)-based recurrent neural network (RNN) to handle the body-motion-dependent deformations. Compared with the linear regression models, these nonlinear NNs can more accurately learn the nonlinearities that are exhibited in the body-garment data samples. In addition, unlike the works proposed by Guan et al. (2012) and De Aguiar et al. (2010) which consider cloth dynamics by simply including garment geometries in the two previous steps, the GRU-based RNN can learn the needed number of the previous simulation steps, and is usually more accurate and robust in handling temporal data (Chung et al. 2014). Compared with the linear regression model, this simulator produces fewer artifacts caused by ignoring nonlinearity. Thanks to the RNN, it can retain more cloth history-dependent deformations, e.g., wrinkles. Moreover, it exhibits a stronger generalization ability that can generalize to body shapes and motions that have not been seen in the training data. Further, through GPU implementation, this data-driven simulator can achieve interactive simulation speed and suit real-time virtual try-on applications.

All of the data-driven cloth simulators introduced above need to append a post-processing step for handling collision and removing cloth-body penetration artifacts, which is slow and breaks the overall differentiability. Instead, GarNet (Gundogdu et al. 2019) handles collisions by introducing a penalty term in the loss function for repulsing the wrong geometrical intersections apart. In this way, the outputs from their trained data-driven cloth simulators must be free

from the penetration artifacts and the slow post-processing step is no longer needed. Moreover, it also introduces a physical term in the loss function: a bending loss to penalize the bending deformations that are different from the train data. In addition, it argues that an on-body garment geometry is jointly determined by the global body geometry and the local body-garment interactions. Therefore, it factorizes a garment's geometrical features into three kinds: local point-wise features, local patch-wise features, and global features. To extract these features from a garment's geometry, it introduces a novel NN model that integrates a mesh convolution NN (Verma et al. 2018) into the PointNet (Qi et al. 2017). The mesh convolution NN acquires the patch-wise features, and the PointNet can supply the point-wise and global features. Then, to fit the garment onto a body, these features are merged with the body features that are extracted by an MLP NN in the parallel stream (therefore, the GarNet is a two-stream NN). However, GarNet tends to smooth garments and loses fine-scale geometries, like wrinkles and folds. This drawback is also observed in the previous data-driven cloth simulators (Guan et al. 2012; Santesteban et al. 2019). The research group alleviates this problem in its subsequent work, GarNet++ (Gundogdu et al. 2022) uses the Rayleigh quotient (Strang 2012) to extract a garment's geometrical curvature information and defines a curvature loss term for learning the fine-scaled deformations. The experiments demonstrate that this curvature enables the simulator to more accurately simulate the garment's geometrical details. Apart from the body pose and shape, a garment's style, e.g., a garment's geometry or size, also affects its on-body appearance. TailorNet (Patel et al. 2020) is the first data-driven cloth simulator that considers these three factors at the same time. This work observes that the simulated garments can keep the fine-scaled geometrical details if the data-driven cloth simulator's NN model is trained with fixing body shape and garment style factors. Conversely, if training the model by mixing the three factors, the simulated garments look unnaturally smooth and unrealistic. It hypothesizes that mixing the fine-scaled deformations caused by the different factors tends to average the deformations and flatten the fine-scaled geometries. Therefore, TailorNet decouples a garment's deformation into a low-frequency part (i.e., garment gross deformation) and high-frequency (i.e., fine-scaled) part, and learns them separately by using two sub-networks. In this way, TailorNet can retain more high-frequency geometrical details than the previous works. Most data-driven cloth simulators usually take the simulated garments' mesh topologies as a prior. This limits the simulators' flexibility and generalization ability, i.e., cannot handle the garments out of the training data. Bertiche et al. (2021a) introduce a more flexible data-driven cloth simulator,

DeepPSD, that can deform a garment to fit a body without the garment topology prior. Compared with the works mentioned above, it can more realistically simulate the garments that have not been unseen in the training data. Moreover, the loss function introduces an unsupervised loss term for handling collision and guaranteeing the garment's physical consistency: enforcing the inextensibility to simulate the stretching force and smoothing the geometry to mimic the bending force. Therefore, it mixes supervised learning and unsupervised learning where the former is responsible for turning the NN model's parameters to fit training samples and the latter is designated to enforce cloth physical consistency. In this way, the cloths simulated by DeepPSD look more realistic and natural.

Apart from virtual try-on, fashion design applications also need a fast cloth simulator for interactively viewing and editing the designed on-body garments. To this end, Wang et al. (2018) introduce a data-driven cloth simulator that can convert a 2D garment sketch to a 3D on-body garment mesh and its sewing pattern. It realizes this by training a multiple encoder-decoder NN model which can embed the information of 2D sketch, garment material parameters, garment sewing pattern parameters, body geometry, and 3D garment geometry into a shared latent space. In this space, given the other parameters or altering the garment by editing the 2D sketch, the model can interactively respond to the changes and simulate the corresponding 3D on-body garment. In addition, this shared latent space provides a common metric that can be used for evaluating the difference between different parameters. For instance, there is no way to directly compare the difference between a 2D sketch and a 3D garment mesh. However, given the shared latent space, two different kinds of parameters can be compared by evaluating their distance in this space. With this feature, this simulator can retarget a garment to different body shapes without affecting the garment's style from the perspective of fashion design. It minimizes the difference between the variation of the 2D sketch and the variation of the 3D garment after retargeting the garment to a new body. Further, this data-driven simulator is fast and allows designers to view and alter the garments on the fly.

Most data-driven cloth simulators introduced above are applied in one restricted simulation scenario: on-body garment simulation, and impose strong restrictions on cloths' and obstacles' mesh topology. However, the simulation scenarios in practice are usually more diverse. Whether the data-driven cloth simulators can be applied in other simulation scenarios has not been demonstrated. To fill this research blank area, Li et al. (2022b) evaluate their new data-driven

cloth simulator in more a flexible simulation scenario: a cloth collides with a rigid body, and does not use any prior on their mesh topologies. It also handles cloth-obstacle collision and cloth self-collision by using a penalty term in the loss function. Moreover, this simulator can efficiently handle high-resolution meshes and complex geometries by using an encoder-decoder NN such that the cloth-obstacle interactions can be processed in a smaller latent space. As a data-driven simulator, it also inherits the outstanding simulation efficiency and can achieve a real-time simulation speed.

Simulating cloth micro-mechanics at the yarn-level brings about not only a higher mechanical accuracy, but also impressive visual results, i.e., render yarn-level cloth geometry (Kaldor et al. 2008; Cirio et al. 2014; Cirio et al. 2017). Till now, the lowest level that can simulated by state of the art physical cloth simulators is the yarn-level. However, in real cloths, each yarn is usually a bundle of fibers and these fibers also affect a cloth's macro-level appearance. Simulating the mechanical behaviors of all the fibers in a cloth is computationally intractable. Instead, Montazeri et al. (2021) use a data-driven method to synthesize the fiber-level appearance for adding finer visual details on the yarn-level physics-based cloth simulation results. It realizes this by training an NN model for mapping a yarn's deformations to its fiber geometry. In this way, the simulator only needs to conduct physics-based cloth simulation at the yarn-level and the NN model can efficiently generate the fiber-level geometries according to the yarns' deformations.

Self-supervised Learning Method

Although machine learning methods get rid of the large pre-computed database, they still need a huge amount of training data for conducting supervised learning. On the one hand, building such a training dataset is time- and storage-consuming. On the other hand, machine learning models' generalization ability is usually restricted by the dataset. To get rid of these limitations, Bertiche et al. (2021b) propose the first self-supervised learning model, PBNS, for simulating a garment's static drape on a body. Concretely, the model consists of 4 fully-connected layers, i.e., MPL, for mapping a template garment to a garment that can fit a given body. To learn in a self-supervised way, the model defines a physically inspired loss function which consists of four terms: cloth loss, collision loss, gravity loss, and pin loss. The cloth loss measures cloth stretching and bending energy. The collision loss penalizes incorrect geometry intersections. The gravity loss is equivalent to the cloth gravity potential. Last but not least, the pin loss is used for fixing

the simulated garment in a given spatial position. Therefore, minimizing this loss function is equivalent to searching for the garment's stable state (where the garment has the minimum internal and external potential energy) and also correcting the penetration artifacts. In this way, PBNS not only does not rely on any huge training dataset but also can always simulate physically constant garments thanks to the physically inspired loss function. However, this model can only simulate static on-body garments because it overlooks the dynamics information. By contrast, Bertiche et al. (2022) additionally consider cloth dynamics by introducing an RNN into the model to handle the temporal information and adding an inertia term into the loss function to measure garment kinetic energy. By additionally learning garment dynamics, this simulator gains a higher simulation realism than PBNS.

In summary, the data-driven method is an effective way to dramatically improve cloth simulation efficiency. Its development process witnesses an improvement in the simulation realism and flexibility. Compared with physics-based cloth simulators, data-driven cloth simulators are more feasible for real-time simulation applications, e.g., virtual try-ons and video games. Nevertheless, their simulation results may still include penetration artifacts and, unfortunately, there is still no effective way to fully solve this problem. Moreover, they are less flexible than physics-based cloth simulators. For instance, even a state-of-the-art data-driven cloth simulator is still unable to handle scenarios that are too different from the training samples. Thus, physics-based cloth simulation methods are the best choice when pursuing high simulation realism.

2.2 Parameter Estimation

Cloth physical parameter estimation starts in textile engineering and aims at measuring cloth mechanical properties for standardizing the productions (Hu 2008). There are various testing apparatuses and the corresponding international or local standards (e.g., KES-F) for evaluating cloth different physical properties. The testing apparatuses are expensive and require the manipulators to have sophisticated expertise in the testing. Moreover, taking the tested cloth physical parameters as the simulation parameters of a physics-based cloth simulator usually cannot reproduce the real cloth in simulation because a cloth physical model is only an approximation of actual physics in real cloths and there is no simple map from the textile-tested parameters to the simulation parameters (Kuijpers et al. 2020). Additionally, the textile testing usually only measures cloth static mechanical properties and overlooks cloth dynamics physical

properties which, however, are also important to cloth behaviors. For simplicity, we refer to the cloth physical parameters used by physics-based cloth simulators as cloth physical parameters (which are different from the cloth physical parameters defined in textile).

Apart from cloth physical models, cloth physical parameters used in physics-based cloth simulators also determine the simulation realism because they determine whether the simulated cloth looks like a specific real cloth. Manually tuning these parameters is tediously time-consuming and laborious. To alleviate this problem and gain a higher simulation realism, many works have been proposed for estimating cloth physical parameters. Estimating cloth physical parameters can be taken as an extending application of the data-driven cloth simulation. Rather than learning cloth global geometries or local wrinkles, it only learns cloth physical parameters from training data to reproduce the observed cloths. Generally, cloth physical parameter estimation methods can be classified into: black-box methods and white-box methods. This section will show the works that adopt black-box methods and white-box methods in and . It will highlight the advantages of white-box methods (especially the differentiable physics simulation) in sample efficiency, learning accuracy, and convergence speed.

2.2.1 Black-box Method

Black-box methods directly map observed cloth dynamics to cloth physical parameters without explicitly modeling the underlying physics. Bouman et al. (2013) conduct a novel human perceptual experiment and the result shows that the cloth area weight and bending stiffness are the two most significant cloth physical parameters that determine people's feelings about a cloth's material when they observe its motion. Based on this conclusion, they train a regression model to predict cloth area weight and bending stiffness. As a machine learning method, it can only learn from the pre-defined features which must be carefully designed to guarantee the learning accuracy. It adopts the optical flow (Liu 2009) to extract cloth motion information from video and uses robust statistical features (Portilla and Simoncelli 2000) such that the model can handle diverse training data: unknown light settings and various external forces. By contrast, deep learning methods are more flexible and can identify the features from raw training data. Therefore, it is more feasible to learn cloth parameters from more random and noisy training samples, such as a cloth motion captured in a complicated environment including various unrelated objects. Yang et al. (2017) train a Long Short-Term Memory (LSTM) Neural Network + Convolution Neural Network (CNN) model, which is similar to Long-term Recurrent

Convolutional Network or LRCN (Donahue et al. 2017), for estimating cloth physical parameters from videos of wind-blown cloth dynamics. Compared with machine learning methods, it does not require the manually defined features and can learn cloth parameters from videos with unpredictably complex backgrounds. However, this work discretizes continuous cloth physical parameters into discrete material types and essentially converts a continuous regression problem into a much simpler classification problem. Nevertheless, the experiments demonstrate that even if the model is trained by simulated data, it can accurately classify cloth materials from both simulated data and real data. Moreover, compared with the work proposed by Bouman et al. (2013), it achieves a higher material classification accuracy. Instead of doing classification, Ju and Choi (2020) aim at estimating the cloth physical parameters (regression) that determine a simulated cloth static Cusick drape in the physics-based cloth simulator, CLO3D (*CLOTH3D* 2018). It introduces a boundary vector for encoding cloth drape shape and its entries consist of the 3D positions of the points which are sampled on the boundary of a drape cloth. In addition, through a correlation analysis, it demonstrates that, in CLO3D, the simulated cloth static Cusick drapes are mainly determined by the bending stiffness parameters and shearing stiffness parameters. It also observes that the correlation between bending stiffness and shearing stiffness is weak. Based on this observation, it introduces a two-stream CNN where both streams take as input cloth drape boundary vectors, but one of the streams predicts the bending stiffness parameters, and the other outputs the shearing stiffness parameters. To build the training dataset, it uses CLO3D to synthesize cloth static Cusick drapes. Nevertheless, the trained model is capable of estimating the cloth physical parameters from real-world captured data and reproducing the observed drape cloths. In their subsequent work (Ju et al. 2022), the NN is modified by exchanging its input and output: taking as input the simulation parameters and output cloth static drape, such that it is converted into a data-driven cloth simulator. By wrapping this NN with a user interface, it allows fashion designers to view a cloth's static drape on the fly with tuning its physical parameters and avoids the long waiting time when using physics-based simulators. However, compressing a drape cloth geometry by using the boundary vector would inevitably lose some geometrical information and degrade the parameter estimation accuracy. Moreover, cloth nonlinear mechanical property is not considered in these parameter estimation methods.

To alleviate these problems, Feng et al. (2022) use multi-view depth images as training input such that cloth geometrical information can be fully captured. In addition, it introduces a non-

linear anisotropic bending model for embedding cloth nonlinear mechanical properties. Apart from cloth internal physical parameters, the parameters of external forces also affect a cloth's motions and therefore should also be accurately estimated. Rasheed et al. (2020) train a LRCN for estimating the static friction coefficient between the cloth and a surface. The trained network can map a video clip of a dragged cloth sliding on a surface to an estimated static friction coefficient between the cloth and the surface. Even though it is trained by the synthesized data generated by a physics-based cloth simulator (Li et al. 2018), the trained model exhibits a decent generalization ability to real data. Apart from friction, Runia et al. (2020) estimate cloth internal physical parameters and external wind speed velocity from the motion of a cloth in wind, i.e., a flag blown by wind. However, different from the research mentioned above, the NN proposed by this work (called Spectral Decomposition Network) only plays a role like an embedding function and is only designated to extract cloth physically related features from images or video frames. The cloth physical parameters and wind speed velocity are estimated by using Bayesian Optimization or, for short, BO (Frazier 2018). The optimization is performed in a simulation-in-loop manner: the BO algorithm repeatedly samples cloth physical parameters which are then given to the physics-based cloth simulator (Narain et al. 2012) to conduct simulation whose result is then compared with the ground truth to determine if the optimization has converged. Even if this work needs a large training dataset to train the NN, this work explicitly models cloth physics and it is similar to the white-box methods to be introduced below.

2.2.2 White-box Method

White-box methods explicitly model cloth physics and search for feasible cloth physical parameters by minimizing the difference between the simulation results and the given observations. Compared with black-box methods, white-box methods do not rely on a huge amount of training data. Instead, they are usually much more sample efficient and only need one cloth static geometry or dynamic motion to search for the cloth physical parameters that can enable the underlying physics-based cloth simulator to closely reproduce them. Jojic and Huang (1997) estimate cloth physical parameters from both the simulated and real cloth static drape range (the boundary of a static drape cloth on a 2D image). It uses Powell's method (Powell 1971) to optimize cloth physical parameters by adjusting the simulation results toward the given cloth drape range. The experiment results show that this model can accurately reproduce observed drape cloths. However, a cloth's physics properties are not only indicated by its static ge-

ometry but also reflected in its dynamic motion. Bhat et al. (2003) additionally learn from the dynamics of a hung cloth (video clips). It adopts the simulated annealing optimization method (Bertsimas and Tsitsiklis 1993) to estimate four real cloth static and dynamic physical parameters by narrowing the difference between the simulation and the given video clips. The experiments demonstrate that their model can accurately reproduce the four distinctive kinds of real cloth's static geometries and dynamic motions. Moreover, the results also show that this method has a great temporal and spatial generalization ability. The former allows the simulator to predict cloth future states that are unseen in the training data. The latter enables the simulator to simulate the cloths with the other topologies out of the training data. e.g., learning cloth physical parameters on squared cloth samples and simulating a skirt made from this kind of cloth. In a white-box method, its underlying cloth physical model determines the best achievable parameter estimation accuracy. A coarse cloth physical model that cannot simulate the real cloth's complicated behaviors can never simulate the observed cloths no matter how the simulation parameters are optimized. For instance, Jojic and Huang (1997) take cloth as a linear isotropic elastic model so that it cannot learn the nonlinear and anisotropic mechanical properties that are shown in the training data. By contrast, Wang et al. (2011) introduce a piecewise linear model for simulating cloth nonlinear and anisotropic mechanical behaviors in stretching and bending deformations. Concretely, this model splits the (stretching and bending) load-deformation process into multiple segments and cloth physical parameters have different values in each segment. Meanwhile, it also proposes a simple and economical apparatus which can cooperate with computer vision methods for testing real cloth stretching and bending deformations under different loads. By using BFGS optimization (Nocedal and Wright 2006), it adjusts the cloth physical parameters to fit the measured results. With the estimated parameters, the simulator can realistically simulate the mechanical behaviors of the corresponding real cloths. However, this work assumes that a cloth deforms uniformly in the deformation process and uses the average strain to approximate the cloth's real deformation. As real cloths usually exhibit non-uniform deformations, this approximation inevitably introduces parameter estimation and simulation errors. To more accurately measure cloth non-uniform deformations, Miguel et al. (2012) introduce an apparatus that can reconstruct cloth 3D geometry while manipulating the tested cloth by accurately controlled forces (generated by 8 linear actuators). In this way, it can accurately capture a cloth's non-uniform deformations. When estimating cloth physical parameters, the model can adjust the cloth simulation parameters

by minimizing the geometry difference (i.e., the L2 distance between the position of vertices in the simulation and reconstructed 3D geometry) rather than using the coarse average strain. Additionally, to model more diverse nonlinear mechanical behaviors, instead of using a fixed number of linear segments to approximate the nonlinearity, this work gradually increases the number of segments before the difference between the simulation and measurement becomes smaller than a threshold.

However, none of the work mentioned above models and evaluates the cloth history-dependent behaviors, e.g., hysteresis and plastic deformation. By contrast, Miguel et al. (2013) use cloth internal friction to account for hysteresis and adds Dahl's model (Dahl 1968) into an elastic cloth physical model to simulate it. In addition, this work observes that cloth hysteresis usually grows as cloth deformation increases, which cannot be modeled by the original Dahl's model. To fit this observation, it modifies the original Dahl's model by reparameterization to make the frictional stress strain-dependent. Moreover, it also introduces a simple apparatus (similar to the apparatus created by Wang et al. (2011)) for measuring cloth force-deformation relationships and uses the trust-region reflective algorithm (Matlab build-in optimization algorithm) to optimize cloth physical parameters: elastic parameters and frictional parameters. Through experiments, this work demonstrates this friction model can simulate the cloth hysteresis effect and persistent deformations. However, the persistent deformations simulated by their Dahl's model are visually trivial and are easy to undo. By contrast, Jung et al. (2016) use the plastic model to simulate cloth persistent stretching deformations. In addition, real clothes usually exhibit creeping behaviors in the stretching deformation where a cloth's response force gradually reduces over time while keeping its stretching deformation. This work introduces a Kelvin plastic model to learn and simulate cloth creeping behaviors.

These cloth parameter estimation methods model a cloth as a continuum sheet which are too coarse to embed cloth micro-mechanics at the yarn-level. As demonstrated in the yarn-level cloth simulation research (Cirio et al. 2014; Cirio et al. 2017), cloths' micro-mechanics have a significant influence on their macro-level behaviors and modeling cloths at the yarn-level can naturally embed cloth material nonlinearity and anisotropy. Therefore, estimating cloth parameters at the yarn-level can more accurately measure the cloth's physical properties. Additionally, the estimated parameters can taken as input by a yarn-level cloth simulator which can simulate more realistic and visually pleasing results than a sheet-level one. Whereas real-

world textile testing is usually conducted at the swatch-level. To the yarn-level cloth simulation and parameter estimation, each cloth swatch is so large that tracking and simulating the mechanical behaviors of all the yarns are computationally intractable. To estimate cloth physical parameters at the yarn-level, Sperl et al. (2022) insert an intermediate sheet-level cloth physical simulator between the real-world cloth testing results and the yarn-level cloth simulator. This sheet-level cloth simulator first fits the real-world testing results by optimizing its cloth physical parameters. Then, cloth's geometry simulated by the sheet-level cloth simulator is approximated by the periodic tilting yarn-level cloth model where the entire cloth is modeled as connected yarn-level cloth patches (Sperl et al. 2020). Next, to estimate cloth yarn-level physical parameters, the yarn-level simulator adjusts its parameters to fit the sheet-level cloth geometry.

Recently, differentiable physics simulation is thriving in solving inverse problems. It relies on fully differentiable physical models to conduct gradient-based optimization, e.g., gradient-descent, and adjust the simulated object's physical parameters. Compared with the conventional machine learning methods, the same as white-box methods, they exhibit outstanding sample efficiency as well. Moreover, compared with gradient-free optimization (such as Powell's method and BO), they exhibit much faster convergence speed and learning accuracy. However, it is usually more difficult to implement because of the requirements on the physical model's differentiability. In addition, physics-based cloth simulators usually involve many non-differentiable functions. For example, differentiable collision handling is one the most challenging problems in differentiable physics because collisions involve instant variations of position and velocity where the gradients are usually undefined (Zhong et al. 2022). Liang et al. (2019) introduce the first differentiable cloth simulator and propose a differentiable collision handling algorithm. This algorithm models the collision response as a constrained optimization problem for correcting the invalid geometrical intersections with a minimum adjustment on the cloth mesh vertices position. To keep the gradient information in this optimization operation, it introduces an efficient differentiable Quadratic Programming method to solve this constrained optimization problem (Amos and Kolter 2017). Compared with the work of Wang et al. (2011), this differentiable cloth simulator shows a superior parameter estimation accuracy and can learn multiple parameters (density, stretching, and bending) in one motion rather than optimizing the parameters individually from isolated deformations. However, this work does not model the stick-slip friction behaviors at the contact points. Li et al. (2022a) use the Signorini-Coulomb friction

model to simulate the stick-slip friction and converts a PD-based physics-based cloth simulator (Ly et al. 2020) to a differentiable cloth simulator. However, the Signorini-Coulomb friction uses a jump function to model stick-slip friction and therefore is non-smooth at the stick-slip transition point where the gradients are undefined. However, this work empirically validates that ignoring the gradients does not stop the differentiable cloth simulator from learning parameters. Moreover, compared with the work of Liang et al. (2019), it can more realistically simulate the frictional behaviors at the contact areas. Nevertheless, the potential of encountering undefined gradients makes this model suspicious. Furthermore, the PD-based physics-based cloth simulation method essentially sacrifices the simulation mechanical accuracy for efficiency. This feature becomes a drawback in differentiable physics for accurately estimating cloth physical parameters. In addition, the methods introduced by Liang et al. (2019) and Li et al. (2022a) need to learn from the precise cloth 3D geometries which is not widely available and difficult to capture in the real world due to the occlusions and 3D reconstruction errors. Instead, Jatavallabhula et al. (2021) append differentiable rendering models to differentiable cloth simulators such that the gradients can propagate from the difference between the rendered images and the captured 2D images to optimize the cloth physical parameters. In this way, this differentiable physics model can learn from 2D images or videos which are cheap and easy to capture.

This section has shown the black methods and white-box methods for cloth physical parameter estimation and highlighted that accurately estimating cloth physical parameters is important to realistic cloth simulations. The black-box methods directly map the observed cloths to the physical parameter without modeling any cloth physics. They usually rely on huge training datasets which are expensive to build. By contrast, the white-box methods are far more data efficient through explicitly modeling cloth physics. In addition, differentiable cloth simulations, a white-box method, are built on fully differentiable cloth physical models so that can leverage gradient-based optimization. Compared with the other white-box methods, it is more accurate and can converge more quickly.

Chapter 3

Fine-grained Differentiable Physics: A Yarn-level Model for Fabrics

Differentiable physics modeling combines physics models with gradient-based learning to provide model explicability and data efficiency. It has been used to learn dynamics, solve inverse problems and facilitate design, and is at its inception of impact. Current successes have concentrated on general physics models such as rigid bodies, continuum deformable sheets, etc, assuming relatively simple structures and forces. Their granularity is intrinsically coarse and therefore incapable of modelling complex physical phenomena. Fine-grained models are still to be developed to incorporate sophisticated material structures and force interactions with gradient-based learning. Following this motivation, we propose a new differentiable model for cloths, where we dive into the granularity of yarns and model individual yarn physics and yarn-to-yarn interactions. To this end, we propose several differentiable forces, whose counterparts in empirical physics are indifferentiable, to facilitate gradient-based learning. Through comprehensive evaluation and comparison, we demonstrate our model's *explicability* in learning meaningful physical parameters, *versatility* in incorporating complex physical structures and heterogeneous materials, *data-efficiency* in learning, and *high-fidelity* in capturing subtle dynamics. Code is available in: Fine-grained Differentiable Physics[1].

[1] https://github.com/realcrane/Fine-grained-Differentiable-Physics-A-Yarn-level-Model-for-Fabrics.git

3.1 Introduction

Differentiable physics models (DPMs) have recently spiked interests, e.g., rigid bodies (Heiden et al. 2020), cloth (Liang et al. 2019), and soft bodies (Hu et al. 2019). The essence of DPMs is making physics models differentiable, so that gradient-based learning can be used to make systems adhere strictly to physical dynamics. This is realized via back-propagation through a series of observed actions, where the system can quickly learn the underlying dynamics. While enjoying neural networks' capability of modeling arbitrary non-linearity, DPMs also improve the model explicability as the learnable model parameters bear physical meanings. As a result, such models provide a new avenue for many applications such as inverse problems, e.g., estimating the mass of a moving rigid body (Belbute-Peres et al. 2018), and control, e.g., learning to shake a bottle to shape the fluid in it (Li et al. 2019).

Early research attempted to model simple and general physical systems such as rigid bodies (Belbute-Peres et al. 2018), followed by a range of systems including deformable objects (Li et al. 2019), cloth (Liang et al. 2019), contacts (Zhong et al. 2021). However, existing models are only generally-purposed which do not consider complex structures/topologies and force interactions. Taking cloth (i.e., fabrics) as an example, existing models (Liang et al. 2019; Li et al. 2019) can learn general cloth dynamics, but only when the cloth is relative simple and homogeneous. Recent research (Wang et al. 2020) has started to explore articulated systems but the model capacity is insufficient to capture the full dynamics of complex systems such as fabrics. Since real-world physical systems (e.g., materials in engineering) often have sophisticated structures and consist of heterogeneous materials, we argue that it is crucial to design fine-grained DPMs, for differentiable physics to be truly applicable and meaningful to real-world applications.

This work focuses on fabrics which are composite materials consisting of basic slim units arranged in different patterns. A common example in fabrics is woven cloth which is made from yarns of different materials (e.g., silk, cotton, nylon) interlaced in various patterns (e.g., plain, satin, twill). Fabrics present new challenges in differentiable modeling. First, the dynamics heterogeneity caused by material and structural diversity needs to be incorporated into modeling, which is especially crucial for solving inverse problems where the physical properties are learned from data. General DPMs without sufficient granularity can only approximate the dynamics and are unable to learn meaningful parameters. Second, certain forces that are essential for

fabrics dynamics are indifferentiable. One such example is friction. The standard Coulomb model for rigid bodies has been made differentiable recently (Belbute-Peres et al. 2018; Zhong et al. 2021). However, it is overly simplified for fabrics because the yarn-to-yarn friction shows richer dynamics (Zhou et al. 2019) that is beyond the capacity of existing methods. Further, the contact modeling together with friction requires new treatments that previous methods did not have to consider.

To overcome these challenges, we propose a new DPM for fabrics at a *more fine-grained* level and apply it to cloth modeling. Unlike general DPMs, we start with a fine-grained yarn-level model. By modeling each yarn individually, we provide the capacity of modeling fabrics with mixed yarns and different woven patterns, which could not be handled previously. To facilitate gradient-based learning, we propose new differentiable forces on/between yarns, including contact, friction and shear. Finally, we incorporate implicit Euler and implicit differentiation to compute gradients induced by an optimization problem embedded in the simulation.

To our best knowledge, our model is the first differentiable physics model which provides sufficient granularity for heterogeneous materials such as fabrics. We comprehensively evaluate its learning capability, data efficiency and fidelity. Since there is no similar model, we compare our model with the most similar work (Liang et al. 2019), which, however, simply models cloth as continuum elastic sheet, and traditional Bayesian optimization on inverse problems. We also compare our work on control learning with popular Reinforcement Learning methods. We show that our model is more explicable, has higher data efficiency, generates more accurate predictions in inverse problem and control learning respectively.

3.2 Related work

Differentiable physics simulator. A differentiable simulator integrates differentiable physics engine into the forward and backward propagation of learning. As a strong inductive bias, these simulation engines increase data efficiency and learning accuracy over gradient-free models. Due to these advantages, differentiable simulation demonstrates superiority in a number of problem domains such as inverse problem, robot control and motion planning. The early works focused initially on simple rigid bodies (Belbute-Peres et al. 2018; Degrave et al. 2019) and later simulation of high degrees of freedom systems, such as fluids (Schenck and Fox 2018), elastic bodies (Hu et al. 2019; Huang et al. 2021), and cloth (Liang et al. 2019). More recently,

Jatavallabhula et al. (2021) introduced an end-to-end differentiable simulator that can learn from images by combining differentiable rendering and differentiable simulation. Comparatively, we explore fine-grained DPMs for composite materials, which leads to new challenges in differentiable modeling.

Cloth simulation. Cloth simulation initially appeared in textile engineering and was then introduced to computer graphics (Long et al. 2011). Cloth has been modeled as particle systems (Breen et al. 1992), mass-spring systems (Provot 1995), and continuum (Narain et al. 2012). Kaldor et al. (2008) proposed a yarn-level knit cloth simulator and found that cloth microstructures have a considerable influence on cloth dynamics. Since then the cloth simulation community has shifted the focus to yarn-level cloth simulation. Based on the objectives, the recent research can be classified to increasing efficiency (Kaldor et al. 2010; Cirio et al. 2017), combining continuum models and yarn-level models, (Casafranca et al. 2020; Sperl et al. 2020) introducing woven cloth simulation (Cirio et al. 2014), and optimizations (Pizana et al. 2020; Sánchez-Banderas et al. 2020). Our work is orthogonal to these papers in that we introduce a new methodology to incorporate differentiable physics into yarn-level models.

Machine learning and cloth simulation. Machine learning was initially introduced to cloth simulation to make data-driven simulators, which have inherent advantages in simulation efficiency over physical-based methods (James and Fatahalian 2003a; Kim and Vendrovsky 2008), and can help improve fidelity (Lähner et al. 2018). In parallel, machine learning has been applied to discover the physical properties from visual information. Bouman et al. (2013) proposed a linear regression model for evaluating cloth density and stiffness from the dynamics of wind-blown cloth. Yang et al. (2017) introduced a neural network for classifying cloths based on how their dynamics are affected by stretching and bending stiffness. Rasheed et al. (2020) proposed a model for estimating the friction coefficient between cloth and other objects. By combining physically-based cloth simulators and neural networks, Runia et al. (2020) estimated cloth parameters by training neural networks to adjust a simulator's parameters so that the simulated cloth mimics the observed one in videos. Different from these gradient-free models, Liang et al. (2019) and Li et al. (2022a) proposed sheet-level differentiable cloth models that can be used to estimate cloth parameters. In this work, we dive into fine-grained physics and propose a new yarn-level differentiable fabrics model which can be embedded into deep neural networks as a layer.

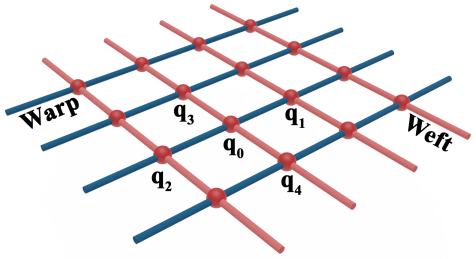


Figure 3.1: Blue and red rods denote warps and wefts respectively. $\mathbf{q}$s are the crossing nodes.

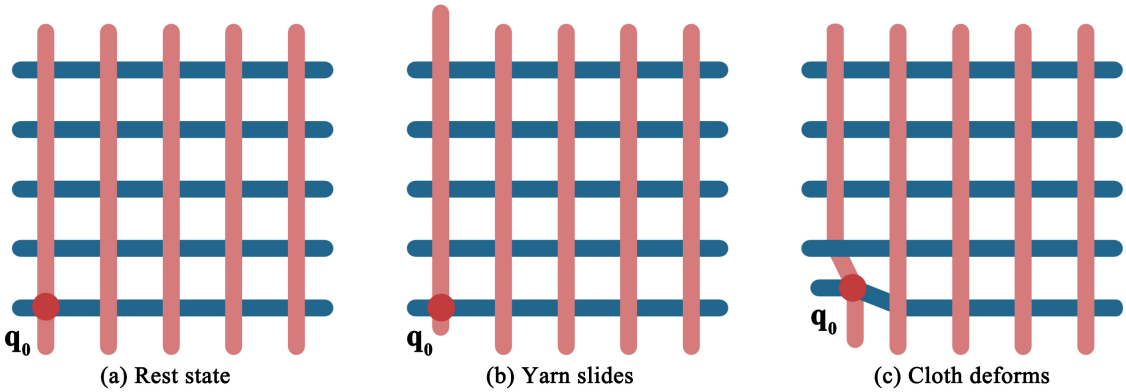


Figure 3.2: (1) Blue and red rods denote warps and wefts respectively. $\mathbf{q}_0$ denotes the highlighted crossing node's position in the Eulerian-on-Lagrangian coordinate. (2) u_0 in $\mathbf{q}_0$ varies when the weft slides. (3) $\mathbf{x}_0$ in $\mathbf{q}_0$ changes when the cross over moves in space, i.e., the simulated cloth deforms.

3.3 Methodology

We first explain the cloth representation (Section 3.3.1) and the (physics) system equation for simulation (Section 3.3.2). Then we present the cloth force models (Section 3.3.3), and how we solve the system equation to enable back-propagation (Section 3.3.4).

3.3.1 Cloth representation

Similar to the simulator proposed by Cirio et al. (2014), our cloth consists of two perpendicular groups of parallel yarns named *warps* and *wefts*. Every pair of warp and weft are in contact with each other at one crossing node (Fig. 3.1), with a persisting contact. We employ an Eulerian-on-Lagrangian discretization (Sueda et al. 2011), and denote the Degrees of Freedom (DoFs) of every crossing node as $\mathbf{q}_i \equiv (\mathbf{x}_i, u_i, v_i)$. $\mathbf{x}_i \in \mathbb{R}^3$ is the Lagrangian coordinates indicating spatial locations and (u_i, v_i) is the Eulerian coordinates indicating sliding movements between yarns, as shown in Fig. 3.2. The end points of yarns do not contact with other yarns and hence they are treated as special crossing nodes that have no Eulerian terms, i.e., $\mathbf{q}_j \equiv \mathbf{x}_i$. Therefore, on

a $r(rows) \times c(columns)$ cloth, there are $(r-2) \times (c-2)$ crossing nodes with five DoFs and $2r+2c-4$ crossing nodes with three DoFs. Every two neighboring crossing nodes on the same warp/weft delimit a warp/weft segment. A warp segment with end points $\mathbf{q}_0$ and $\mathbf{q}_1$ is denoted as $[\mathbf{q_0}, \mathbf{q_1}]$ and its position is $(\mathbf{x}_0, \mathbf{x}_1, u_0, u_1)$, as shown in Fig. 3.1. This way, a woven cloth is discretized into crossing nodes and segments which are the primitive units of the cloth. Every segment is assumed to be straight so that linear interpolation can be employed on the segment, i.e., the spatial location of a point in the segment $[\mathbf{q_0}, \mathbf{q_1}]$ is $\mathbf{x}(u) = \frac{u-u_0}{\Delta u}\mathbf{x}_0 + \frac{u_1-u}{\Delta u}\mathbf{x}_1$, where u is the point's position in Eulerian coordinates and $\Delta u = u_1 - u_0$ is the crossing nodes distance in Eulerian coordinates. We use L to denote the rest length of the yarn segment and R to denote the yarn radius.

3.3.2 System equation for simulation

A cloth's state at time t, $\mathcal{S}_{(t)} = \{\mathcal{Q}_{(t)}, \dot{\mathcal{Q}}_{(t)}\}$, includes all the crossing node DoFs $\mathcal{Q} = \{\mathbf{q}_i | i = 1, 2, \ldots, N\}$ and their velocities $\dot{\mathcal{Q}} = \{\dot{\mathbf{q}}_i | i = 1, 2, \ldots, N\}$, where N is the number of crossing nodes. Knowing the state, we can calculate the internal and external forces:

$$\mathbf{F} = \mathbf{M}\ddot{\mathbf{q}} = \frac{\partial T}{\partial \mathbf{q}} - \frac{\partial V}{\partial \mathbf{q}} - \dot{\mathbf{M}}\dot{\mathbf{q}} \tag{3.1}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are the *general* position, velocity, and acceleration respectively, with a dimension $l = 3 \times r \times c + 2 \times (r-2) \times (c-2)$. $\mathbf{M} \in \mathbb{R}^{l \times l}$ is the general mass matrix. The model assumes mass is distributed homogeneously in one segment, so the mass matrix of a warp segment $[\mathbf{q}_0, \mathbf{q}_1]$ is

$$\mathbf{M}_{0,1} = \frac{1}{6}\Delta u \rho \begin{pmatrix} 2\mathbf{I}_3 & \mathbf{I}_3 & -2\mathbf{w} & -\mathbf{w} \\ \mathbf{I}_3 & 2\mathbf{I}_3 & -\mathbf{w} & -2\mathbf{w} \\ -2\mathbf{w}^\top & -\mathbf{w}^\top & 2\mathbf{w}^\top\mathbf{w} & \mathbf{w}^\top\mathbf{w} \\ -\mathbf{w}^\top & -2\mathbf{w}^\top & \mathbf{w}^\top\mathbf{w} & 2\mathbf{w}^\top\mathbf{w} \end{pmatrix} \tag{3.2}$$

where $\mathbf{w} = \frac{\mathbf{x}_1 - \mathbf{x}_0}{\Delta u}$, and ρ is yarn density. T and V are the kinetic and potential energy. The right hand terms in Eq. (3.1) are positional derivatives of kinetic energy, conservative forces, and part of the time derivative of $\mathbf{M}\dot{\mathbf{q}}$. Non-conservative forces are added to the right side of the equation. Section 3.3.3 gives the details of all the forces considered in our model. To simulate a cloth's motion, the simulator needs to recurrently predicting its future state $\mathcal{S}_{(t+1)}$ given the

current state $\mathcal{S}_{(t)}$:

$$\mathbf{q}_{(t+1)} = \mathbf{q}_{(t)} + h\dot{\mathbf{q}}_{(t)} \tag{3.3}$$

$$\dot{\mathbf{q}}_{(t+1)} = \dot{\mathbf{q}}_{(t)} + h\ddot{\mathbf{q}}_{(t)} \tag{3.4}$$

where h is the time step size (time lapse between every two consecutive states) and the subscript in brackets t indicates the associate variable at time t. To gain high simulation stability, implicit Euler method (Baraff and Witkin 1998) is commonly used:

$$\mathbf{q}_{(t+1)} = \mathbf{q}_{(t)} + h\dot{\mathbf{q}}_{(t+1)} \tag{3.5}$$

$$\dot{\mathbf{q}}_{(t+1)} = \dot{\mathbf{q}}_{(t)} + h\ddot{\mathbf{q}}_{(t+1)} = \dot{\mathbf{q}}_{(t)} + h\mathbf{M}^{-1}\mathbf{F}_{(t+1)} \tag{3.6}$$

Although $\mathbf{F}_{(t+1)}$ is unknown without knowing the future state $\mathcal{S}_{(t+1)}$, it can be approximated by Taylor expansion:

$$\mathbf{F}_{(t+1)} = \mathbf{F}_{(t)} + \frac{\partial \mathbf{F}}{\partial \mathbf{q}}\Delta\mathbf{q} + \frac{\partial \mathbf{F}}{\partial \dot{\mathbf{q}}}\Delta\dot{\mathbf{q}} \tag{3.7}$$

Then,Eq. (3.5) and Eq. (3.6) are converted to the governing equation of the physical system:

$$\left(\mathbf{M} - \frac{\partial \mathbf{F}_{(t)}}{\partial \mathbf{q}}h^2 - \frac{\partial \mathbf{F}_{(t)}}{\partial \dot{\mathbf{q}}}h\right)\dot{\mathbf{q}}_{(t+1)} = h\left(\mathbf{F}_{(t)} - \frac{\partial \mathbf{F}_{(t)}}{\partial \dot{\mathbf{q}}}\dot{\mathbf{q}}_{(t)}\right) + \mathbf{M}\dot{\mathbf{q}}_{(t)} \tag{3.8}$$

which needs to be solved to calculate $\dot{\mathbf{q}}_{(t+1)}$ and update the cloth's state:

$$\dot{\mathbf{q}}_{(t+1)} = \dot{\mathbf{q}}_{(t)} + \Delta\dot{\mathbf{q}} \tag{3.9}$$

$$\mathbf{q}_{(t+1)} = \mathbf{q}_{(t)} + h\dot{\mathbf{q}}_{(t+1)} \tag{3.10}$$

3.3.3 Force models

To simulate cloth, we need to compute $\dot{\mathbf{M}}\dot{\mathbf{q}}$, the positional derivatives of kinetic energy (i.e., $\frac{\partial T}{\partial \mathbf{q}}$), internal and external forces in Eq. (3.8). This section starts from the derivatives of mass matrix $\mathbf{M}$. To the mass matrix of a warp segment $[\mathbf{q}_0, \mathbf{q}_1]$ defined in Eq. (3.2), its partial derivatives with respect to nodes' position is

$$\begin{pmatrix} \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{M}_{0,1}}{\partial u_0} & \frac{\partial \mathbf{M}_{0,1}}{\partial u_1} \end{pmatrix}^\top \tag{3.11}$$

As $\mathbf{x}_0$ and $\mathbf{x}_1$ are vectors:

$$\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0} = \begin{pmatrix} \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}} \\ \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}} \\ \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}} \end{pmatrix} \text{ and } \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1} = \begin{pmatrix} \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(1)}} \\ \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(2)}} \\ \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(3)}} \end{pmatrix} \tag{3.12}$$

The component $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}}$ is

$$\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}} = \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$

and $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}}$, $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}}$, $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(1)}}$, $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(2)}}$ and $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(3)}}$ are in a similar form as $\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}}$. In each term, we have:

$$\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} = -\begin{pmatrix} \frac{1}{\Delta u} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(2)}} = -\begin{pmatrix} 0 \\ \frac{1}{\Delta u} \\ 0 \end{pmatrix}, \text{ and } \frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(3)}} = -\begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u} \end{pmatrix}$$

$$\frac{\partial \mathbf{w}}{\partial \mathbf{x}_1^{(1)}} = \begin{pmatrix} \frac{1}{\Delta u} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial \mathbf{w}}{\partial \mathbf{x}_1^{(2)}} = \begin{pmatrix} 0 \\ \frac{1}{\Delta u} \\ 0 \end{pmatrix}, \text{ and } \frac{\partial \mathbf{w}}{\partial \mathbf{x}_1^{(3)}} = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u} \end{pmatrix}$$

$$\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} = \frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}}\mathbf{w} + \mathbf{w}^\top \frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} = -2\frac{\mathbf{x}_1^{(1)} - \mathbf{x}_0^{(1)}}{\Delta u^2}$$

where $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(2)}}$, $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(3)}}$, $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_1^{(1)}}$, $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_1^{(2)}}$ and $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_1^{(3)}}$ have a similar form as $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}}$.

Unsurprisingly, we can find that

$$\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}} = -\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(1)}}, \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}} = -\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(2)}} \text{ and } \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}} = -\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(3)}}$$

After deriving the partial derivatives of $\mathbf{M}_{0,1}$ with respect to the Lagrangian coordinates, we

give its partial derivatives with respect to Eulerian coordinates:

$$\begin{aligned}\frac{\partial \mathbf{M}_{0,1}}{\partial u_0} =& -\frac{1}{6}\rho \begin{pmatrix} 2\mathbf{I}_3 & \mathbf{I}_3 & -2\mathbf{w} & -\mathbf{w} \\ \mathbf{I}_3 & 2\mathbf{I}_3 & -\mathbf{w} & -2\mathbf{w} \\ -2\mathbf{w}^\top & -\mathbf{w}^\top & 2\mathbf{w}^\top\mathbf{w} & \mathbf{w}^\top\mathbf{w} \\ -\mathbf{w}^\top & -2\mathbf{w}^\top & \mathbf{w}^\top\mathbf{w} & 2\mathbf{w}^\top\mathbf{w} \end{pmatrix} \\ &+ \frac{1}{6}\Delta u\rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial u_0} & -\frac{\partial \mathbf{w}}{\partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial u_0} & -2\frac{\partial \mathbf{w}}{\partial u_0} \\ -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & -\frac{\partial \mathbf{w}^\top}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_0} & \frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_0} \\ -\frac{\partial \mathbf{w}^\top}{\partial u_0} & -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & \frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_0} \end{pmatrix}\end{aligned} \tag{3.13}$$

where $\frac{\partial \mathbf{M}_{0,1}}{\partial u_1}$ has a similar form as $\frac{\partial \mathbf{M}_{0,1}}{\partial u_0}$ and:

$$\frac{\partial \mathbf{w}}{\partial u_0} = \frac{\mathbf{x}_1 - \mathbf{x}_0}{\Delta u^2} = \frac{\mathbf{w}}{\Delta u} \text{ and } \frac{\partial \mathbf{w}}{\partial u_1} = -\frac{\mathbf{x}_1 - \mathbf{x}_0}{\Delta u^2} = -\frac{\mathbf{w}}{\Delta u}$$

$$\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_0} = \frac{\partial \mathbf{w}^\top}{\partial u_0}\mathbf{w} + \mathbf{w}^\top\frac{\partial \mathbf{w}}{\partial u_0} = 2\frac{\mathbf{w}^\top\mathbf{w}}{\Delta u}$$

$$\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial u_1} = \frac{\partial \mathbf{w}^\top}{\partial u_1}\mathbf{w} + \mathbf{w}^\top\frac{\partial \mathbf{w}}{\partial u_1} = -2\frac{\mathbf{w}^\top\mathbf{w}}{\Delta u}$$

Likewise, we can find

$$\frac{\partial \mathbf{M}_{0,1}}{\partial u_0} = -\frac{\partial \mathbf{M}_{0,1}}{\partial u_1}$$

So far, we have given the full details of $\mathbf{M}_{0,1}$' s partial derivatives with respect to positions in

Eq. (3.11). Now we give its time derivative:

$$\dot{\mathbf{M}}_{0,1} = \frac{1}{6}\rho(\dot{u}_1 - \dot{u}_0)\begin{pmatrix} 2\mathbf{I}_3 & \mathbf{I}_3 & -2\mathbf{w} & -\mathbf{w} \\ \mathbf{I}_3 & 2\mathbf{I}_3 & -\mathbf{w} & -2\mathbf{w} \\ -2\mathbf{w}^\top & -\mathbf{w}^\top & 2\mathbf{w}^\top\mathbf{w} & \mathbf{w}^\top\mathbf{w} \\ -\mathbf{w}^\top & -2\mathbf{w}^\top & \mathbf{w}^\top\mathbf{w} & 2\mathbf{w}^\top\mathbf{w} \end{pmatrix} + \frac{1}{6}\rho\Delta u \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial t} & -\frac{\partial \mathbf{w}}{\partial t} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial t} & -2\frac{\partial \mathbf{w}}{\partial t} \\ -2\frac{\partial \mathbf{w}^\top}{\partial t} & -\frac{\partial \mathbf{w}^\top}{\partial t} & 2\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial t} & \frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial t} \\ -\frac{\partial \mathbf{w}^\top}{\partial t} & -2\frac{\partial \mathbf{w}^\top}{\partial t} & \frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial t} & 2\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial t} \end{pmatrix} \tag{3.14}$$

where

$$\frac{\partial \mathbf{w}}{\partial t} = \frac{\partial}{\partial t}\frac{\mathbf{x}_1 - \mathbf{x}_0}{\Delta u} = \frac{(\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_0)\Delta u - (\mathbf{x}_1 - \mathbf{x}_0)(\dot{u}_1 - \dot{u}_0)}{\Delta u^2}$$
$$\frac{\partial \mathbf{w}^\top\mathbf{w}}{\partial t} = \frac{\partial \mathbf{w}^\top}{\partial t}\mathbf{w} + \mathbf{w}^\top\frac{\partial \mathbf{w}}{\partial t}$$

In addition, the derivatives of $\dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}$ with respect to the nodes' positions are:

$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \mathbf{x}_0} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_0^{(1)}}\dot{\mathbf{q}}_{0,1} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_0^{(2)}}\dot{\mathbf{q}}_{0,1} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_0^{(3)}}\dot{\mathbf{q}}_{0,1} \end{pmatrix}, \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \mathbf{x}_1} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_1^{(1)}}\dot{\mathbf{q}}_{0,1} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_1^{(2)}}\dot{\mathbf{q}}_{0,1} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_1^{(3)}}\dot{\mathbf{q}}_{0,1} \end{pmatrix}$$
$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial u_0} = \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial u_0}\dot{\mathbf{q}}_{0,1}, \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial u_1} = \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial u_1}\dot{\mathbf{q}}_{0,1} \tag{3.15}$$

The components in Eq. (3.15) are:

$$\frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \mathbf{x}_0^{(1)}} = \frac{1}{6}\rho(\dot{u}_1 - \dot{u}_0)\begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$
$$+ \frac{1}{6}\rho\Delta u \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$

and

$$\frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial u_0} = \frac{1}{6}\rho(\dot{u}_1 - \dot{u}_0)\begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial u_0} & -\frac{\partial \mathbf{w}}{\partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial u_0} & -2\frac{\partial \mathbf{w}}{\partial u_0} \\ -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & -\frac{\partial \mathbf{w}^\top}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \\ -\frac{\partial \mathbf{w}^\top}{\partial u_0} & -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \end{pmatrix}$$
$$- \frac{1}{6}\rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial t} & -\frac{\partial \mathbf{w}}{\partial t} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial t} & -2\frac{\partial \mathbf{w}}{\partial t} \\ -2\frac{\partial \mathbf{w}^\top}{\partial t} & -\frac{\partial \mathbf{w}^\top}{\partial t} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial t} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial t} \\ -\frac{\partial \mathbf{w}^\top}{\partial t} & -2\frac{\partial \mathbf{w}^\top}{\partial t} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial t} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial t} \end{pmatrix}$$
$$+ \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial u_0} & -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial u_0} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial u_0} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial u_0} \end{pmatrix}$$

where

$$\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} = \begin{pmatrix} \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(2)}} = \begin{pmatrix} 0 \\ \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \\ 0 \end{pmatrix}, \text{ and } \frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(2)}} = \begin{pmatrix} 0 \\ 0 \\ \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_1^{(1)}} = -\begin{pmatrix} \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_1^{(2)}} = -\begin{pmatrix} 0 \\ \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \\ 0 \end{pmatrix}, \text{ and } \frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_1^{(2)}} = -\begin{pmatrix} 0 \\ 0 \\ \frac{\dot{u}_1 - \dot{u}_0}{\Delta u^2} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} = \frac{\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_1}{(u_1 - u_0)^2} - \frac{2(\mathbf{x}_1 - \mathbf{x}_0)(\dot{u}_1 - \dot{u}_0)}{(u_1 - u_0)^3}$$

$$\frac{\partial^2 \mathbf{w}}{\partial t \partial u_1} = -\frac{\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_1}{(u_1 - u_0)^2} + \frac{2(\mathbf{x}_1 - \mathbf{x}_0)(\dot{u}_1 - \dot{u}_0)}{(u_1 - u_0)^3}$$

$$\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} = \frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_1^{(1)}} \frac{\partial \mathbf{w}}{\partial t} + \mathbf{w}^\top \frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_1^{(1)}} + \frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_1^{(1)}} \mathbf{w} + \frac{\partial \mathbf{w}^\top}{\partial t} \frac{\partial \mathbf{w}}{\partial \mathbf{x}_1^{(1)}}$$

$$\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial u_0} = \frac{\partial \mathbf{w}^\top}{\partial u_0} \frac{\partial \mathbf{w}}{\partial t} + \mathbf{w}^\top \frac{\partial^2 \mathbf{w}}{\partial t \partial u_0} + \frac{\partial^2 \mathbf{w}^\top}{\partial t \partial u_0} \mathbf{w} + \frac{\partial \mathbf{w}^\top}{\partial t} \frac{\partial \mathbf{w}}{\partial u_0}$$

The derivatives of $\dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}$ with respect to the nodes' velocities are:

$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(2)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(3)}} \end{pmatrix} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(2)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(2)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(3)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(3)}} \end{pmatrix}$$

$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(2)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(3)}} \end{pmatrix} = \begin{pmatrix} \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(2)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(2)}} \\ \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(3)}} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(3)}} \end{pmatrix}$$

$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_0} = \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{u}_0} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_0}$$

$$\frac{\partial \dot{\mathbf{M}}_{0,1}\dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_1} = \frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{u}_1} \dot{\mathbf{q}}_{0,1} + \dot{\mathbf{M}}_{0,1} \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_1} \tag{3.16}$$

where

$$\frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} = \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \mathbf{x}_0^{(1)}} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$

$$\frac{\partial \dot{\mathbf{M}}_{0,1}}{\partial \dot{u}_0} = -\frac{1}{6}\rho \begin{pmatrix} 2\mathbf{I}_3 & \mathbf{I}_3 & -2\mathbf{w} & -\mathbf{w} \\ \mathbf{I}_3 & 2\mathbf{I}_3 & -\mathbf{w} & -2\mathbf{w} \\ -2\mathbf{w}^\top & -\mathbf{w}^\top & 2\mathbf{w}^\top \mathbf{w} & \mathbf{w}^\top \mathbf{w} \\ -\mathbf{w}^\top & -2\mathbf{w}^\top & \mathbf{w}^\top \mathbf{w} & 2\mathbf{w}^\top \mathbf{w} \end{pmatrix}$$
$$+ \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \dot{u}_0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \dot{u}_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial t \partial \dot{u}_0} & -2\frac{\partial^2 \mathbf{w}}{\partial t \partial \dot{u}_0} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \dot{u}_0} & -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \dot{u}_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \dot{u}_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \dot{u}_0} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \dot{u}_0} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial t \partial \dot{u}_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \dot{u}_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial t \partial \dot{u}_0} \end{pmatrix}$$

$$\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \text{ and } \frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial u_0} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Kinetic energy is computed segment-wise, e.g., for a segment $[\mathbf{q}_0, \mathbf{q}_1]$:

$$T_{0,1} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \mathbf{M}_{0,1} \dot{\mathbf{q}}_{0,1} = \frac{1}{2}\begin{pmatrix} \dot{\mathbf{x}}_0^\top & \dot{\mathbf{x}}_1^\top & \dot{u}_0 & \dot{u}_1 \end{pmatrix} \mathbf{M}_{0,1} \begin{pmatrix} \dot{\mathbf{x}}_0 \\ \dot{\mathbf{x}}_1 \\ \dot{u}_0 \\ \dot{u}_1 \end{pmatrix} \tag{3.17}$$

Its derivative with respect to each node's position is:

$$\frac{\partial T_{0,1}}{\partial \mathbf{q}_{0,1}} = \begin{pmatrix} \mathbf{F}_{\mathbf{x}_0}^{(k)} \\ \mathbf{F}_{\mathbf{x}_1}^{(k)} \\ F_{u_0}^{(k)} \\ F_{u_1}^{(k)} \end{pmatrix} \tag{3.18}$$

$$\begin{aligned} \mathbf{F}_{\mathbf{x}_0}^{(k)} &= \frac{\partial T_{0,1}}{\partial \mathbf{x}_0} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} \\ \mathbf{F}_{\mathbf{x}_1}^{(k)} &= \frac{\partial T_{0,1}}{\partial \mathbf{x}_1} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} \\ F_{u_0}^{(k)} &= \frac{\partial T_{0,1}}{\partial u_0} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial u_0}\dot{\mathbf{q}}_{0,1} \\ F_{u_1}^{(k)} &= \frac{\partial T_{0,1}}{\partial u_1} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial u_1}\dot{\mathbf{q}}_{0,1} \end{aligned}$$

where $\mathbf{F}_{x_0}^{(k)}$ and $F_{u_0}^{(k)}$ are the inertia of $\mathbf{q}_0$ in Lagrangian and Eulerian coordinates respectively. Similarly, $\mathbf{F}_{x_1}^{(k)}$ and $F_{u_1}^{(k)}$ are the inertia of $\mathbf{q}_1$.The derivative of the forces with respect to positions are:

$$\frac{\partial^2 T_{0,1}}{\partial \mathbf{q}_{0,1}\partial \mathbf{q}_{0,1}} = \begin{pmatrix} \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_0} & \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_1} \\ \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_0} & \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_1} \\ \frac{\partial \mathbf{F}_{u_0}^{(k)}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{F}_{u_0}^{(k)}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{F}_{u_0}^{(k)}}{\partial u_0} & \frac{\partial \mathbf{F}_{u_0}^{(k)}}{\partial u_1} \\ \frac{\partial \mathbf{F}_{u_1}^{(k)}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{F}_{u_1}^{(k)}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{F}_{u_1}^{(k)}}{\partial u_0} & \frac{\partial \mathbf{F}_{u_1}^{(k)}}{\partial u_1} \end{pmatrix} \tag{3.19}$$

The derivative of the force in Lagrangian coordinates with respect to Lagrangian coordinates is

$$\begin{aligned} \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0} &= \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} \\ &= \frac{1}{2}\begin{pmatrix} \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}\partial \mathbf{x}_0^{(1)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}\partial \mathbf{x}_0^{(2)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)}\partial \mathbf{x}_0^{(3)}}\dot{\mathbf{q}}_{0,1} \\ \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}\partial \mathbf{x}_0^{(1)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}\partial \mathbf{x}_0^{(2)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)}\partial \mathbf{x}_0^{(3)}}\dot{\mathbf{q}}_{0,1} \\ \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}\partial \mathbf{x}_0^{(1)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}\partial \mathbf{x}_0^{(2)}}\dot{\mathbf{q}}_{0,1} & \dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)}\partial \mathbf{x}_0^{(3)}}\dot{\mathbf{q}}_{0,1} \end{pmatrix} \end{aligned}$$

$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_1}$, $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_0}$ and $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_1}$ are in similar forms as $\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0}$. Also, the derivative of the force in Lagrangian coordinate with respect to Eulerian coordinates is:

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_0} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial u_0}\dot{\mathbf{q}}_{0,1}$$

$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_1}$, $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_0}$ and $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_1}$ are in similar forms as $\frac{\partial \mathbf{F}_{\mathbf{x}_0}}{\partial u_0}$. Correspondingly, the derivative of the force in Eulerian coordinates with respect to Lagrangian coordinates is:

$$\frac{\partial F_{u_0}^{(k)}}{\partial \mathbf{x}_0} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1}$$

$\frac{\partial F_{u_0}^{(k)}}{\partial \mathbf{x}_1}$, $\frac{\partial F_{u_1}^{(k)}}{\partial \mathbf{x}_0}$ and $\frac{\partial F_{u_1}^{(k)}}{\partial \mathbf{x}_1}$ are in similar forms as $\frac{\partial F_{u_0}^{(k)}}{\partial \mathbf{x}_0}$. The derivative of the force in Eulerian coordinates with respect to Eulerian coordinates is:

$$\frac{\partial F_{u_0}^{(k)}}{\partial u_0} = \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial u_0}\dot{\mathbf{q}}_{0,1}$$

$\frac{\partial F_{u_0}^{(k)}}{\partial u_1}$, $\frac{\partial F_{u_1}^{(k)}}{\partial u_0}$ and $\frac{\partial F_{u_1}^{(k)}}{\partial u_1}$ are in similar forms as $\frac{\partial F_{u_0}^{(k)}}{\partial u_0}$.

Specially, the entries in $\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0}$ are:

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} = \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} = \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} \\ \mathbf{0} & \mathbf{0} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} = \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} \\ \mathbf{0} & \mathbf{0} & \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} & 2\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} \end{pmatrix}$$

where

$$\frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} = \frac{2}{\Delta u^2}\ ,\ \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} = \frac{2}{\Delta u^2}\ ,\text{and}\ \frac{\partial^2 \mathbf{w}^{\top}\mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} = \frac{2}{\Delta u^2}$$

The other components are

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(2)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(3)}} = \mathbf{0}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(1)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(3)}} = \mathbf{0}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(1)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(2)}} = \mathbf{0}$$

Moreover, as

$$\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_1^{(1)}} = -\frac{2}{\Delta u^2} \text{ , } \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_1^{(2)}} = -\frac{2}{\Delta u^2} \text{ ,and } \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_1^{(3)}} = -\frac{2}{\Delta u^2}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_1^{(2)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_1^{(3)}} = \mathbf{0}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_1^{(1)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_1^{(3)}} = \mathbf{0}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_1^{(1)}} = \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_1^{(2)}} = \mathbf{0}$$

We can find that

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_0^{(1)}} = -\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial \mathbf{x}_1^{(1)}}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_0^{(2)}} = -\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial \mathbf{x}_1^{(2)}}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_0^{(3)}} = -\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial \mathbf{x}_1^{(3)}}$$

Therefore,

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0} = -\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_1}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_1} = -\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \mathbf{x}_0}$$

To compute the derivatives of the forces in Lagrangian coordinates with respect to Eulerian coordinates, we need to compute:

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial u_0} = \begin{pmatrix} \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial u_0} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial u_0} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial u_0} \end{pmatrix} \text{ and } \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial u_1} = \begin{pmatrix} \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial u_1} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(2)} \partial u_1} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(3)} \partial u_1} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1 \partial u_0} = \begin{pmatrix} \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(1)} \partial u_0} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(2)} \partial u_0} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(3)} \partial u_0} \end{pmatrix} \text{ and } \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1 \partial u_1} = \begin{pmatrix} \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(1)} \partial u_1} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(2)} \partial u_1} \\ \frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1^{(3)} \partial u_1} \end{pmatrix}$$

Take one component $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial u_0}$ as example:

$$\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0^{(1)} \partial u_0} = -\frac{1}{6} \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \\ -\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & -2\frac{\partial \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)}} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)}} \end{pmatrix}$$
$$+ \frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} & -\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} & -2\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)} \partial u_0} & -\frac{\partial^2 \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)} \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)} \partial u_0} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial \mathbf{x}_0^{(1)} \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} \end{pmatrix}$$

in which

$$\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_0} = -\begin{pmatrix} \frac{1}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial u_0} = -\begin{pmatrix} 0 \\ \frac{1}{\Delta u^2} \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial u_0} = -\begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u^2} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(1)} \partial u_1} = \begin{pmatrix} \frac{1}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(2)} \partial u_1} = \begin{pmatrix} 0 \\ \frac{1}{\Delta u^2} \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_0^{(3)} \partial u_1} = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u^2} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(1)} \partial u_0} = -\begin{pmatrix} \frac{1}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(2)} \partial u_0} = -\begin{pmatrix} 0 \\ \frac{1}{\Delta u^2} \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(3)} \partial u_0} = -\begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u^2} \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(1)} \partial u_0} = \begin{pmatrix} \frac{1}{\Delta u^2} \\ 0 \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(2)} \partial u_0} = \begin{pmatrix} 0 \\ \frac{1}{\Delta u^2} \\ 0 \end{pmatrix}, \frac{\partial^2 \mathbf{w}}{\partial \mathbf{x}_1^{(3)} \partial u_0} = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\Delta u^2} \end{pmatrix}$$

It should be easy to compute $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial u_0}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_0 \partial u_1}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1 \partial u_0}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial \mathbf{x}_1 \partial u_1}$ and then compute $\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_0}$, $\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_1}$, $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_0}$, $\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_1}$. The derivatives of the forces in Eulerain coordinates with respect to Lagrangian coordinates are the transpose of the forces in Lagrangian coordinates with respect to Eulerian coordinates:

$$\frac{\partial F_{u_0}^{(k)}}{\partial \mathbf{x}_0} = \left(\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_0} \right)^\top \quad \frac{\partial F_{u_0}^{(k)}}{\partial \mathbf{x}_1} = \left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_0} \right)^\top$$
$$\frac{\partial F_{u_1}^{(k)}}{\partial \mathbf{x}_0} = \left(\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial u_1} \right)^\top \quad \frac{\partial F_{u_1}^{(k)}}{\partial \mathbf{x}_1} = \left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial u_1} \right)^\top$$

To compute the derivatives of the forces in Eluerian coordinates with respect to Eulerian coordinates, we need to compute: $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial u_0}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial u_1}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_1 \partial u_0}$, and $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_1 \partial u_1}$. For example,

$$\begin{aligned}
\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial u_0} =& -\frac{1}{6}\rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial u_0} & -\frac{\partial \mathbf{w}}{\partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial u_0} & -2\frac{\partial \mathbf{w}}{\partial u_0} \\ -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & -\frac{\partial \mathbf{w}^\top}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \\ -\frac{\partial \mathbf{w}^\top}{\partial u_0} & -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \end{pmatrix} \\
& -\frac{1}{6}\rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial \mathbf{w}}{\partial u_0} & -\frac{\partial \mathbf{w}}{\partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial \mathbf{w}}{\partial u_0} & -2\frac{\partial \mathbf{w}}{\partial u_0} \\ -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & -\frac{\partial \mathbf{w}^\top}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \\ -\frac{\partial \mathbf{w}^\top}{\partial u_0} & -2\frac{\partial \mathbf{w}^\top}{\partial u_0} & \frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} & 2\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial u_0} \end{pmatrix} \\
& +\frac{1}{6}\Delta u \rho \begin{pmatrix} \mathbf{0} & \mathbf{0} & -2\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_0} & -\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_0} \\ \mathbf{0} & \mathbf{0} & -\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_0} & -2\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_0} \\ -2\frac{\partial^2 \mathbf{w}^\top}{\partial u_0 \partial u_0} & -\frac{\partial^2 \mathbf{w}^\top}{\partial u_0 \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_0} \\ -\frac{\partial^2 \mathbf{w}^\top}{\partial u_0 \partial u_0} & -2\frac{\partial^2 \mathbf{w}^\top}{\partial u_0 \partial u_0} & \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_0} & 2\frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_0} \end{pmatrix}
\end{aligned}$$

in which

$$\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_0} = 2\frac{\mathbf{w}}{\Delta u^2} \text{ and } \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_0} = 6\frac{\mathbf{w}^\top \mathbf{w}}{\Delta u^2}$$

Similarly, we can compute

$$\frac{\partial^2 \mathbf{w}}{\partial u_0 \partial u_1} = -2\frac{\mathbf{w}}{\Delta u^2} \quad \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_0 \partial u_1} = -6\frac{\mathbf{w}^\top \mathbf{w}}{\Delta u^2}$$

$$\frac{\partial^2 \mathbf{w}}{\partial u_1 \partial u_0} = -2\frac{\mathbf{w}}{\Delta u^2} \quad \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_1 \partial u_0} = -6\frac{\mathbf{w}^\top \mathbf{w}}{\Delta u^2}$$

$$\frac{\partial^2 \mathbf{w}}{\partial u_1 \partial u_1} = 2\frac{\mathbf{w}}{\Delta u^2} \quad \frac{\partial^2 \mathbf{w}^\top \mathbf{w}}{\partial u_1 \partial u_1} = 6\frac{\mathbf{w}^\top \mathbf{w}}{\Delta u^2}$$

to compute $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_0 \partial u_1}$, $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_1 \partial u_0}$, and $\frac{\partial^2 \mathbf{M}_{0,1}}{\partial u_1 \partial u_1}$. The derivatives of the inertia with respect to nodes' velocities are:

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_0}}{\partial \dot{\mathbf{x}}_0} = \begin{pmatrix} \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(1)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(2)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(2)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(3)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(3)}} \end{pmatrix}$$

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_0}}{\partial \dot{\mathbf{x}}_1} = \begin{pmatrix} \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(1)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(2)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(2)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(3)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(3)}} \end{pmatrix}$$

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_1}}{\partial \dot{\mathbf{x}}_0} = \begin{pmatrix} \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(1)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(1)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(2)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(2)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_0^{(3)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_0^{(3)}} \end{pmatrix}$$

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_1}}{\partial \dot{\mathbf{x}}_1} = \begin{pmatrix} \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(1)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(1)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(2)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(2)}} \\ \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{\mathbf{x}}_1^{(3)}}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{\mathbf{x}}_1^{(3)}} \end{pmatrix}$$

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_0}}{\partial \dot{u}_0} = \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{u}_0}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_0}$$

$$\frac{\partial \mathbf{F}^{(k)}_{\mathbf{x}_1}}{\partial \dot{u}_0} = \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^\top}{\partial \dot{u}_0}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^\top \frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_0}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(k)}}{\partial \dot{u}_1} = \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^{\top}}{\partial \dot{u}_1}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_0}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_1}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(k)}}{\partial \dot{u}_1} = \frac{1}{2}\frac{\partial \dot{\mathbf{q}}_{0,1}^{\top}}{\partial \dot{u}_1}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\dot{\mathbf{q}}_{0,1} + \frac{1}{2}\dot{\mathbf{q}}_{0,1}^{\top}\frac{\partial \mathbf{M}_{0,1}}{\partial \mathbf{x}_1}\frac{\partial \dot{\mathbf{q}}_{0,1}}{\partial \dot{u}_1}$$

As woven cloths are interlaced yarns, the internal forces can be further classified into 1) forces caused by yarn deformation and 2) forces resulting from yarn-to-yarn interactions. We treat each yarn as an elastic rod that can generate elastic energy, including stretching and bending (Jawed et al. 2018), ignoring the twisting due to its triviality in cloth dynamics (Cirio et al. 2014). The elastic energy is $V^{elastic} = V^{(stretch)} + V^{(bend)}$, where $V^{(stretch)}$ and $V^{(bend)}$ are the stretching and bending energy respectively.

Stretch force resists length changes of segments (with the rest length $\|\mathbf{w}\| = 1$). Therefore, the stretching energy is generated when the length changes. We compute the energy of segment $[\mathbf{q_0}, \mathbf{q_1}]$, in a similar way as (Loock et al. 2001; Spillmann and Teschner 2007):

$$V_{0,1}^{(stretch)} = \frac{1}{2}Y\pi R^2\Delta u(\|\mathbf{w}\| - 1)^2 \tag{3.20}$$

where Y is yarn's elastic modulus and R is yarns' radius. The stretching forces at the two nodes are:

$$\mathbf{F}_{x_1}^{(stretch)} = -\mathbf{F}_{x_0}^{(stretch)} = -\frac{\partial V_{0,1}}{\partial \mathbf{x}_1} = -Y\pi R^2(\|\mathbf{w}\| - 1)\mathbf{d}_{0,1} \tag{3.21}$$

$$F_{u_1}^{(stretch)} = -F_{u_0}^{(stretch)} = -\frac{\partial V_{0,1}}{\partial u_1} = \frac{1}{2}Y\pi R^2(\|\mathbf{w}\|^2 - 1) \tag{3.22}$$

where $\mathbf{d}_{0,1}$ is the unit vector points from $\mathbf{q}_0$ to $\mathbf{q}_1$, $\mathbf{d}_{0,1} = \frac{\mathbf{x}_1 - \mathbf{x}_0}{\|\mathbf{x}_1 - \mathbf{x}_0\|}$. The derivatives of the stretching forces with respect to nodes' positions are:

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(stretch)}}{\partial \mathbf{x}_1} = \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(stretch)}}{\partial \mathbf{x}_0} = -\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(stretch)}}{\partial \mathbf{x}_0} = -\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(stretch)}}{\partial \mathbf{x}_1} = Y\pi R^2(\frac{1}{l_1}\mathbf{P}_{0,1} - \frac{1}{\Delta u}\mathbf{I}) \tag{3.23}$$

$$\frac{\partial F_{u_1}^{(stretch)}}{\partial u_1} = \frac{\partial F_{u_0}^{(stretch)}}{\partial u_0} = -\frac{\partial F_{u_1}^{(stretch)}}{\partial u_0} = -\frac{\partial F_{u_0}^{(stretch)}}{\partial u_1} = -Y\pi R^2\frac{\|\mathbf{w}\|^2}{\Delta u} \tag{3.24}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(stretch)}}{\partial u_1} = \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(stretch)}}{\partial u_0} = -\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(stretch)}}{\partial u_0} = -\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(stretch)}}{\partial u_1} = Y\pi R^2\frac{\|\mathbf{w}\|^2}{\Delta u}\mathbf{d}_{0,1} \tag{3.25}$$

$$\frac{\partial F_{u_1}^{(stretch)}}{\partial \mathbf{x}_1} = \frac{\partial F_{u_0}^{(stretch)}}{\partial \mathbf{x}_0} = -\frac{\partial F_{u_1}^{(stretch)}}{\partial \mathbf{x}_0} = -\frac{\partial F_{u_0}^{(stretch)}}{\partial \mathbf{x}_1} = \frac{Y\pi R^2}{\Delta u}\mathbf{w}^{\top} \tag{3.26}$$

where $\mathbf{P}_{0,1} = \mathbf{I}_3 - \mathbf{d}_{0,1}\mathbf{d}_{0,1}^{\top}$.

Bending energy is defined as the integration of bending energy density along the two segments. The bending energy on the two connected warp segments $[\mathbf{q}_2, \mathbf{q}_0]$ and $[\mathbf{q}_0, \mathbf{q}_1]$ is

$$V_{2,0,1}^{(bend)} = B\pi R^2 \frac{\theta^2}{u_1 - u_2} \tag{3.27}$$

where B is yarn bending modulus and $\theta = \arcsin(-\mathbf{d}_{0,1}^\top \mathbf{d}_{0,2})$ is the angle between the two segments. Its derivatives with respective to the node position are the bending forces:

$$\mathbf{F}_{\mathbf{x}_1}^{(bend)} = -\frac{2B\pi R^2\theta}{l_1(u_1 - u_2)\sin\theta}\mathbf{P}_{0,1}\mathbf{d_{0,2}} \tag{3.28}$$

$$\mathbf{F}_{\mathbf{x}_2}^{(bend)} = -\frac{(2B\pi R^2\theta)}{l_2(u_1 - u_2)\sin\theta}\mathbf{P}_{0,2}\mathbf{d_{0,1}} \tag{3.29}$$

$$\mathbf{F}_{\mathbf{x}_0}^{(bend)} = -(\mathbf{F}_{\mathbf{x}_1}^{(bend)} + \mathbf{F}_{\mathbf{x}_2}^{(bend)}) \tag{3.30}$$

$$F_{u_1}^{(bend)} = -F_{u_2}^{(bend)} = \frac{2B\pi R^2\theta^2}{(u_1 - u_2)^2} \tag{3.31}$$

$$F_{u_0}^{(bend)} = 0 \tag{3.32}$$

The derivatives of the bending forces with respected to the nodes' position are

$$\begin{aligned}\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_1} =& \frac{2B\pi R^2}{l_1^2(u_1 - u_0)\sin\theta}\Bigg(\theta\Bigg(\mathbf{P}_{0,1}\mathbf{d}_{0,2}\mathbf{d}_{0,1}^\top + \frac{\cos\theta}{\sin^2\theta}\mathbf{P}_{0,1}\mathbf{d}_{0,2}\mathbf{d}_{0,2}^\top\mathbf{P}_{0,1} + \cos\theta\mathbf{P}_{0,1} \\ &+ \mathbf{d}_{0,1}\mathbf{d}_{0,2}^\top\mathbf{P}_{0,1}\Bigg) - \frac{1}{\sin\theta}\mathbf{P}_{0,1}\mathbf{d}_{0,2}\mathbf{d}_{0,2}^\top\mathbf{P}_{0,1}\Bigg)\end{aligned} \tag{3.33}$$

$$\begin{aligned}\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_2} =& \frac{2B\pi R^2}{l_2^2(u_1 - u_0)\sin\theta}\Bigg(\theta\Bigg(\mathbf{P}_{0,2}\mathbf{d}_{0,1}\mathbf{d}_{0,2}^\top + \frac{\cos\theta}{\sin^2\theta}\mathbf{P}_{0,2}\mathbf{d}_{0,1}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,2} + \cos\theta\mathbf{P}_{0,2} \\ &+ \mathbf{d}_{0,2}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,2}\Bigg) - \frac{1}{\sin\theta}\mathbf{P}_{0,2}\mathbf{d}_{0,1}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,2}\Bigg)\end{aligned} \tag{3.34}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_2} = -\frac{2B\pi R^2}{l_2 l_1(u_1 - u_2)\sin\theta}\left(\theta\left(\mathbf{P}_{0,1} - \frac{\cos\theta}{\sin^2\theta}\mathbf{P}_{0,1}\mathbf{d}_{0,2}\mathbf{d}_{0,1}^\top\right) + \frac{1}{\sin\theta}\mathbf{P}_{0,1}\mathbf{d}_{0,2}\mathbf{d}_{0,1}^\top\right)\mathbf{P}_{0,2} \tag{3.35}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_1} = -\frac{2B\pi R^2}{l_1 l_2(u_1 - u_2)\sin\theta}\left(\theta\left(\mathbf{P}_{0,2} - \frac{\cos\theta}{\sin^2\theta}\mathbf{P}_{0,2}\mathbf{d}_{0,1}\mathbf{d}_{0,2}^\top\right) + \frac{1}{\sin\theta}\mathbf{P}_{0,2}\mathbf{d}_{0,1}\mathbf{d}_{0,2}^\top\right)\mathbf{P}_{0,1} \tag{3.36}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_1} + \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_2}\right) \tag{3.37}$$

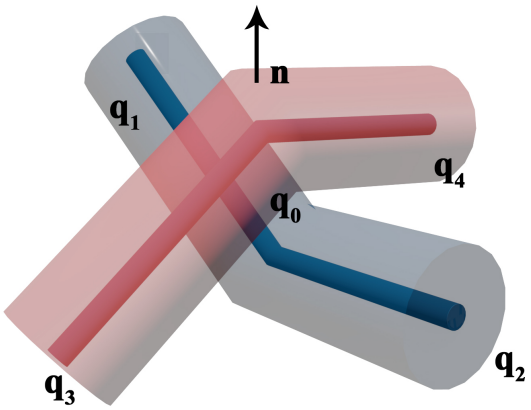


Figure 3.3: Compression force on $\mathbf{q}_0$ along normal $\mathbf{n}$ at $\mathbf{q}_0$.

$$\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_1} + \frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_2}\right) \tag{3.38}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(bend)}}{\partial \mathbf{x}_1} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_1} + \frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_1}\right) \tag{3.39}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(bend)}}{\partial \mathbf{x}_2} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_2} + \frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_2}\right) \tag{3.40}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(bend)}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial \mathbf{x}_0} + \frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial \mathbf{x}_0}\right) \tag{3.41}$$

$$\frac{\partial F_{u_1}^{(bend)}}{\partial u_1} = \frac{\partial F_{u_2}^{(bend)}}{\partial u_2} = \frac{\partial F_{u_1}^{(bend)}}{\partial u_2} = \frac{\partial F_{u_2}^{(bend)}}{\partial u_1} = -\frac{2B\pi R^2\theta^2}{(u_1-u_2)^2} \tag{3.42}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial u_1} = -\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial u_2} = \frac{2B\pi R^2\theta}{l_1(u_1-u_2)^2\sin\theta}\mathbf{P}_{0,1}\mathbf{d}_{0,2} \tag{3.43}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial u_1} = -\frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial u_2} = \frac{2B\pi R^2\theta}{l_2(u_1-u_2)^2\sin\theta}\mathbf{P}_{0,2}\mathbf{d}_{0,1} \tag{3.44}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(bend)}}{\partial u_1} = -\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(bend)}}{\partial u_2} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(bend)}}{\partial u_1} + \frac{\partial \mathbf{F}_{\mathbf{x}_2}^{(bend)}}{\partial u_1}\right) \tag{3.45}$$

$$\frac{\partial F_{u_1}^{(bend)}}{\partial \mathbf{x}_1} = -\frac{\partial F_{u_2}^{(bend)}}{\partial \mathbf{x}_1} = \frac{2B\pi R^2\theta}{l_1(u_1-u_2)^2\sin\theta}\mathbf{d}_{0,2}^{\top}\mathbf{P}_{0,1} \tag{3.46}$$

$$\frac{\partial F_{u_1}^{(bend)}}{\partial \mathbf{x}_2} = -\frac{\partial F_{u_2}^{(bend)}}{\partial \mathbf{x}_2} = \frac{2B\pi R^2\theta}{l_2(u_1-u_2)^2\sin\theta}\mathbf{d}_{0,1}^{\top}\mathbf{P}_{0,2} \tag{3.47}$$

$$\frac{\partial F_{u_1}^{(bend)}}{\partial \mathbf{x}_0} = -\frac{\partial F_{u_2}^{(bend)}}{\partial \mathbf{x}_0} = -\left(\frac{\partial F_{u_1}^{(bend)}}{\partial \mathbf{x}_1} + \frac{\partial F_{u_1}^{(bend)}}{\partial \mathbf{x}_2}\right) \tag{3.48}$$

The yarn-to-yarn interaction forces include friction, shear, and parallel yarn collisions.

Yarn-to-yarn contact. While the aforementioned forces are differentiable, the yarn-to-yarn

forces are not. Existing differentiable contact models mainly correct after-contact positions and velocities (Belbute-Peres et al. 2018; Liang et al. 2019; Zhong et al. 2021) via (multiple) optimization solves, which is too simplistic for fabrics. Yarn-to-yarn contact has its unique features. It is relatively sticky and often has small relative velocities. We need a contact model that reflects this and leads to a differentiable contact force which affects the friction/shear. The contact force is a combination of the stretching $\mathbf{F}^{(stretch)}$ and bending forces $\mathbf{F}^{(bend)}$ at every crossing node along the contact normal $\mathbf{n}$. We assume no-slip contact and compute the contact force by:

$$F_n = \mathrm{ReLU}(\frac{1}{2}\mathbf{n}^\top(\mathbf{F}_u^{(stretch)} + \mathbf{F}_u^{(bend)} - \mathbf{F}_v^{(stretch)} - \mathbf{F}_v^{(bend)})) \tag{3.49}$$

where u and v represent the forces from warp and weft segments. The rectified linear unit (ReLU) ensures the non-negativity of the contact force. The normal $\mathbf{n}$, from warp to weft yarn (Fig. 3.3), is approximated by the normal of the best-fit plane of $\mathbf{q_0}$-$\mathbf{q_4}$.

Friction. The friction between warps and wefts prohibits relative movements, which is crucial to the overall dynamics of the fabric. In differentiable physics, contacts under simple settings have been modeled, such as the standard Coulomb model for kinetic friction (Zhong et al. 2021). But this is insufficient for our purpose for two reasons. First, the static friction plays a key role in stick-slip behaviours of yarns (Zhou et al. 2019) and needs to be modeled. Second, the standard Coulomb friction model is a piece-wise function, which is intrinsically indifferentiable at the static-to-kinetic transition point. Therefore, we need a new differentiable friction model.

The low relative speed between yarns is a special situation where the static-to-kinetic transition could actually be continuous (as opposed to the Coulomb model), experimentally shown by Stribeck (Stribeck 1902). This indicates that a continuous and differentiable model has the potential to be more accurate for yarns than the widely used Coulomb model. Further, the breakaway force causing the static-to-kinetic transition depends on the rate of the external force (Johannes et al. 1973), and the nonlinear stick-slip behavior is related to self-excited vibrations before transition (Awrejcewicz 1988). Inspired by the above research, we propose a new differentiable yarn-to-yarn friction model (Fig. 3.4):

$$F_{Slide} = -\Big(\frac{k_f\delta u - K(\delta u)\mu F_n}{2}K(\mu F_n - F_u) + \frac{k_f\delta u + K(\delta u)\mu F_n}{2}\Big) - d_f\dot{u}_0 \tag{3.50}$$

where $\delta u = u_0 - \bar{u}_0$ and $K(x) = \tanh(px)$. $\bar{u}_0$ is the anchor position when there is no relative

movement between the warp and the weft segment, μ is the friction coefficient, and F_u is the external force. We introduce a hyperparameter p to control the conversion speed between static and kinetic friction. To understand Eq. (3.50), there are three situations: $F_u = 0$ (no external force), $0 \leq F_u \leq \mu F_n$ (static friction), and $F_u > \mu F_n$ (kinetic friction). When $F_u = 0$, $\delta u = 0$, $K(\mu F_n - F_u) = 1$ and the speed $\dot{u}_0 = 0$, so $F_{Slide} = 0$; when $0 \leq F_u \leq \mu F_n$, we allow a small displacement δu to mimic the self-excited vibration in static friction, governed by a Hooke's spring $k_f \delta u$ with stiffness k_f. When F_u is small, i.e., $K(\mu F_n - F_u)$ is close to 1, $F_{Slide} \approx -k_f \delta u - d_f \dot{u}_0$ where d_f is a damping coefficient. F_{Slide} is mainly the static friction minus a small damping term (as $\dot{u}_0$ is small). Due to the time discretization in simulation, the spring force has a delayed response which causes small-range vibrations. When $K(\mu F_n - F_u)$ starts to decrease to 0 and the breakaway force is achieved $F_u = \mu F_n$, $F_{Slide} = -\frac{k_f \delta u + K(\delta u)\mu F_n}{2} - d_f \dot{u}_0$ and $\frac{k_f \delta u + K(\delta u)\mu F_n}{2}$ is the average of the spring force and the maximum static friction. Finally when $F_u > \mu F_n$, $K(\mu F_n - F_u)$ quickly becomes -1 and $K(\delta u)$ becomes 1 as δu increases. Then $F_{Slide} = -\mu F_n - d_f \dot{u}_0$, which is the kinetic friction minus damping. Fig. 3.4 shows our friction can closely approximate the Stribeck effect while maintaining differentiability, as opposed to the indifferentiable Coulomb model. Also, it incorporates self-excited vibrations within $F_{Slide} \in [-F_k, F_k]$ which is when $0 \leq F_u \leq \mu F_n$. The derivative of friction force with respect to node position in Eulerian coordinate is

$$\begin{aligned}\frac{\partial F_{Slide}}{\partial u_0} =& -\frac{k_f - ((1-\tanh^2 \delta u)\mu F_n + \tanh \delta u \mu \frac{\partial F_n}{\partial u_0})}{2} \tanh(\mu F_n - F_u) \\ & - \frac{k_f \delta u - \tanh \delta u \mu F_n}{2}(1 - \tanh^2(\mu F_n - F_u)) \left(\frac{\partial F_u}{\partial u_0} - \mu \frac{\partial F_n}{\partial u_0}\right) \\ & - \frac{k_f + (1-\tanh^2 \delta u)\mu F_n + \tanh \delta u \mu \frac{\partial F_n}{\partial u_0}}{2}\end{aligned} \tag{3.51}$$

The derivative of friction force with respect to node velocity in Eulerian coordinate is

$$\frac{\partial F_{Slide}}{\partial \dot{u}_0} = \frac{k_f \delta u - \tanh \delta u \mu F_n}{2}(1 - \tanh^2(\mu F_n - F_u)) \frac{\partial F_u}{\partial \dot{u}_0} - d_f \tag{3.52}$$

Shear. A shear force is generated when there is relative rotation between a warp and a weft at a crossing node (Parsons et al. 2010), which increases non-linearly when the shear angle increases (Mohammed et al. 2000; Peng et al. 2004; Cao et al. 2008). Previous differential models do not consider this type of forces. Therefore, we propose a new differentiable shear

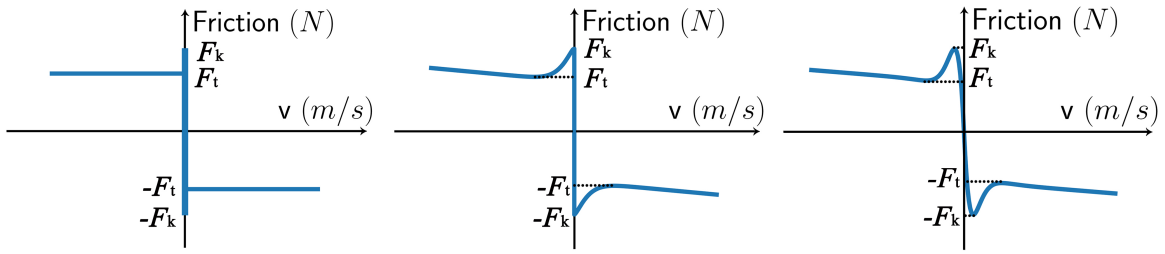


Figure 3.4: F_k and F_t are the static and kinetic friction. Coulomb model (left) is an indifferentiable multi-value function. Stribeck effect (middle) is empirically observed (Stribeck 1902). Our model (right) incorporates the Stribeck effect and also simulates self-excited vibrations around $\mathsf{v} = 0$.

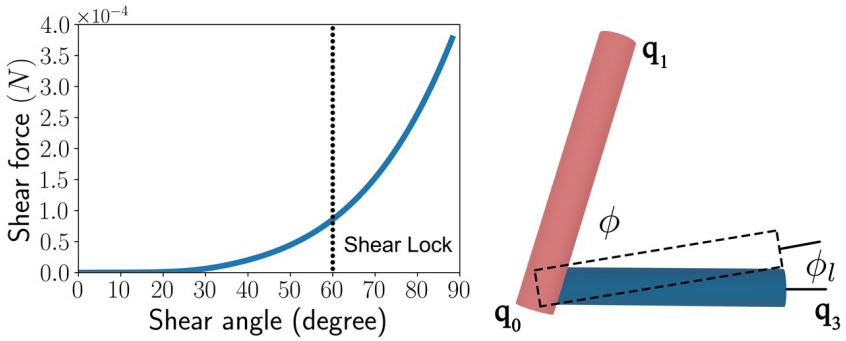


Figure 3.5: Shear force strength vs shear angle $\bar{\phi} - \phi$, (left) and graphical illustration (right).

model. There are different stages when the shear angle increases (King et al. 2005). The shear force first grows almost linearly initially, then 'shear lock' is triggered (Wang et al. 1999) when the angle passes a threshold and the shear force starts to increase exponentially as the angle increases, producing highly non-linear behaviors. We therefore define the shear energy as a function of the shear angle $\bar{\phi} - \phi$ (Fig. 3.5): $\frac{1}{2}k_s L(\phi - \bar{\phi})^2$, where $\bar{\phi} = \frac{\pi}{2}$ is the rest shear angle. $k_s = S\pi R^2(1 + F_n)$ is the shear stiffness and S is the shear modulus. We embed the 'shear lock' by boosting k_s exponentially with $\gamma = (\sqrt{2L^2} - 2\sin\frac{\phi}{2}L)/R$ as long as it stays smaller than the lock threshold $\phi_l = 2\arcsin\frac{R}{L}$: k_s equals to $S\pi R^2(1 + F_n)$ if $\phi > \phi_l$; and $S\pi R^2(1 + F_n)\gamma^c$ otherwise, where c controls the increase rate of k_s with respect to ϕ when 'shear lock' occurs. Although k_s has discontinuities within $[0, \frac{\phi}{2}]$, our new shear stiffness can be defined as:

$$k_s = \frac{1}{2}(F_n + 1)S\pi R^2\left((1 + \gamma^c) + (1 - \gamma^c)\tanh\left(\frac{\bar{\phi}^5(\phi - \phi_l)}{(\phi(\phi - \phi_l)(\phi - \bar{\phi}))^2 + \bar{\phi}^4\sigma^2}\right)\right) \tag{3.53}$$

where σ governs the transition smoothness between lock and no-lock. The smaller the σ is, the

smoother the transition is. The shear forces at those crossing nodes are

$$\mathbf{F}_{\mathbf{x}_1}^{(shear)} = -\frac{\partial V_{1,0,3}^{(shear)}}{\partial \mathbf{x}_1} = -\frac{1}{2}\frac{\partial k_s}{\partial \mathbf{x}_1} L(\phi - \bar{\phi})^2 + \frac{k_s L(\phi - \bar{\phi})}{l_1 \sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3} \tag{3.54}$$

$$\mathbf{F}_{\mathbf{x}_3}^{(shear)} = -\frac{\partial V_{1,0,3}^{(shear)}}{\partial \mathbf{x}_3} = -\frac{1}{2}\frac{\partial k_s}{\partial \mathbf{x}_3} L(\phi - \bar{\phi})^2 + \frac{k_s L(\phi - \bar{\phi})}{l_3 \sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1} \tag{3.55}$$

$$\mathbf{F}_{\mathbf{x}_0}^{(shear)} = -(\mathbf{F}_{\mathbf{x}_1} + \mathbf{F}_{\mathbf{x}_3}) \tag{3.56}$$

For the sake of simplicity, we define:

$$g(\phi) = \frac{\bar{\phi}^5(\phi - \phi_l)}{(\phi(\phi - \phi_l)(\phi - \bar{\phi}))^2 + \bar{\phi}^4\sigma^2}$$

$$f(\phi) = \tanh g(\phi)$$

The numerator and denominator of $g(\phi)$ are

$$g_{num}(\phi) = \bar{\phi}^5(\phi - \phi_l)$$

and

$$g_{den}(\phi) = (\phi(\phi - \phi_l)(\phi - \bar{\phi}))^2 + \bar{\phi}^4\sigma^2$$

Then, we have:

$$\frac{\partial k_s}{\partial \mathbf{x}_3} = \frac{1}{2}(F_n + 1)SR^2\left(c\gamma^{c-1}\frac{\partial \gamma}{\partial \mathbf{x}_3} - c\gamma^{c-1}\frac{\partial \gamma}{\partial \mathbf{x}_3}f(\phi) + (1 - \gamma^c)(1 - f(\phi)^2)\frac{\partial g(\phi)}{\partial \mathbf{x}_3}\right)$$

$$\frac{\partial k_s}{\partial \mathbf{x}_1} = \frac{1}{2}(F_n + 1)SR^2\left(c\gamma^{c-1}\frac{\partial \gamma}{\partial \mathbf{x}_1} - c\gamma^{c-1}\frac{\partial \gamma}{\partial \mathbf{x}_1}f(\phi) + (1 - \gamma^c)(1 - f(\phi)^2)\frac{\partial g(\phi)}{\partial \mathbf{x}_1}\right)$$

where

$$\frac{\partial \gamma}{\partial \mathbf{x}_1} = -\frac{L}{R}\cos\frac{\phi}{2}\frac{\partial \phi}{\partial \mathbf{x}_1},\ \frac{\partial \gamma}{\partial \mathbf{x}_3} = -\frac{L}{R}\cos\frac{\phi}{2}\frac{\partial \phi}{\partial \mathbf{x}_3},$$

$$\frac{\partial g(\phi)}{\partial \mathbf{x}_1} = \frac{\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_1}g_{den}(\phi) - g_{num}(\phi)\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_1}}{g_{den}^2(\phi)},$$

$$\frac{\partial g(\phi)}{\partial \mathbf{x}_3} = \frac{\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_3}g_{den}(\phi) - g_{num}(\phi)\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_3}}{g_{den}^2(\phi)}.$$

The terms $\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_1}$, $\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_1}$, $\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_3}$, and $\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_3}$ are:

$$\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_1} = \bar{\phi}^5 \frac{\partial \phi}{\partial \mathbf{x}_1} = -\bar{\phi}^5 \frac{\mathbf{P}_{0,1}\mathbf{d}_{0,3}}{l_1 \sin\phi}$$

$$\frac{\partial g_{num}(\phi)}{\partial \mathbf{x}_3} = \bar{\phi}^5 \frac{\partial \phi}{\partial \mathbf{x}_3} = -\bar{\phi}^5 \frac{\mathbf{P}_{0,3}\mathbf{d}_{0,1}}{l_3 \sin\phi}$$

$$\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_1} = 2(\phi(\phi-\phi_l)(\phi-\bar{\phi}))\left(\frac{\partial \phi}{\partial \mathbf{x}_1}(\phi-\phi_l)(\phi-\bar{\phi}) + \phi\frac{\partial \phi}{\partial \mathbf{x}_1}(\phi-\bar{\phi}) + \phi(\phi-\phi_l)\frac{\partial \phi}{\partial \mathbf{x}_1}\right)$$

$$\frac{\partial g_{den}(\phi)}{\partial \mathbf{x}_3} = 2(\phi(\phi-\phi_l)(\phi-\bar{\phi}))\left(\frac{\partial (\phi)}{\partial \mathbf{x}_3}(\phi-\phi_l)(\phi-\bar{\phi}) + \phi\frac{\partial \phi}{\partial \mathbf{x}_3}(\phi-\bar{\phi}) + \phi(\phi-\phi_l)\frac{\partial \phi}{\partial \mathbf{x}_3}\right)$$

The derivatives of the shear forces with respect to the nodes' positions in Lagrangian coordinate are:

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_1} = -\frac{1}{2}\frac{\partial^2 k_s}{\partial \mathbf{x}_1 \mathbf{x}_1}L(\phi-\bar{\phi})^2 - L(\phi-\bar{\phi})\frac{\partial k_s}{\partial \mathbf{x}_1}\frac{\partial \phi}{\partial \mathbf{x}_1} + \frac{\partial}{\partial \mathbf{x}_1}\frac{k_s L(\phi-\bar{\phi})}{l_1 \sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_3} = -\frac{1}{2}\frac{\partial^2 k_s}{\partial \mathbf{x}_3 \mathbf{x}_3}L(\phi-\bar{\phi})^2 - L(\phi-\bar{\phi})\frac{\partial k_s}{\partial \mathbf{x}_3}\frac{\partial \phi}{\partial \mathbf{x}_3} + \frac{\partial}{\partial \mathbf{x}_3}\frac{k_s L(\phi-\bar{\phi})}{l_3 \sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_3} = -\frac{1}{2}\frac{\partial^2 k_s}{\partial \mathbf{x}_1 \mathbf{x}_3}L(\phi-\bar{\phi})^2 - L(\phi-\bar{\phi})\frac{\partial k_s}{\partial \mathbf{x}_1}\frac{\partial \phi}{\partial \mathbf{x}_3} - L(\phi-\bar{\phi})\frac{\partial \phi}{\partial \mathbf{x}_1}\frac{\partial k_s}{\partial \mathbf{x}_3} + \frac{\partial}{\partial \mathbf{x}_3}\frac{k_s L(\phi-\bar{\phi})}{l_1 \sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_1} = -\frac{1}{2}\frac{\partial^2 k_s}{\partial \mathbf{x}_3 \mathbf{x}_1}L(\phi-\bar{\phi})^2 - L(\phi-\bar{\phi})\frac{\partial k_s}{\partial \mathbf{x}_3}\frac{\partial \phi}{\partial \mathbf{x}_1} - L(\phi-\bar{\phi})\frac{\partial \phi}{\partial \mathbf{x}_3}\frac{\partial k_s}{\partial \mathbf{x}_1} + \frac{\partial}{\partial \mathbf{x}_1}\frac{k_s L(\phi-\bar{\phi})}{l_3 \sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}$$

where

$$\frac{\partial}{\partial \mathbf{x}_1}\frac{k_s L(\phi-\bar{\phi})}{l_1 \sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3} = \frac{k_s L}{l_1^2 \sin\phi}\Bigg(\Big(\phi-\bar{\phi}\Big)\Big(-\mathbf{P}_{0,1}\mathbf{d}_{0,3}\mathbf{d}_{0,1}^\top + \frac{\cos\phi}{\sin^2\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}\mathbf{d}_{0,3}^\top\mathbf{P}_{0,1} - \cos\phi\mathbf{P}_{0,1} - \mathbf{d}_{0,1}\mathbf{d}_{0,3}^\top\mathbf{P}_{0,1}\Big) - \frac{1}{\sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}\mathbf{d}_{0,3}^\top\mathbf{P}_{0,1}\Bigg)$$

$$\frac{\partial}{\partial \mathbf{x}_3}\frac{k_s L(\phi-\bar{\phi})}{l_3 \sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1} = \frac{k_s L}{l_3^2 \sin\phi}\Bigg(\Big(\phi-\bar{\phi}\Big)\Big(-\mathbf{P}_{0,3}\mathbf{d}_{0,1}\mathbf{d}_{0,3}^\top + \frac{\cos\phi}{\sin^2\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,3} - \cos\phi\mathbf{P}_{0,3} - \mathbf{d}_{0,3}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,3}\Big) - \frac{1}{\sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,3}\Bigg)$$

$$\frac{\partial}{\partial \mathbf{x}_3}\frac{k_s L(\phi-\bar{\phi})}{l_1 \sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3} = \frac{k_s L}{l_3 l_1 \sin\phi}\Bigg(\Big(\phi-\bar{\phi}\Big)\Big(\frac{\cos\phi}{\sin^2\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,3} + \mathbf{P}_{0,1}\mathbf{P}_{0,3}\Big) - \frac{1}{\sin\phi}\mathbf{P}_{0,1}\mathbf{d}_{0,3}\mathbf{d}_{0,1}^\top\mathbf{P}_{0,3}\Bigg)$$

$$\frac{\partial}{\partial \mathbf{x}_1}\frac{k_s L(\phi-\bar{\phi})}{l_3\sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1} = \frac{k_s L}{l_1 l_3\sin\phi}\left(\left(\phi-\bar{\phi}\right)\left(\frac{\cos\phi}{\sin^2\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}\mathbf{d}_{0,3}^{\top}\mathbf{P}_{0,1}+\mathbf{P}_{0,3}\mathbf{P}_{0,1}\right)\right.$$
$$\left.-\frac{1}{\sin\phi}\mathbf{P}_{0,3}\mathbf{d}_{0,1}\mathbf{d}_{0,3}^{\top}\mathbf{P}_{0,1}\right)$$

Moreover, the other terms are

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_1}+\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_3}\right), \frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_1}+\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_3}\right)$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}}{\partial \mathbf{x}_1} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_1}+\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_1}\right), \frac{\partial \mathbf{F}_{\mathbf{x}_0}}{\partial \mathbf{x}_3} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_3}+\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_3}\right)$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}}{\partial \mathbf{x}_0} = -\left(\frac{\partial \mathbf{F}_{\mathbf{x}_1}}{\partial \mathbf{x}_0}+\frac{\partial \mathbf{F}_{\mathbf{x}_3}}{\partial \mathbf{x}_0}\right)$$

Yarn-to-yarn collision. The last internal force is the yarn-to-yarn collisions between parallel yarns. Although it is theoretically possible to use an existing approach (Belbute-Peres et al. 2018; Liang et al. 2019), it would require forming an optimization for all segments and therefore become prohibitively slow. Therefore, we introduce a new penalty energy, defined as a function of the nodes' distance in Eulerian coordinates for a warp segment $[\mathbf{q}_0, \mathbf{q}_1]$ (similar for a weft segment):

$$V_{0,1}^{(collision)} = \frac{1}{2}k_c L(\mathrm{ReLU}(d-\Delta u))^2 \tag{3.57}$$

where $d = 4R$ or $2R$ and k_c defines collision stiffness. The yarn-to-yarn collision forces are:

$$F_{u_0} = -\frac{\partial V_{0,1}}{\partial u_0} = k_c L(\Delta u - d) \tag{3.58}$$

$$F_{u_1} = -\frac{\partial V_{0,1}}{\partial u_1} = -k_c L(\Delta u - d) \tag{3.59}$$

The derivatives of the forces with respect to the nodes' position in Eulerian coordiates:

$$\frac{\partial F_{u_0}}{\partial u_0} = \frac{\partial F_{u_1}}{\partial u_1} = -\frac{\partial F_{u_0}}{\partial u_1} = -\frac{\partial F_{u_1}}{\partial u_0} = -k_c L \tag{3.60}$$

External forces and collisions. Without loss of generality, we consider two external forces: gravity and wind force. Their impacts can be modeled by defining proper potential energies. We define a gravitational energy which is computed segment-wise. To a warp segment $[\mathbf{q}_0, \mathbf{q}_1]$,

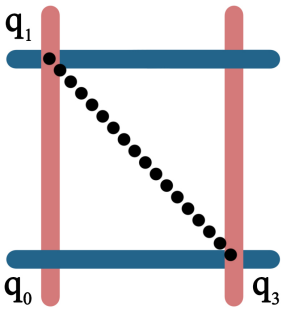


Figure 3.6: Treat a square hold in 4 segments as two triangles.

it gravitational energy is defined as

$$V_{0,1}^{(gravity)} = \rho \Delta u \mathbf{g}^\top \frac{\mathbf{x}_0 + \mathbf{x}_1}{2} \tag{3.61}$$

where $\mathbf{g} \in \mathbb{R}_3$ is the gravity of earth which is approximately set to $(0, 0, 9.8) m/s^2$. The gravity at the nodes are

$$\mathbf{F}_{\mathbf{x}_0}^{(gravity)} = -\frac{\partial V_{0,1}^{(gravity)}}{\partial \mathbf{x}_0} = -\frac{1}{2}\rho \mathbf{g} \Delta u \tag{3.62}$$

$$\mathbf{F}_{\mathbf{x}_1}^{(gravity)} = -\frac{\partial V_{0,1}^{(gravity)}}{\partial \mathbf{x}_1} = -\frac{1}{2}\rho \mathbf{g} \Delta u \tag{3.63}$$

$$F_{u_0}^{(gravity)} = -\frac{\partial V_{0,1}^{(gravity)}}{\partial u_0} = \frac{1}{2}\rho \mathbf{g}^\top (\mathbf{x}_1 + \mathbf{x}_0) \tag{3.64}$$

$$F_{u_1}^{(gravity)} = -\frac{\partial V_{0,1}^{(gravity)}}{\partial u_1} = -\frac{1}{2}\rho \mathbf{g}^\top (\mathbf{x}_1 + \mathbf{x}_0) \tag{3.65}$$

The derivative of the force with respect to the nodes' position are:

$$\frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(gravity)}}{\partial u_0} = \frac{1}{2}\rho \mathbf{g} \quad \frac{\partial \mathbf{F}_{\mathbf{x}_0}^{(gravity)}}{\partial u_1} = -\frac{1}{2}\rho \mathbf{g} \tag{3.66}$$

$$\frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(gravity)}}{\partial u_0} = \frac{1}{2}\rho \mathbf{g} \quad \frac{\partial \mathbf{F}_{\mathbf{x}_1}^{(gravity)}}{\partial u_1} = -\frac{1}{2}\rho \mathbf{g} \tag{3.67}$$

$$\frac{\partial F_{u_0}^{(gravity)}}{\partial \mathbf{x}_1} = \frac{1}{2}\rho \mathbf{g}^\top \quad \frac{\partial F_{u_0}^{(gravity)}}{\partial \mathbf{x}_0} = \frac{1}{2}\rho \mathbf{g}^\top \tag{3.68}$$

$$\frac{\partial F_{u_1}^{(gravity)}}{\partial \mathbf{x}_1} = -\frac{1}{2}\rho \mathbf{g}^\top \quad \frac{\partial F_{u_1}^{(gravity)}}{\partial \mathbf{x}_0} = -\frac{1}{2}\rho \mathbf{g}^\top \tag{3.69}$$

To apply wind force to the surface of the cloth, we need to compute an area-based force. Every square composed of four segments can be split into two triangles when computing wind force

(shown in Fig. 3.6). The wind force has three properties affecting its influence on the cloth: wind velocity $\mathbf{v}_w$, density ρ_w, and drag d_w. In this work, we set $\mathbf{v}_w = (0, 5, 0)$, $\rho_w = 2$, and $d_w = 0.5$. The wind force imposed on a triangle face $[\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_3]$ is defined as:

$$\mathbf{F}^{(wind)} = \rho_w a |v_n| v_n \mathbf{n}_f + d_w \mathbf{v}_t \tag{3.70}$$

where a is the face area, $\mathbf{n}_f$ is the face normal, and wind velocity along the face's normal direction and tangential direction respectively are

$$v_n = \mathbf{n}_f \left(\mathbf{v}_w - \frac{\dot{\mathbf{x}}_0 + \dot{\mathbf{x}}_1 + \dot{\mathbf{x}}_3}{3} \right),$$

$$\mathbf{v}_t = \frac{\dot{\mathbf{x}}_0 + \dot{\mathbf{x}}_1 + \dot{\mathbf{x}}_3}{3} - v_n \mathbf{n}_f.$$

The force is evenly imposed on the three nodes:

$$\mathbf{F}_{\mathbf{x}_0}^{(wind)} = \mathbf{F}_{\mathbf{x}_1}^{(wind)} = \mathbf{F}_{\mathbf{x}_3}^{(wind)} = \frac{1}{3} \mathbf{F}^{(wind)}. \tag{3.71}$$

Finally, after calculating all forces, the resultant force at every crossing node is the combined force of all segments that connect to that node. The cloth is simulated by solving Eq. (3.8). We adopt a collision handling method originally designed for triangular meshes stored in bounding volume hierarchy (Tang et al. 2010) where continuous collision detection (CCD) can detect edge-edge and vertex-face collision. The detected vertices, edges, and faces are grouped into non-rigid impact zones (Harmon et al. 2008) for computing collision response. We treat collision response as a constrained optimization problem to prevent penetrations (Liang et al. 2019):

$$\begin{aligned} \underset{(\mathbf{x}_{colli} - \mathbf{x})}{\text{minimize}} \quad & \frac{1}{2} (\mathbf{x}_{colli} - \mathbf{x})^\top \mathbf{W} (\mathbf{x}_{colli} - \mathbf{x}) \\ \text{subject to} \quad & \mathbf{G}\mathbf{x}_{colli} + \mathbf{h} \leq \mathbf{0} \end{aligned}$$

where $\mathbf{W}$ is a weight matrix, $\mathbf{x}$ is the Lagrangian part of $\mathbf{q}$, $\mathbf{x}_{colli}$ is the updated $\mathbf{x}$ where no collision can be detected. $\mathbf{G}$ and $\mathbf{h}$ are constraint parameters. We assume neither self-collision nor cloth-object collision can generate considerable yarn-sliding motions, so we exclude the Eulerian terms.

3.3.4 Derivatives of the simulator

Now we have a fully differentiable simulator with parameters $\boldsymbol{\omega}$. The $\boldsymbol{\omega}$ is the cloth physical parameters (stretching, bending, shearing, etc) when solving inverse problems, or the to be learned external forces in control experiments. Given a loss function $\mathcal{L}$, its gradient with respect to the parameters $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}}$ can help learn the right physics parameters via back-propagation. For simplicity, we use $\mathbf{A}\dot{\mathbf{q}} = \mathbf{b}$ to represent Eq. (3.8). The differential of $\mathbf{A}\dot{\mathbf{q}} = \mathbf{b}$ is (Magnus and Neudecker 2019):

$$\mathbf{A}\mathsf{d}\dot{\mathbf{q}} = \mathsf{d}\mathbf{b} - \mathsf{d}\mathbf{A}\dot{\mathbf{q}} \tag{3.72}$$

We can form the Jacobians of $\dot{\mathbf{q}}$ with respect to $\mathbf{A}$ or $\mathbf{b}$ with Eq. (3.72). For example, to compute the $\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{A}}$, we need to set $\mathsf{d}\mathbf{A} = \mathbf{I}$ and $\mathsf{d}\mathbf{b} = \mathbf{0}$, then solve the equation and the result is $\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{A}}$. As pointed out by (Amos and Kolter 2017), it is unnecessary to explicitly compute these Jacobians in back-propagation. We want to compute the product of the vector passed from back-propagation, $\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}$, and the Jacobians of $\dot{\mathbf{q}}$, i.e., $\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{A}}$ and $\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{b}}$. Assume $\mathbf{A} \in \mathbb{R}^{3\times 3}$, $\dot{\mathbf{q}} \in \mathbb{R}^3$, and $\mathbf{b} \in \mathbb{R}^3$, then

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}} = \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{b}} = \left(\begin{pmatrix} \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_1} & \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_2} & \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_3} \end{pmatrix} \begin{pmatrix} \frac{\partial \dot{\mathbf{q}}_1}{\partial \mathbf{b}_1} & \frac{\partial \dot{\mathbf{q}}_1}{\partial \mathbf{b}_2} & \frac{\partial \dot{\mathbf{q}}_1}{\partial \mathbf{b}_3} \\ \frac{\partial \dot{\mathbf{q}}_2}{\partial \mathbf{b}_1} & \frac{\partial \dot{\mathbf{q}}_2}{\partial \mathbf{b}_2} & \frac{\partial \dot{\mathbf{q}}_2}{\partial \mathbf{b}_3} \\ \frac{\partial \dot{\mathbf{q}}_3}{\partial \mathbf{b}_1} & \frac{\partial \dot{\mathbf{q}}_3}{\partial \mathbf{b}_2} & \frac{\partial \dot{\mathbf{q}}_3}{\partial \mathbf{b}_3} \end{pmatrix} \right)^\top \tag{3.73}$$

As

$$\frac{\partial \dot{\mathbf{q}}_1}{\partial \mathbf{b}_1} = \frac{\partial \left((\mathbf{A}^{-1})_{1,1}\mathbf{b}_1 + (\mathbf{A}^{-1})_{1,1}\mathbf{b}_2 + (\mathbf{A}^{-1})_{1,1}\mathbf{b}_3 \right)}{\partial \mathbf{b}_1} = \mathbf{A}^{-1}_{1,1}$$

and similarly for $\frac{\partial \dot{\mathbf{q}}_i}{\partial \mathbf{b}_j}$, Eq. (3.73) can be represented as:

$$\left(\begin{pmatrix} \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_1} & \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_2} & \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_3} \end{pmatrix} \begin{pmatrix} (\mathbf{A}^{-1})_{1,1} & (\mathbf{A}^{-1})_{1,2} & (\mathbf{A}^{-1})_{1,3} \\ (\mathbf{A}^{-1})_{2,1} & (\mathbf{A}^{-1})_{2,2} & (\mathbf{A}^{-1})_{2,3} \\ (\mathbf{A}^{-1})_{3,1} & (\mathbf{A}^{-1})_{3,2} & (\mathbf{A}^{-1})_{3,3} \end{pmatrix} \right)^\top = (\mathbf{A}^{-1})^\top \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \tag{3.74}$$

After computing $\frac{\partial \mathcal{L}}{\partial \mathbf{b}}$, we need to compute $\frac{\partial \mathcal{L}}{\partial \mathbf{A}}$. The $\mathbf{b}$ in Eq. (3.72) can be set to 0 because it is irrelevant when computing $\frac{\partial \mathcal{L}}{\partial \mathbf{A}}$. Then we have

$$\mathbf{A}\mathsf{d}\dot{\mathbf{q}} = -\mathsf{d}\mathbf{A}\dot{\mathbf{q}} \tag{3.75}$$

The derivative of $\dot{\mathbf{q}}$ with respect to $\mathbf{A}_{i,j}$, the entry in the ith row and jth column of the matrix $\mathbf{A}$, is

$$\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{A}_{i,j}} = \mathbf{A}^{-1} \begin{pmatrix} \mathbf{0} \\ -\dot{\mathbf{q}}_j \\ \mathbf{0} \end{pmatrix} \tag{3.76}$$

According to chain rule,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i,j}} = \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{A}_{i,j}} = \frac{\partial \mathcal{L}}{\partial \mathbf{b}}^{\top} \mathbf{A}\mathbf{A}^{-1} \begin{pmatrix} \mathbf{0} \\ -\dot{\mathbf{q}}_j \\ \mathbf{0} \end{pmatrix} = -\left(\frac{\partial \mathcal{L}}{\partial \mathbf{b}}\right)_i \dot{\mathbf{q}}_j \tag{3.77}$$

The more general form is

$$\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = -\frac{\partial \mathcal{L}}{\partial \mathbf{b}} \dot{\mathbf{q}}^{\top} \tag{3.78}$$

By using chain rule, we have $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}} = \frac{\partial \mathcal{L}}{\partial \mathbf{A}} \frac{\partial \mathbf{A}}{\partial \boldsymbol{\omega}}$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}} = \frac{\partial \mathcal{L}}{\partial \mathbf{b}} \frac{\partial \mathbf{b}}{\partial \boldsymbol{\omega}}$. Finally, we define the loss function: $\mathcal{L}(\mathbf{q}, \hat{\mathbf{q}}) = \frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} \|\mathbf{q}_{n,t} - \hat{\mathbf{q}}_{n,t}\|_2^2$, where $\mathbf{q}_{n,t}$ and $\hat{\mathbf{q}}_{n,t}$ are the ground-truth and predicted general position. N and T are the total number of nodes and simulation steps respectively. We use Stochastic Gradient Descent and run 70 epochs for training.

Underconstrainedness Mitigation. Learning physical parameters via solving an inverse problem is intrinsically under-constrained, leading to multiple solutions or implausible parameter values when fitting data, e.g., unconstrained learning leads to negative density. We mitigate this issue by incorporating prior knowledge. Instead of directly learning parameters $\boldsymbol{\omega}$, we set $\boldsymbol{\omega} = a \times \text{sigmoid}(y) + b$ where a and b are tunable scalars, and we learn y instead. a and b essentially limit the range of $\boldsymbol{\omega}$. We can induce prior knowledge of parameter ranges such as yarn density ranges, because although the exact value is to be learned and not known *a priori*, their ranges are available in practice. Our experiments demonstrate that this strategy effectively mitigates the multi-solution issue.

3.4 Experiments

We employ a traditional indifferentiable yarn-level simulator (Cirio et al. 2014) to generate the ground-truth data, and build a dataset of fabrics with three types of yarns and three types of woven patterns. The yarns vary in density, elastic modulus, and bending modulus (Table 3.1).

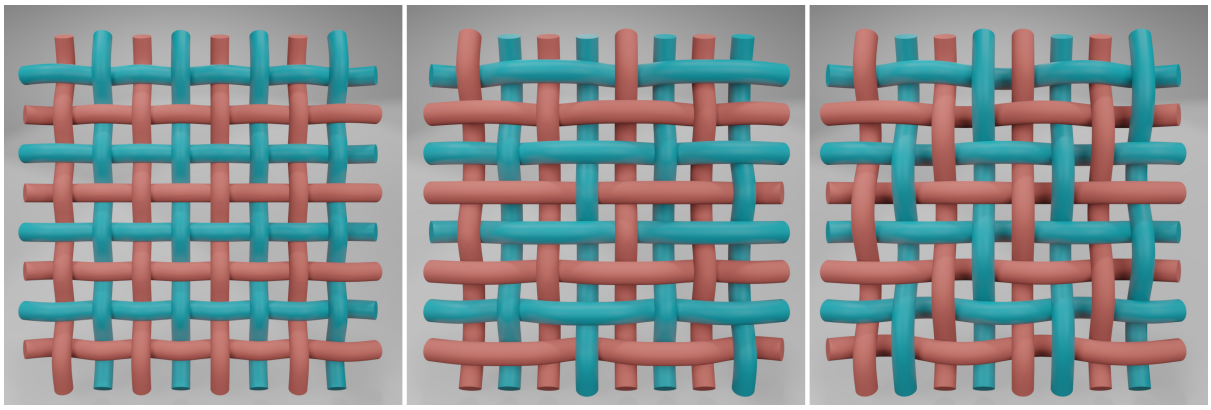


Figure 3.7: Woven patterns. Left-to-right: plain, satin, and twill. Teal and coral indicate different yarns.

Table 3.1: Ground-truth parameters of three yarns.

Parameter/Yarn	Yarn1	Yarn2	Yarn3
Density(kg/m)	0.0020	0.0025	0.0024
Stretch modulus(N/m)	500000	170000	120000
Bending modulus(N/m)	0.00014	0.00011	0.00009

The three woven patterns include plain, twill and satin. We use hybrid fabrics made from two types of yarns, and exhaustively combine three yarns with three woven patterns (Fig. 3.7) to generate 9 types of fabrics. We denote them as XXX-(X, X) where the prefix is the woven pattern and the numbers in the brackets are the yarns, e.g., Plain-(1, 2) means a plain pattern woven with Yarn1 and Yarn2. In our ground-truth data, we use a square piece of cloth hanging at its two corners and blown by wind with a constant magnitude. The simulation is conducted for 500 steps with $h = 0.001s$. Training details are in Appendix A.

3.4.1 Learning physical parameters

We first demonstrate our model's effectiveness in learning meaningful physical parameters, under various model sizes and different amounts of training data.

Learning capacity. We first test whether meaningful physical parameters can be learned from cloths of different sizes. Small-size cloths tend to show low-frequency features, e.g., the general shape, as opposed to high-frequency features, such as wrinkles and buckling. We test our model on simulation data with sizes: 5×5, 10×10, 17×17 and 25×25, trained on the first 25 frames. Table 3.2 shows that our model can effectively estimate yarn parameters with underlying physics models of different sizes. This has several implications. First, although cloth size does affect the overall dynamics of the motion in the ground-truth data (e.g., larger cloths have more wrinkles),

it does not affect our model's learning capability. Second, since our model can reliably learn the yarn parameters on a small fraction of a cloth, it saves the computation of learning from large cloths, which improves the learning scalability. We can learn from small cloths and then scale to simulate large cloths. Further, inter-yarn parameters including shear S and friction coefficient μ are highly correlated so that the model can easily end up learning only plausible parameter values rather than the true values. This is where we expect our model to suffer from the under-constrainedness problem as (Liang et al. 2019). Surprisingly, our model can learn the right parameters under different sizes. By introducing prior knowledge as aforementioned, the learned parameters are restricted within valid ranges.

Table 3.2: Inter/intra parameters learned on Plain-(1, 2), with ground-truth $S = 1000Pa$, $\mu = 0.5$.

Size	Shear S	Friction μ	Yarn	Density	Stretch	Bend
5×5	949	0.437	1	2.028×10^{-3}	479523	1.387×10^{-4}
			2	2.450×10^{-3}	172928	1.112×10^{-4}
10×10	932	0.455	1	1.991×10^{-3}	484719	1.325×10^{-4}
			2	2.448×10^{-3}	173843	1.026×10^{-4}
17×17	947	0.402	1	1.969×10^{-3}	505421	1.323×10^{-4}
			2	2.440×10^{-3}	171304	1.034×10^{-4}
25×25	913	0.380	1	2.069×10^{-3}	510215	1.488×10^{-4}
			2	2.443×10^{-3}	173920	1.201×10^{-4}

Data efficiency. Data efficiency is crucial as obtaining the ground-truth data can be expensive. Precise 3D geometry capture of real cloths is difficult and time-consuming, while simulation of high-res cloths is prohibitively slow. We further investigate the data efficiency by varying the amount of training data. We gradually increase the training data from the first 5 frames to the first 25 frames. Table 3.3 shows that our model has high data efficiency. It can learn reasonably well from as few as the first 5 frames. The benefits are two-fold. First, our model needs just a few frames to train, making it highly applicable. The second benefit is bigger but less obvious. The first 5 frames (from a static pose) normally contains little dynamics as the cloth just starts to move. This indicates that our model only requires a few frames of low-dynamics motions. This eases real-world measurements on cloth because no large motions are needed. This also saves time if simulation data is used, as small time step size is usually demanded in high-dynamic motion simulations.

All simulations can be found in the supplementary video. We also include simulations with collisions and simulations on large cloths using parameters learnt on small cloths. More results

Table 3.3: Plain-(1,2) learnt parameters on different training data. Left: Yarn1, Right:Yarn2.

Frames	Density	Stretch	Bend	Density	Stretch	Bend
5	2.030×10^{-3}	494301	1.357×10^{-4}	2.450×10^{-3}	169597	1.130×10^{-4}
10	2.037×10^{-3}	491717	1.379×10^{-4}	2.443×10^{-3}	169543	1.130×10^{-4}
25	2.038×10^{-3}	491873	1.367×10^{-4}	2.447×10^{-3}	167217	1.096×10^{-4}

Table 3.4: Testing errors ($\times 10^{-6}$) of our model (left) and (Liang et al. 2019) (middle) and BO (right) trained on 5, 10 and 25 frames generated by yarn-level simulator (Cirio et al. 2014).

Fabrics/Frames	5	25	5	25	5	25
Plain-(1,2)	1.152×10^{-4}	3.962×10^{-5}	1.461	0.4124	0.512	0.109
Plain-(1,3)	1.516×10^{-4}	3.555×10^{-5}	1.608	0.4567	1.280	0.738
Plain-(2,3)	5.233×10^{-4}	2.117×10^{-5}	1.952	0.2294	28.19	18.16

and details are in Appendix A.

3.4.2 Comparisons

Prediction & Data Efficiency

To our best knowledge, there is no similar fine-grained DPM in the literature. The closest method is a general sheet model (Liang et al. 2019), so we compare our model with theirs. We employ their settings, and use a 17×17 model and 50 frames simulation data, with 5, 10 and 25 frames for training and the whole 50 for testing. Since the two methods model cloths at different levels of granularity, their physical parameters are not directly comparable. We therefore compare their Mean Squared Error (MSE). We also include a traditional parameter estimation method based on Bayesian Optimization (Snoek et al. 2012) combined with a yarn-level simulator (Cirio et al. 2014) as another baseline. For simplicity, we refer Bayesian Optimization as to BO. In BO, we randomly select 5 initial points and use the expected improvement (Jones et al. 1998) as the acquisition function. As the learning process of differentiable simulation consists of forward simulation and backward simulation, training 70 epochs can be considered as running 140 simulations. Therefore, we run 140 iterations when using BO. Moreover, we impose the same parameter ranges in the BO as we did in our model.

From Table 3.4, our model uses data more efficiently than (Liang et al. 2019) and BO. From training on 5 frames to 25 frames, our model reduces the error by as much as 96% on Plain-(2, 3), while the largest improvements by the sheet-level model and BO optimization are 88% on Plain-(2, 3) and 78% on Plain-(1,2) respectively. Moreover, as shown in Fig. 3.8 left, our

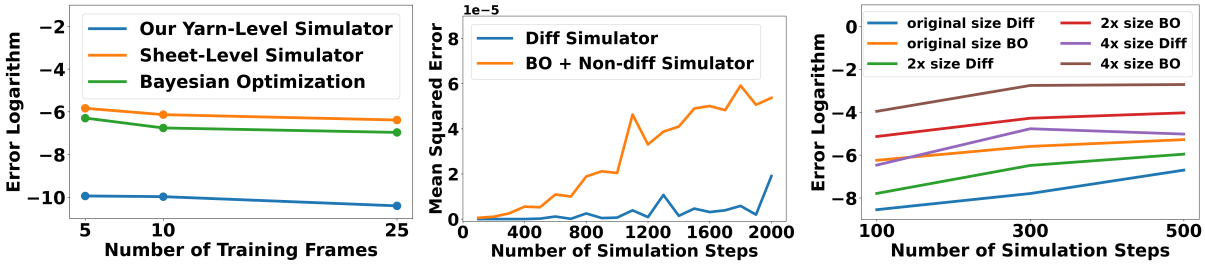


Figure 3.8: Simulation errors: data efficiency (left), long (middle) and big cloths simulation (right).

error on 5 frames is already several magnitudes smaller than the baselines. Further reducing it requires the model to be able to learn subtle dynamics very well. (Liang et al. 2019) essentially treats fabrics as a sheet. Since the simulation is from a yarn-level simulator (Cirio et al. 2014) which contains rich dynamics, the sheet model cannot precisely capture the subtle dynamics caused by individual yarns and their interactions. Further, the model granularity difference has more profound impact than just prediction. Being able to learn yarn parameters has immediate benefits for manufacturing and design, in terms of providing guidance on the choices of yarns and woven patterns. In addition, although BO can sometimes perform slightly better than the sheet model benefiting from a yarn-level simulator, its optimization process is not as efficient as ours. We only show results on Plain here and refer the reader to Appendix A for Satin, Twill, and video comparisons.

Error significance. The MSE errors in Table 3.4 seem to be small, this is because the cloth is small and only simulated for a short period of time. But the results suggest errors in parameter estimation, which are amplified when the cloth is larger and simulated for a longer time. To demonstrate this, first, we run forward simulations for 2000 steps with parameters learned by our model and BO. Second, we show the compound influence using the parameters estimated by our model, (Liang et al. 2019), and BO, and simulate a 17×17 cloth for 500 steps in the original size, 2 times size, and 4 times size. Fig. 3.8 middle-right show both results, which demonstrates the importance of accurate parameter estimation. The errors of BO and (Liang et al. 2019) quickly become several times higher than our model when we scale the size and simulation time. We also show a visual comparison in Fig. 3.9 and refer the reader to Appendix A for more results.

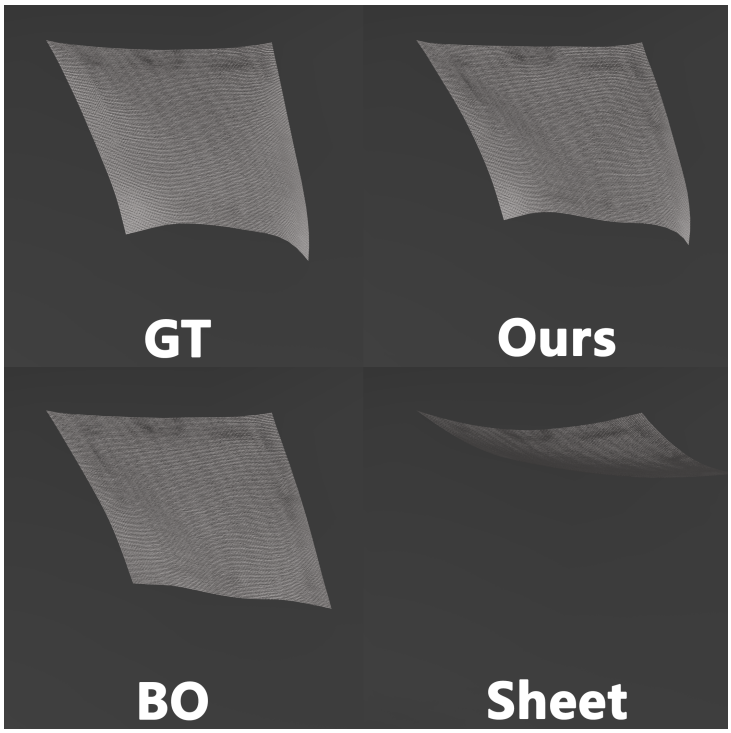


Figure 3.9: Simulation snapshots of the same step. The parameters estimated by our model is visually closest to the ground truth.

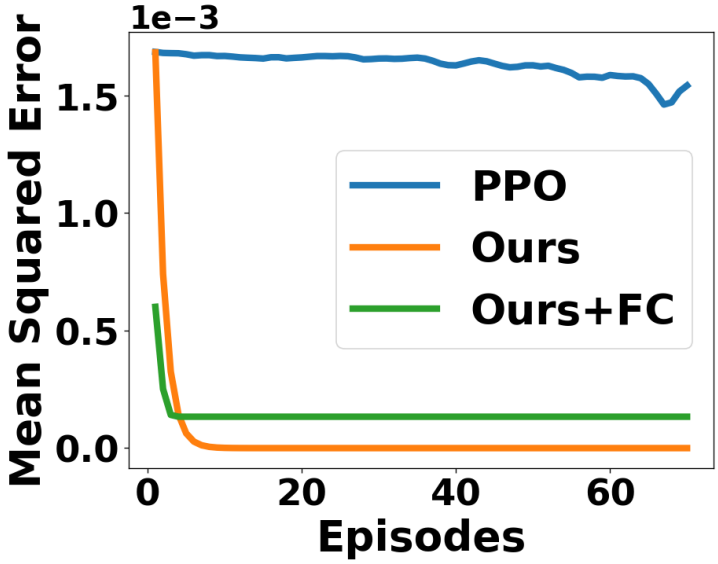


Figure 3.10: The MSE errors against epochs. Our approach learns faster than PPO.

Control Learning

We also show that our model can facilitate control learning. We design a task with a cloth placed on a table and aims to learn forces applied onto the four corners of the cloth to throw it into a box next to the table. The forces are only applied in the first 5 frames. We use our model to learn a sequence of forces which can throw the cloth into the box and compare it with a reinforcement learning baseline model: PPO (Schulman et al. 2017). In addition, we also present a variant of our model by appending two fully-connected layers after our model output (Ours + FC). We use the center of the box's bottom as the target location. When training our model, we use the l_2 distance between the cloth center of mass and the target position as the loss. When training PPO, we use the same l_2 distance for the reward.

The result shows both our model and Ours+FC can quickly learn the forces to throw the cloth into the box. By contrast, PPO is model-free and much slower because it needs to sample in a huge action space. By contrast, the full differentiability of our model enables a quicker search for effective control forces. More details can be found in Appendix A.

In a broader context, there are also model-free methods (Yang et al. 2017; Pfaff et al. 2021) which can also learn physics. The differences between our model and theirs are: 1. model explicability. The parameters that our model learns are interpretable and have physical meanings, so that it can guide manufacturing and design. 2. data efficiency. The data efficiency is much higher in our method. Our model can use as few as 5 frames for learning while model-free methods typically require hundreds to thousands.

3.5 Discussion and Conclusion

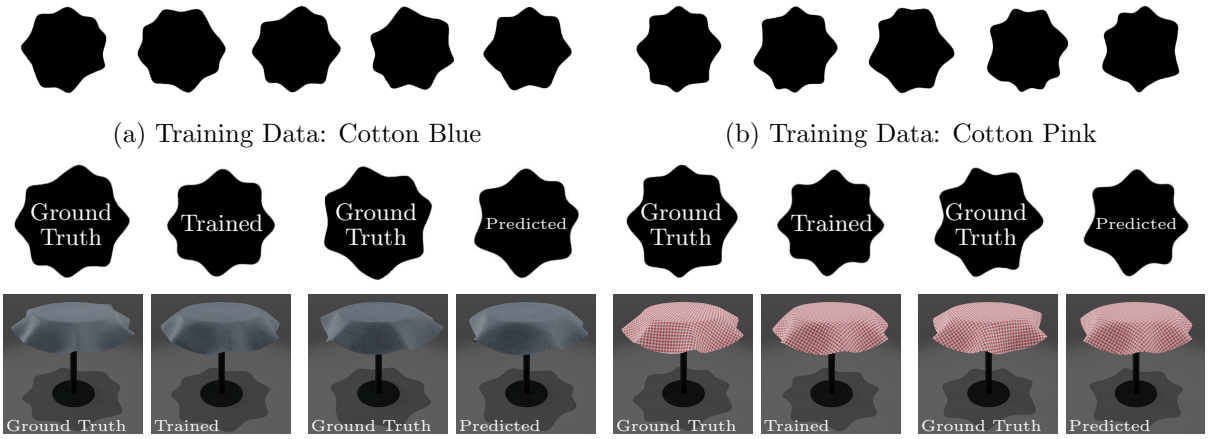
Our method is model-based, which requires domain knowledge and cannot simply 'plug and play' on data as model-free methods (Pfaff et al. 2021). However, strong inductive biases from domain knowledge are necessary for differentiable physics to be applied in applications, because the model behaviour needs to be explainable in such applications, and cannot be merely black-box regression. Representative application domains include fabric manufacture/design and computer graphics, where both simulation and inverse problems need to be solved. Albeit focused on cloth, our model can be readily extended to general composite materials with mesh structures, e.g., from metal/plastic nets to buildings. In addition, our model can be embedded

as a layer into a neural network, which helps learning control policies for cloth manipulation. Further, our model potentially enables a synergy between empirical physics modeling and deep learning, where our model can serve as a deterministic physics layer and other layers can incorporate non-linearity such as high-frequency dynamics in the system (Shen et al. 2021). Finally, our modeling of general forces such as friction and shear contributes to differentiable physical modeling in a wider range, given the universal presence of such forces in the real world.

To our best knowledge, we proposed the first yarn-level differentiable fabric simulator, in the pursuit of fine-grained DPMs capable of incorporating domain knowledge. Through comprehensive evaluation, our model can effectively solve inverse problems, provide high data efficiency and facilitate control. We investigated differentiable modeling of common forces such as friction and shear, which provides a foundation for future attempts on fine-grained differentiable physics modeling. Running efficiency and memory optimization are not included in the main objectives of this work. In future, we will leverage GPU parallel and optimize memory consumption to enable differentiable cloth simulator to handle cloths with higher DoFs.

Chapter 4

Bayesian Differentiable Physics for Fabric Parameter Estimation



(a) Training Data: Cotton Blue (b) Training Data: Cotton Pink

(c) Training Cotton Blue (d) Testing Cotton Blue (e) Training Cotton Pink (f) Testing Cotton Pink

Figure 4.1: Our Bayesian Differentiable Physical (BDP) model can effectively learn the probabilistic distribution of fabric physical parameters from limited Cusick drape data. By learning from Cusick drape silhouettes (a and b), our model shows a strong data-fitting capacity, in reproducing training data (c and e), and also a superior prediction capability in predicting testing fabric drapes (d and f).

We propose a new method to estimate physical properties of fabrics through images. Unlike the previous methods which learn through meshes/videos/images captured under relatively casual settings/simulations, we propose to learn from data collected in strictly tested measuring protocols, to enable more accurate capture of fabric physical properties and potentially tighter integration with application domains such as textile and fabric design. To this end, we create a new Cusick drape dataset. Coupled with the new data, we also propose a new Bayesian differentiable fabrics model for physical property estimation. It can provide highly plausible

estimation on the physical properties of fabrics from very limited data samples. Through exhaustive evaluations and comparisons, we show our method is *accurate* in physical property estimation, *efficient* in learning from limited data samples, and *general* in capturing physical property variations both within and across samples. Code and data are available in `https://github.com/realcrane/Bayesian-Differentiable-Physics-for-Cloth-Digitalization`.

4.1 Introduction

Understanding physics through deep learning has achieved great successes recently (Wu et al. 2015; Wu et al. 2016; Bear et al. 2022). One challenge in this domain is to infer fabric physical properties from images/videos/meshes (Sanchez-Gonzalez et al. 2020) because fabrics, unlike rigid bodies, are highly deformable and often show strong material heterogeneity in dynamics. Given that estimating fabric physical parameters is a key task in a variety of areas, the recent attempt to employ deep learning has spiked huge interest (Bouman et al. 2013; Xiao and Wang 2019; Rasheed et al. 2020), because it can potentially bypass the existing fabric testing protocols, e.g., KES-F (Kawabata 1980) and FAST (Minazio 1995) which are expensive, slow and laborious, and therefore revolutionize several industries such as textile and fashion.

Despite the initial successes, there is a notable difference between the data collection processes in current deep learning methods and the aforementioned industries. Compared with the widely employed measuring protocols in textile where various variables are strictly controlled, e.g., temperature, air moisture, current deep learning methods mainly learn from the data captured in less-controlled settings (Bouman et al. 2013; Wang et al. 2011). Admittedly, this enables simpler data collection, but the learned results are merely sufficient for general motion prediction/simulation (Yang et al. 2017; Narain et al. 2012), and are far from accurate for detailed simulation, manufacturing, design, etc (Gao and Chen 2021; Delavari and Dabiryan 2021; Luible and Magnenat-Thalmann 2008).

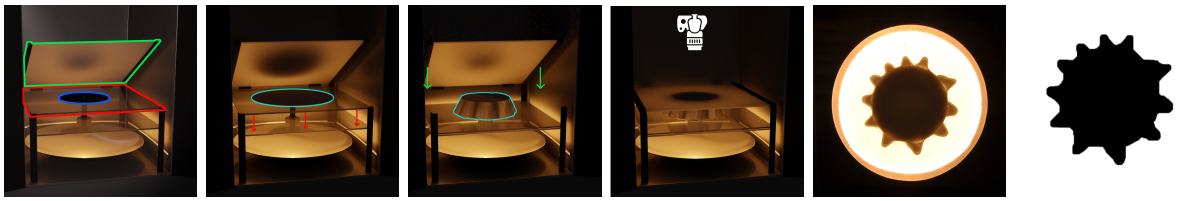
We aim to fill this gap by proposing a new method that can incorporate data from widely employed textile testing protocols and accurately infer the physical parameters of fabrics. For data, we use Cusick drape testing under the British Standard (British Standards Institute 2008) which has been proven to be highly rigorous. At a high-level, Cusick drape testing captures the silhouette of a static draping fabric sample in an image and uses this image as features to evaluate fabric's drapability. Therefore, it is a vision-based approach which is machine-learning

friendly and has been proven effective in describing the fabric physical properties (Chu et al. 1950; Cusick 1961; Cusick 1965; Cusick 1968; Collier 1991; Collier et al. 1989; Jeong 1998; Jeong and D. G. 1998).

However, adapting the existing methods (Sanchez-Gonzalez et al. 2020; Liang et al. 2019; Gong et al. 2022; Li et al. 2022a) for Cusick drape data is difficult. First, the existing methods often overly simplify the material, e.g., assuming homogeneous materials or/and negligible cross-sample material variations (Liang et al. 2019; Gong et al. 2022). In reality, not only does the material of fabric samples of the same type show variations, but the mechanical heterogeneity in different parts of the same sample is also obvious. Learning homogeneous material naturally leads to inaccurate parameter estimation. Further, most existing methods often need a large amount of training data, especially black-box approaches (Sanchez-Gonzalez et al. 2020). Collecting such data is prohibitively time-consuming and labour intensive for the Cusick drape test. Compared with hours of videos used by existing methods, there are few public datasets of Cusick drape. So we collect our own data, but the dataset is not even remotely as large as videos (Runia et al. 2020). Finally, a particular challenge we face is that a standard Cusick drape meter only provides the final image of a fabric drape, i.e., no 3D geometry or motion is captured, ruling out the methods (Sanchez-Gonzalez et al. 2020; Runia et al. 2020) that require such observations.

To overcome these challenges, we propose a new Bayesian learning scheme. Starting from the joint probability of the observed Cusick drape images and the initial states of fabric samples, we first introduce latent variables to represent the probabilistic fabric state transitions during draping and the underlying fabric physical properties. Due to the limited data (i.e., one image per drape), inferring the latent variables is a formidable task. Therefore, we propose to replace the probabilistic state transitions with a deterministic mapping which is enabled by our new differentiable heterogeneous fabrics model. Owning to its high sample efficiency, this model can already learn the heterogeneous material from extremely limited data (i.e., merely one image) of a draping sample. Further, to account for the cross-sample material variations, we impose learnable priors over the material parameters, leading to a new Bayesian differentiable fabrics model, which can learn the probabilistic distributions of type-specific physical parameters.

Through exhaustive evaluations, we show that our method is *accurate* in inferring highly plausible physical parameters, *efficient* in training with limited data, and *general* in capturing physical



(a) Tester (b) Initial State (c) Drape State (d) Capturing (e) Raw Photo (f) Silhouette

Figure 4.2: Cusick drape testing. The Cusick drape meter has an inner support panel (blue), an outer support panel (red) and a frosted glass lid (green). A round cloth is first laid flat on the support panels (light blue in Initial State). Then the outer support panel is lowered to allow the cloth to naturally drape (Drape State). Next, the glass lid is closed so that the light source at the bottom can project the cloth to the lid which is recorded by a camera at the top (Capturing). Finally, the cloth Silhouette is extracted from the Raw Photo. The whole Tester is in a black chamber so the testing process is not observable.

parameter variations across fabric samples. Since there is no existing deep learning method designed for similar tasks, we compare our method with possible alternative solutions including different fabric models and optimization methods. Formally, our contributions include:

- a new method for fabric material parameter estimation based on limited Cusick drape data.
- a new Bayesian differentiable fabric model to enable accurate physical parameter estimation.
- a new dataset collected from the standard Cusick drape testing.

4.2 Related Work

Fabric Drape Measurement and Simulation. Drapability is an important fabric characteristic (El Messiry and El-Tarfawy 2020). Since the first drapability measurement (Peirce 1930), many methods have been proposed, among which the Cusick drape test (Chu et al. 1950; Cusick 1965; Cusick 1965; Cusick 1968) is the most widely acknowledged one (British Standards Institute 1973; British Standards Institute 2008; British Standards Institute 1998). Refer to the works proposed by Sanad et al. (2012) and Sanad et al. (2013) for more details. Further, studies have shown strong correlations between fabric physical properties and drapability (Treloar 1965; Collier 1991; Lojen and Jevsnik 2007), which inspires the design of simulators (Breen et al. 1994b) and the models for drape prediction (Ghith et al. 2015; Stylios et al. 2002). Recently, there are some attempts in estimating the fabric physical parameters from Cusick drape (Kim 2011; Kenkare et al. 2008; Pandurangan et al. 2008; Ju and Choi 2020; Ju et al. 2022). However,

their fabric models cannot account for two commonly observed phenomena in Cusick drape testing: asymmetric draped shapes and cross-sample variation in draped shapes. Al-Gaadi et al. (2012) attribute the asymmetry to hysteresis caused by fabric internal frictions and adds a viscoelastic model to simulate it. However, capturing cross-sample drape shapes variation is still an open problem. Our model aims to capture such variation.

Cloth Parameter Inference. Machine learning has been used to estimate cloth physical parameters and the methods falls into three types: black-box models, physics-based models, and white-box models. Black-box models map observed cloth dynamics (e.g., from videos or simulations) to parameters without explicitly representing the underlying physics (Yang et al. 2017; Rasheed et al. 2020; Ju and Choi 2020; Ju et al. 2022). Physics-based models incorporate physical cloth simulators and search for feasible simulation parameters by minimizing the difference between the simulation results and training data (Runia et al. 2020). Finally, white-box models explicitly model the underlying physics. Wang et al. (2011) propose a nonlinear anisotropic cloth model and use optimization to estimate cloth simulation parameters. In contrast, Liang et al. (2019) propose the first fully differentiable cloth simulator for estimating cloth parameters. More recently, Li et al. (2022a) introduce a differentiable cloth simulator to simulate the Stick-Slip contact friction. Jatavallabhula et al. (2021) append differentiable rendering modules to various physical simulators to learn parameters from video. However, these methods cannot be applied to the standard Cusick drape data. Black-box models do not include physical parameters at all and need large amounts of data to train. Both physics-based and white-box models can learn physical parameters, but they require dynamics information. By contrast, our model estimates fabric physical parameters from a small number of standard Cusick drape images.

Physics-based Deep Learning. Our research can be seen as a part of recent attempts in leveraging deep learning to solve differential equations, which has spiked research interests to address issues such as noise modeling, finite element mesh generation and high dimensionality (Karniadakis et al. 2021; Lu et al. 2021; Beck et al. 2023). Deep Neural Networks (DNNs) can learn to generate Finite Element meshes for steady state problems (Zhang et al. 2020; Zhang et al. 2021). Also, they can be part of Partial Differential Equations (PDEs) for purposes such as reduced-order modeling (Shen et al. 2021; Han et al. 2018), noise estimation (Yang et al. 2021) and differentiable simulation (Gong et al. 2022; Holl et al. 2020). Further, DNNs can replace

PDEs completely in physics-informed neural networks (Raissi et al. 2019) where the process of solving PDEs is replaced by inference on trained DNNs. Different from existing work, we propose a Bayesian differentiable physics model for fabrics to explicitly learn the stochasticity of the fabric physical parameters.

4.3 Methodology

We first illustrate our Cusick Drape testing workflow Section 4.3.1. Then, we introduce our Bayesian Differentiable Physics model Section 4.3.2 and the inference method for learning from our Cusick Drape testing results Section 4.3.4. Finally, Section 4.3.5 ends this section by an introduction to our implementation.

4.3.1 Cusick Drape Test

Our Cusick Drape meter comes with a chamber within which there are two support panels and one frosted glass lid (Fig. 4.2a). During testing, we first cut a fabric sample into a round shape (Fig. 4.3 (a)) and pin its center to the center of the blue panel in Fig. 4.2a (diameter is 18cm), shown in Fig. 4.2b. Then we lower the transparent panel (the red panel in Fig. 4.2a) and let the fabric naturally drape until the transparent panel does not contact with the fabric (Fig. 4.2c). Finally, an image $\mathcal{I} \in \{pix \in \mathbb{Z} : 0 \leq pix \leq 255\}^{L \times L}$ (Fig. 4.2e) is taken by a DSLR camera from the top (Fig. 4.2d). To minimize possible external perturbations, the chamber is closed when capturing.

4.3.2 A Bayesian Model for Cusick Drape

We discretize a fabric sample's geometry into a triangular mesh with v vertices (Fig. 4.3(b)). Then we define the sample's state as $\mathcal{S}_t = \{\mathbf{x}_t, \dot{\mathbf{x}}_t\}$ where $\mathbf{x}_t \in \mathbb{R}^{3\times v}$ and $\dot{\mathbf{x}}_t \in \mathbb{R}^{3\times v}$ are the vertices position and velocity respectively at time t. Therefore, a draping motion with discretized time is $\mathcal{S}_{0:n} = \{\mathcal{S}_t : t \in \mathbb{Z}^+; t \leq n\}$, with a time step size h. Since we only observe the final image $\mathcal{I}$ and the initial state $\mathcal{S}_0$, their joint probability is:

$$p(\mathcal{I}, \mathcal{S}_0) = \int \cdots \int p(\mathcal{I}|\mathcal{S}_n, \tau) \prod_{i=0}^{n-1} p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau) p(\tau) d\mathcal{S}_{1:n} d\tau \tag{4.1}$$

where we introduce two sets of latent variables, τ and $\mathcal{S}_{1:n}$. $\mathcal{S}_{1:n}$ is the intermediate states of the draping motion which we cannot directly observe. Since the draping is a physical process,

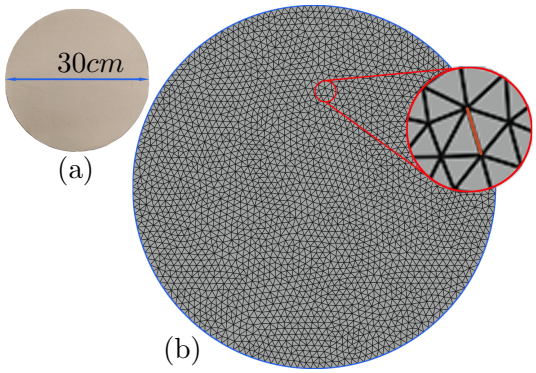


Figure 4.3: (a) A circular (diameter=30cm) fabric sample for Cusick drape test. (b) Fabric sample mesh generated in FreeFEM++(Hecht 2012) by Delaunay triangulation which has 2699 vertices, 7924 edges (7754 bending edges), and 5226 faces. A **bending edge** (highlighted in orange) is shared between two adjacent triangles, so the edges highlighted in blue (the boundary) are not bending edges.

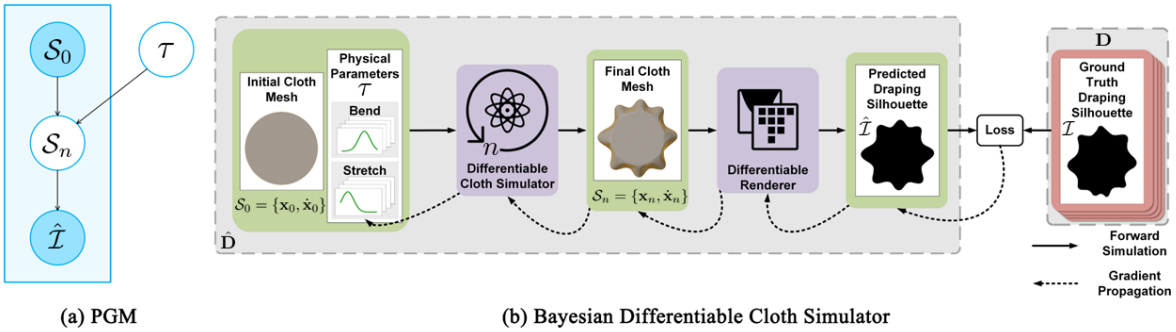


Figure 4.4: (a) The **Probabilistic Graphical Model** (PGM) of our Bayesian Differentiable Simulator. (b) **Model overview.** The physical parameters (stretching stiffness, bending stiffness) are first drawn from their learnable priors. Then the parameters and the cloth initial state are fed to a differentiable cloth simulator to run and predict cloth's final state $\mathcal{S}_n = \{\mathbf{x}_n, \dot{\mathbf{x}}_n\}$. The cloth in the final state is passed to a differentiable renderer. The rendered cloth silhouette is compared with the ground truth to compute the loss for back-propagation to update the parameters in the priors.

it is reasonable to assume $\mathcal{S}_t$ is only affected by $\mathcal{S}_{t-1}$ and τ (Markov assumption). Additionally, the captured image $\mathcal{I}$ is only decided by the final state $\mathcal{S}_n$.

Eq. (4.1) is not easy to estimate due to two challenges. First, unlike the prior works which depend on dense observations on the intermediate state transitions (Yang et al. 2017) to estimate $p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau)$, Cusick drape testing does not capture the fabric sample's motion. Also, we do not observe the full $\mathcal{S}_n$, but only its (simplified) 2D representation $\mathcal{I}$.

To this end, we assume two deterministic mappings can be established for $p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau)$ and $p(\mathcal{I}|\mathcal{S}_n, \tau)$. The determinism assumptions are reasonable as $p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau)$ can be seen as a quasi-deterministic physical process, subject to minor system stochasticity which are largely mitigated

by the rigorous control of Cusick settings. $p(\mathcal{I}|\mathcal{S}_n, \tau)$ can be seen as a rendering process studied in computer graphics, where we are mainly interested in the silhouette. Therefore, we replace $p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau)$ with a state transition function $\mathcal{S}_{i+1} = s(\mathcal{S}_i, \tau)$ where s is a deterministic function then $p(\mathcal{S}_{i+1}|\mathcal{S}_i, \tau) = p(s(\mathcal{S}_i, \tau)|\mathcal{S}_i, \tau) = 1$. Similarly, $p(\mathcal{I}|\mathcal{S}_n, \tau) = p(r(\mathcal{S}_n)|\mathcal{S}_n, \tau) = 1$ where r is a rendering function. Therefore, Eq. (4.1) is transformed to:

$$p(\mathcal{I}, \mathcal{S}_0) = \int p(r(\underbrace{s \dots s}_{n}(\mathcal{S}_0, \tau)))p(\tau)d\tau \tag{4.2}$$

Given an initial state $\mathcal{S}_0$ and the observations $D = \{\mathcal{I}_1, , \mathcal{I}_2, , \dots\}$, we maximize $p(\mathcal{I}, \mathcal{S}_0)$ which is equivalent to finding the posterior distribution of fabric material parameters: $p(\tau|D) \propto p(D|\tau)p(\tau) = p(D|r \circ s \circ s \dots s(\mathcal{S}_0, \tau))p(\tau) = p(D|\hat{\mathcal{I}})p(\tau)$ where the composite function $\hat{\mathcal{I}} = r \circ s \circ s \dots s(\mathcal{S}_0, \tau)$ is deterministic. Overall, the corresponding Probabilistic Graphical Model (PGM) is illustrated in Fig. 4.4(a).

4.3.3 Model Specification

To infer $p(\tau|D)$, we need to instantiate s and r. For r, we use differentiable renderer DIR-B (Chen et al. 2019). Given a fabric sample's final draped state $\mathcal{S}_n$ which is a 3D mesh and the virtual camera pose, DIR-B converts it to a 2D image. We set up the virtual camera pose (relative to the fabric drape) according to the real camera in our Cusick drape meter, so that the images captured in the Cusick drape test can be directly used as training data. Additionally, we only use drape silhouettes and ignore other information such as textures because it is irrelevant to fabric drapability (Cusick 1961; Cusick 1965; Cusick 1968).

A Bayesian Differentiable Fabrics Model

The instantiation of s is more complex than r. We propose to use a differentiable fabric model for s so that we can use back-propagation for learning. However, there are only a few differentiable fabric models (Liang et al. 2019; Li et al. 2022a; Jatavallabhula et al. 2021; Gong et al. 2022) which, unfortunately, are unsuitable. Black-box models (Ju and Choi 2020) require large amounts of data. White-box models(Liang et al. 2019; Li et al. 2022a; Gong et al. 2022) usually model fabrics as homogeneous materials that cannot capture fabrics' complex non-linear anisotropic deformation in Cusick drape. Further, since their models do not consider material stochasticity (neither within-sample nor cross-sample), fitting one model across different images

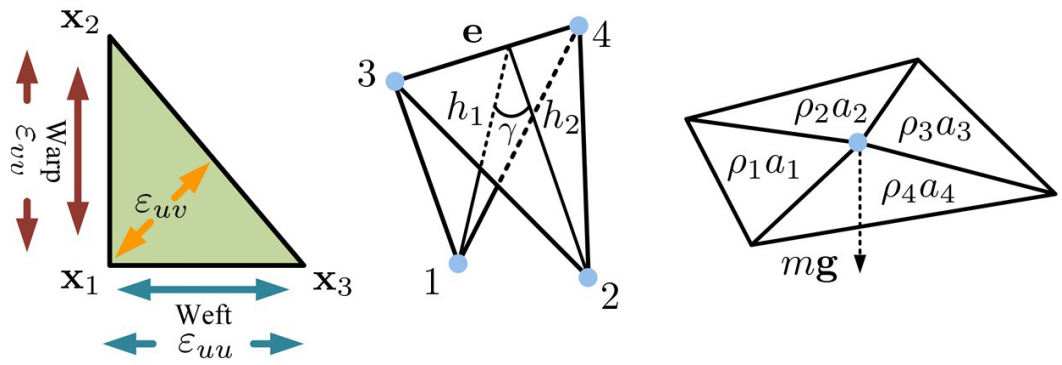


Figure 4.5: Left: cloth weft and warp directions (or course and wale directions in knitted fabrics) and three strain directions in a triangle. Middle: bending force between two adjacent triangles. Right: triangle mass and gravity.

essentially equivalents to finding an average material. Therefore, we build a stochastic heterogeneous model, where we coin the term Bayesian differentiable fabrics model. We give the key equations below.

A physics-based fabric simulator aims to simulate a fabric motion by recurrently predicting its future state $\mathcal{S}_{t+1} = \{\mathbf{x}_{t+1}, \dot{\mathbf{x}}_{t+1}\}$ given the current state $\mathcal{S}_t = \{\mathbf{x}_t, \dot{\mathbf{x}}_t\}$:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + h\dot{\mathbf{x}}_t \tag{4.3}$$

$$\dot{\mathbf{x}}_{t+1} = \dot{\mathbf{x}}_t + h\ddot{\mathbf{x}}_t \tag{4.4}$$

where h is the time step size (time lapse between every two consecutive states) and the second-order time derivative, $\ddot{\mathbf{x}}_t$, is vertices acceleration. To gain high simulation stability, implicit Euler method (Baraff and Witkin 1998) is commonly used:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + h\dot{\mathbf{x}}_{t+1} \tag{4.5}$$

$$\dot{\mathbf{x}}_{t+1} = \dot{\mathbf{x}}_t + h\ddot{\mathbf{x}}_{t+1} \tag{4.6}$$

According to Newton's Second law, we have

$$\mathbf{F} = \mathbf{M}\mathbf{a} = \mathbf{M}\ddot{\mathbf{x}} \tag{4.7}$$

where $\mathbf{M}$ is the general mass matrix and $\mathbf{F}$ is the resultant force which is the combination of internal and external forces. In our differentiable fabric simulator, these forces are decided by

fabric sample's current state, so we can define:

$$\mathbf{F}_t = \mathbf{f}(\mathcal{S}_t) = \mathbf{f}(\mathbf{x}_t, \dot{\mathbf{x}}_t) \tag{4.8}$$

where $\mathbf{f}$ denotes a general function, which must be differentiable, takes as input the current state, $\mathcal{S}_t$, and outputs the resultant force. Through Taylor approximation (Baraff and Witkin 1998), Eq. (4.5) and Eq. (4.6) are converted to the governing equation of the physical system:

$$\left(\mathbf{M} - h\frac{\partial \mathbf{f}}{\partial \dot{\mathbf{x}}} - h^2\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)\Delta\dot{\mathbf{x}} = h\left(\mathbf{F}_t + h\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\dot{\mathbf{x}}_t\right) \tag{4.9}$$

which needs to be solved to calculate $\Delta\dot{\mathbf{x}}$ and update the fabric sample's state:

$$\dot{\mathbf{x}}_{t+1} = \dot{\mathbf{x}}_t + \Delta\dot{\mathbf{x}} \tag{4.10}$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + h\dot{\mathbf{x}}_{t+1} \tag{4.11}$$

In our differentiable fabric simulator, the resultant force consists of all the internal forces and external forces: $\mathbf{F} = \mathbf{F}_{gravity} + \mathbf{F}_{handle} + \mathbf{F}_{stretch} + \mathbf{F}_{bend}$. The $\mathbf{F}_{gravity}$ is simply gravity and $\mathbf{F}_{handle}$ is the force for pinning and supporting a fabric sample, i.e., simulating the inner support panel (Fig. 4.2a). The gravity, $\mathbf{F}_{gravity}$, is calculated on every face and evenly divided by its three vertices. Therefore, the gravity on the kth vertex is

$$\mathbf{F}^{(k)}_{gravity} = m^{(k)}\mathbf{g} = \sum_{j=0}^{n_j^{(k)}} \frac{1}{3}\rho^{(j)}A^{(j)}\mathbf{g} \tag{4.12}$$

where $\rho^{(j)}$ is the jth face's area density and $\mathbf{g} = [0.0, 0.0, -9.8]^\top m/s^2$. $n_j^{(k)}$ is number of the adjacent faces of vertex k (Fig. 4.5 Right). The handle force, $\mathbf{F}_{handle}$, is used to pin and support a fabric sample, i.e., simulating the inner support panel. To each vertex whose distance to the center of the fabric is smaller than 9cm, the handle force is computed as:

$$\mathbf{F}^{(k)}_{handles} = k_h\mathbf{I}_3(\mathbf{x}^{(k)} - \bar{\mathbf{x}}^{(k)}) \tag{4.13}$$

where $\bar{\mathbf{x}}^{(k)}$ denotes the kth vertex's anchor position where the vertex should be fixed and k_h is the handle stiffness , and $\mathbf{I}_3$ denotes a 3-by-3 identity matrix.

Unlike previous methods (Liang et al. 2019; Gong et al. 2022), we model a material variation

across the mesh which is discretized by finite elements. This allows us to localize the learning to each element, i.e., making the learning of $\mathbf{F}_{stretch}$ and $\mathbf{F}_{bend}$ dependent on local deformation. The stretching force (Volino et al. 2009) on a face j is:

$$\mathbf{F}^{(j)}_{stretch} = -A^{(j)} \left(\sum_{m \in (uu,vv,uv)} \sigma^{(j)}_m \left(\frac{\partial \varepsilon^{(j)}_m}{\partial \mathbf{x}_i} \right) \right) \tag{4.14}$$

where $\varepsilon^{(j)}_m$ denotes stretching strain and the $\mathbf{x}_i$ are the three vertices of the face. The stretching stresses $\sigma^{(j)}_m$ is

$$\begin{bmatrix} \sigma^{(j)}_{uu} \\ \sigma^{(j)}_{vv} \\ \sigma^{(j)}_{uv} \end{bmatrix} = \begin{bmatrix} c^{(j)}_{11} & c^{(j)}_{12} & 0 \\ c^{(j)}_{12} & c^{(j)}_{22} & 0 \\ 0 & 0 & c^{(j)}_{33} \end{bmatrix} \begin{bmatrix} \varepsilon^{(j)}_{uu} \\ \varepsilon^{(j)}_{vv} \\ \varepsilon^{(j)}_{uv} \end{bmatrix} = \mathbf{C}^{(j)} \boldsymbol{\varepsilon}^{(j)} \tag{4.15}$$

where $c^{(j)}_{11}$, $c^{(j)}_{12}$, $c^{(j)}_{22}$, and $c^{(j)}_{33}$ are the stretching stiffness in weft/course direction, the stretching stiffness in warp/wale direction, Poisson's ratio, and shearing stiffness. The subscripts uu, vv, and uv denote fabrics weft/course, warp/wale, and diagonal directions respectively (Fig. 4.5 Left). $\boldsymbol{\varepsilon} = [\varepsilon^{(j)}_{uu}, \varepsilon^{(j)}_{vv}, \varepsilon^{(j)}_{uv}]$ is the Voige form strain tensor. The bending force around a bending edge w is defined as

$$\mathbf{F}^{(w)}_{bend} = \mathbf{B}^{(w)} \frac{|\mathbf{e}^{(w)}|}{h^{(w)}_1 + h^{(w)}_2} \sin(\frac{\gamma^{(w)}}{2} - \frac{\bar{\gamma}^{(w)}}{2}) u_i \tag{4.16}$$

where $\mathbf{B}^{(w)}$ is the bending edge w's bending stiffness, $|\mathbf{e}^{(w)}|$ is bending edge's rest length, $h^{(w)}_1$ and $h^{(w)}_2$ are the heights of the two adjacent triangular face, $\gamma^{(w)}$ and $\gamma^{\bar{(w)}}$ are the current and predefined rest dihedral angles between the edge's two adjacent faces (Fig. 4.5 Middle). Refer to work proposed by Bridson et al. (2003) for the u's.

Material non-linearity means the material stiffness changes with deformation magnitude non-linearly. Anisotropy refers to the varied material stiffness in different deformation directions. To encode fabric material non-linearity and anisotropy, our model adopts the piecewise linear physical models in the work proposed by Wang et al. (2011) where the stretching stiffness and bending stiffness are defined as two high-dimensional matrices: $\mathbf{C} \in \mathbb{R}^{6\times4}$ and $\mathbf{B} \in \mathbb{R}^{3\times5}$. Then, local stretching stiffness and bending stiffness are sampled from $\mathbf{C}$ and $\mathbf{B}$ according to the mesh's local deformation and geometry. Wang et al. (2011) model fabrics as continuum elastic shells so a stretching deformation can be described by the Green-Lagrangian strain tensor (Bonet and

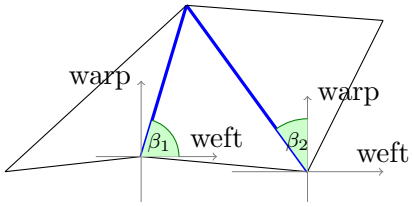


Figure 4.6: The bending bias angles, β_1 and β_2, of the two bending edges (blue).

Wood 2008), which can be re-parameterized by Eigen Decomposition:

$$2\begin{bmatrix}\varepsilon_{uu}^{(j)} & \varepsilon_{uv}^{(j)}\\ \varepsilon_{uv}^{(j)} & \varepsilon_{vv}^{(j)}\end{bmatrix} = (\mathbf{R}_{\varphi}^{(j)})^{\top}\begin{bmatrix}(\lambda_{max}^{(j)}+1)^2-1 & 0\\ 0 & (\lambda_{min}^{(j)}+1)^2-1\end{bmatrix}\mathbf{R}_{\varphi}^{(j)} \tag{4.17}$$

where the eigenvalues indicate the stretching deformation magnitude and $\mathbf{R}_{\varphi}^{(j)}$ is a rotation matrix that indicates the direction of the stretching deformation. $\lambda_{min}^{(j)}$ can be ignored because they find it has less influence on the stretching stiffness. The rotation matrix is decided by the bias angle, φ, between a fabric sample's rest warp-weft coordinate system and its deformed local coordinate system (Peng and Cao 2005). This way, the stretching non-linearity and anisotropy can be encoded as the stiffness which changes with parameters in the 2D space spanned by $\lambda_{max}^{(j)}$ and φ. As in the cloth simulator proposed by Wang et al. (2011), we sample 6 data points (the 6 rows in the matrix $\mathbf{C}$) in the polar space spanned by λ_{max} and φ where each data point contains c_{11}, c_{12}, c_{22}, and c_{33} which compose the 4 columns in the matrix $\mathbf{C}$. (We ignore the face index superscript, (j), to denote the general form.)

To model non-linear bending stiffness, the variables in Eq. (4.16) can represent the bending deformation so we define a parameter α:

$$\alpha^{(w)} = \frac{\sin(\frac{\gamma^{(w)}}{2} - \frac{\bar{\gamma}^{(w)}}{2})}{h_1^{(w)} + h_2^{(w)}} \tag{4.18}$$

which is related to the curvature. To model the bending anisotropy, we define another parameter called bending bias angle, i.e., the angle between a bending edge and fabric warp-weft coordinate system's axes, which indicates bending deformation direction (shown in Fig. 4.6). Therefore, the bending non-linearity and anisotropy can be encoded as the stiffness which changes with the parameter in a polar space spanned by α and bending bias angle. We sample 5 α's and 3 bending bias angles (0°, 45°, and 90°) which are the 5 columns and the 3 rows of bending stiffness matrix $\mathbf{B}$.

Finally, to model the material heterogeneity, each mesh face and bending edge are associated with a $\mathbf{C}$ and $\mathbf{B}$. Therefore, for a fabric consisting of f faces and e bending edges, the learnable parameters are f stretching matrices and e bending stiffness matrices (the HETER model in our experiments). To our Bayesian differentiable fabric simulator, each $\mathbf{C}$ and $\mathbf{B}$ are sampled from the variational distribution $q_\theta(\tau)$. Its learnable parameter is the distribution parameter θ of $q_\theta(\tau)$. As we assume $q_\theta(\tau)$ is distributed as a Gaussian, θ consists of the means and the variances of the stretching stiffness and bending stiffness: $2 \times 4 \times 6 + 2 \times 3 \times 5 = 78$ learnable parameters.

Now we replace the deterministic mapping s in Eq. (4.2) by solving Eq. (4.9) forward in time, so that Eq. (4.2) considers physical parameter within-sample variation. But it still cannot learn cross-sample variations. Therefore, a prior $p(\tau)$ (Eq. (4.2)) is used as a belief of the parameter distribution and a posterior $p(\tau|D)$ is learned through inference.

4.3.4 Model Inference

Directly estimating $p(\tau|D)$ is computationally intractable. We adopt variational inference (Hoffman et al. 2013) to seek an variational distribution $q_\theta(\tau)$ parameterized by θ, to approximate the true posterior $p(\tau|D)$ by minimizing the Kullback-Leibler divergence between them:

$$\begin{aligned}
\theta &= \underset{\theta}{\operatorname{argmin}}\, D_{\mathbb{KL}}(q_\theta(\tau)\|p(\tau|D)) \\
&= \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{q_\theta(\tau)}\left[\log q_\theta(\tau) - \log\left(\frac{p(D|\tau)p(\tau)}{p(D)}\right)\right] \\
&= \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{q_\theta(\tau)}[\log q_\theta(\tau) - \log p(D|\tau)p(\tau)] + \underbrace{\log p(D)}_{\text{const}} \\
&\equiv \underset{\theta}{\operatorname{argmin}}\, \underbrace{\mathbb{E}_{q_\theta(\tau)}[\log q_\theta(\tau) - \log p(D|\tau)p(\tau)]}_{\mathcal{L}(\theta|\mathcal{D},\tau)}
\end{aligned} \tag{4.19}$$

Calculating $\mathcal{L}(\theta|\mathcal{D},\tau)$, negative evidence lower bound, is computationally prohibitive. So we approximate it by Monte Carlo sampling:

$$\mathcal{L}(\theta|\mathcal{D},\tau) \approx \sum_{i=1}^{m} \big(\underbrace{\log q_\theta(\tau_i) - \log p(D|\tau_i)p(\tau_i)}_{l(\theta,\tau)}\big) \tag{4.20}$$

where τ_i denotes the ith Monte Carlo sample from the variational posterior distribution $q_\theta(\tau)$. Moreover, we assume that the fabric physical parameters are distributed as a Gaussian $\tau \sim$

Algorithm 1 Bayesian Differentiable Physics (BDP) cloth parameter estimation

Input: initialize $\theta_1 \leftarrow (\mu, \eta)$
for $e \leftarrow 0, num\_epochs$ **do**
  for $s \leftarrow 0, num\_samples$ **do**
    Initialization $\mathcal{S}_0 = \{\mathbf{x}_0, \dot{\mathbf{x}}_0\}$
    Sample $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\tau = \mu + \log(1 + \exp(\eta)) \odot \epsilon$
    for $t \leftarrow 1, n$ **do**
      Simulation $\mathcal{S}_t = \text{SIM}(\mathcal{S}_{t-1}, \tau)$
    end for
    $\mathcal{I}_s = \text{DIB-R}(\mathbf{x}_T)$
    $l_s = \log q(\tau|\theta) - \log p(\tau) - \log p(\mathcal{I}_s|\tau)$
  end for
  $\mathcal{L} = \sum l_s$
  Calculate gradient $\frac{\partial \mathcal{L}}{\partial \theta}$
  Update $\theta_e = \text{Adam}(\theta_{e-1}, \frac{\partial \mathcal{L}}{\partial \theta})$
end for
Output: θ_e

$\mathcal{N}(\mu, \Sigma)$ where Σ is a diagonal matrix. To enable stochastic gradient back-propagation, we adopt the re-parameterization trick(Blundell et al. 2015; Kingma and Welling 2022) to sample physical parameters $\tau = t(\epsilon, \theta)$ by shifting a stochastic parameter-free noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ through the deterministic function $t(\epsilon, \theta) = \mu + \log(1 + exp(\eta)) \odot \epsilon$ where variational parameters $\theta = \{\mu, \eta\}$. Consequently, the variational distribution q_θ is sought within the Gaussian family and the prior $p(\tau)$ is an isotropic Gaussian distribution with fixed parameters. Additionally, the output distribution is also a Gaussian $\mathcal{N}(\mu_I, \sigma^2)$ whose mean depends on predicted image $\hat{\mathcal{I}}$, i.e., $\mu_I = \hat{\mathcal{I}}$. The variance, σ^2, is fixed and used to control the tolerance to residual error. Therefore, the negative log likelihood $-\log p(D|\tau)$ is:

$$-\sum_{i=1}^{L}\sum_{j=1}^{L} \log\left[\left(\frac{1}{2\pi\sigma^2}\right)^{\frac{1}{2}} e^{-\frac{1}{2\sigma^2}(\mathcal{I}_{ij} - \hat{\mathcal{I}}_{ij})^2}\right] \tag{4.21}$$

which is essentially proportional to the Mean Squared Error (MSE). In back-propagation, the gradients of the variational distribution parameters are calculated by (Blundell et al. 2015):

$$\frac{\partial l(\theta, \tau)}{\partial \mu} = \frac{\partial l(\theta, \tau)}{\partial \tau} + \frac{\partial l(\theta, \tau)}{\partial \mu} \tag{4.22}$$

$$\frac{\partial l(\theta, \tau)}{\partial \eta} = \frac{\partial l(\theta, \tau)}{\partial \tau} \frac{\epsilon}{1 + e^{-\eta}} + \frac{\partial l(\theta, \tau)}{\partial \eta} \tag{4.23}$$

The whole inference process is shown in Algorithm 1.

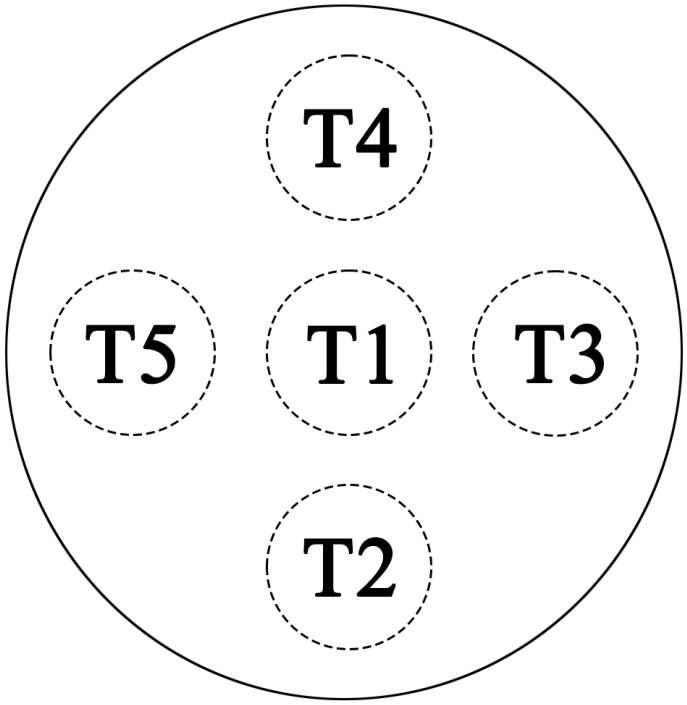


Figure 4.7: The average thickness of a cloth sample is the mean thicknesses measured in the five areas indicated in the figure: T1-5.

4.3.5 Implementation

Our differentiable fabric simulator is implemented in Pytorch's C++ frontend (Paszke et al. 2019). We exploit vectorization and CUDA GPU parallel computing for fast simulation and learning. We use Eigen's sparse solver (Guennebaud, Jacob, et al. 2010) to solve the governing equations Eq. (4.9). To reduce the memory consumption, we use sparse matrices whenever possible. Moreover, we do in-place gradient update for every time step for back-propagation, so our memory usage does not increase with simulation steps (Chen et al. 2016). We use Kaolin differentiable rendering package (Jatavallabhula et al. 2019) for image rendering.

4.4 Data Collection

The Cusick drape data is collected with following the BS EN ISO 9073-9:2008 (British Standards Institute 2008). In every test, fabric samples warp/wale and weft/course directions are aligned across samples to ensure the same initial condition. In addition, we also reconstruct the 3D meshes of fabric drapes. However, the 3D data are mainly for evaluation and the silhouette images are for learning. This is because not every Cusick drape meter can reconstruct 3D meshes out of fabric drapes. Being able to learn from the 2D silhouette images only is crucial in

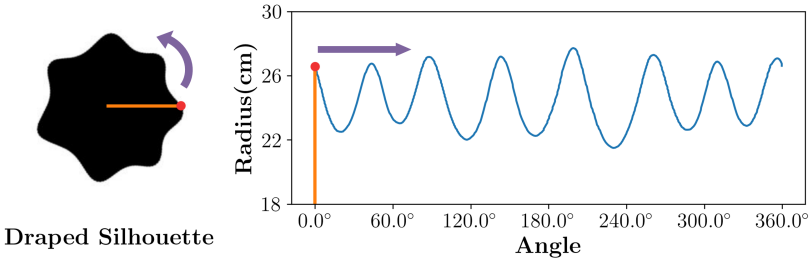
making our method applicable in real-world settings. In addition, we also measure the sample weight and calculate the average area density $\rho = (\sum_{i<13}^{i=1} m_i)/(12\pi R^2)$ where m_i denotes the measured weight of fabric sample i and $R = 0.15m$. We also measure the average thickness which is the mean thickness of the five areas (shown in Fig. 4.7) in one fabric sample.

Our current Cusick drape dataset includes 25 types of common fabrics, each of which with multiple samples. As listed in the table in Appendix B, these fabrics are different in material, woven pattern, area density (ρ), and average thickness. In our Cusick drape test, they also show distinctive drape shapes. We test the sample's two sides multiple times. In every test, our Cusick drape meter captures a drape image and reconstructs its 3D mesh. It took approximately more than 15 working days to do the textile testing. At the end, there are 660 drape images and meshes in our current dataset which will be released with our paper.

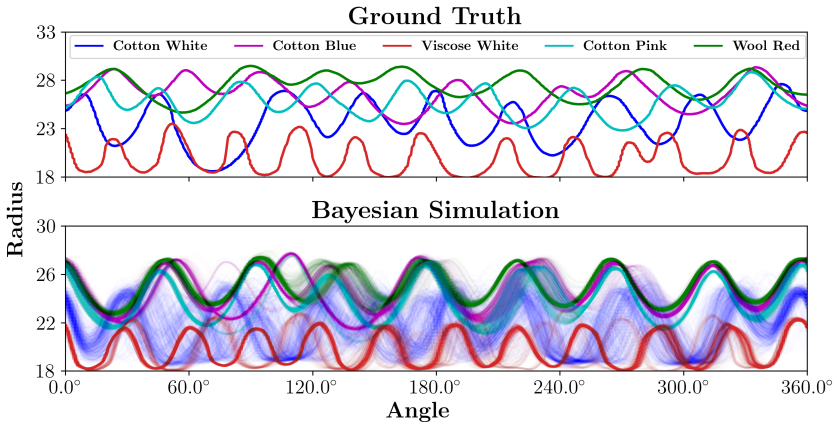
The lack of training data is the deciding factor that hinders the application of deep learning in textile. To our best knowledge, the only open source cloth drape dataset was proposed in (Feng et al. 2022) very recently. Compared with their dataset, our new dataset has multiples advantages. First, our dataset includes 25 types of common fabrics and has more training samples, which comprehensively covers a wide range of fabrics, with accurate description of the materials. The number of types of fabrics and details are unclear in their data. Second, their data only provides the estimated material parameters tied to their estimation methods, which might make it difficult to transfer them to a different cloth model. In contrast, not only do we provide the raw Cusick drape testing data (i.e., images and meshes), we also provide the estimated parameters. Furthermore, their parameters have specific values while ours are distributions that capture cloth material heterogeneity and dynamics stochasticity. Last but not the least, our Cusick drape testing rigidly follows the British standards and the textile measurements (e.g., measuring area density and thickness) are conducted in a rigorously controlled environment so that they are more accurate and easier to be reproduced/verified in future research. By contrast, their drape testing is measured by customized apparatus under a less controlled setting.

4.5 Experiments

In our experiments, we use 5 representative type of fabrics (each with 12 samples) that show visually distinguishable drapability. For convenience, we name fabrics as "material color", e.g., Cotton Blue, Cotton Pink. Fabric 1-5 in the table in Appendix B are used in our experiments



(a) An example of draping radius



(b) Clustered drape radius

Figure 4.8: a: an drape radius-angle graph illustrates the varied radius of an draped fabric sample's boundary w.r.t. angle (Kim 2011). b: the ground truth from five real fabric sample (top) and the simulated 500 samples from our BDP (bottom) for the corresponding samples. It demonstrates our trained BDP models can distinguish different fabrics and are not overly generalized.

which are the Cotton White, Cotton Blue, Viscose White, Cotton Pink, and Wool Red respectively. During learning, we run 100 steps ($n = 100$) for the forward simulation, with time step size $h = 0.05s$ and the total time lapse is $5s$. We employ several metrics for evaluation and comparison. For examining prediction accuracy, we use MSE between the predicted and the ground truth draped images as one metric. In addition, we also use Hausdorff distance (H.Dis) between the predicted 3D mesh and the ground-truth (GT) reconstructed mesh as another metric. Further, we use radius-angle graph (Fig. 4.8a) as another metric which is widely adopted for describing Cusick drape waves (Kim 2011) and comparing drape shapes Fig. 4.8a. In addition, `https://youtu.be/ProN0y1bURY` provides more visual results.

4.5.1 Bayesian Differentiable Fabric Model

Within a fabric type, we randomly select 5 out of 12 samples for training and the rest for testing (Fig. 4.1 (a) and (b)). We use $\mathcal{L}(\theta|D, \tau)$ in Eq. (4.20) as the loss function. After training, we draw parameters from the learned parameter distributions for 500 times and run simulations.

Table 4.1: The average evaluation MSE and H. Dis of HOMO, HETER and our BDP which demonstrate our BDP outperforms HOMO and HETER in generalization.

Metrics	HOMO	HETER	BDP
Avg MSE	5.68×10^{-2}	5.54×10^{-2}	5.38×10^{-2}
Avg H. Dis	3.32×10^{-2}	3.31×10^{-2}	2.49×10^{-2}

Fig. 4.1 shows the best simulated drape on one training and one testing sample. Visually, our method not only can accurately reproduce the training draped shape (Fig. 4.1 (c) and (e)), but can also predict unseen fabrics well across multiple samples (Fig. 4.1 (d) and (f)). The results demonstrate that our model can captures cross-sample variations. Moreover, the radius-variation (Fig. 4.8b) demonstrates the trained Bayesian models are not overly generalized and remains to be discriminative in different fabric types. In Fig. 4.8b, the top figure shows that the five real fabric samples have distinctive drape shapes. The bottom figure shows the drape shapes of samples from our BDP which largely follows the same patterns as the ground truth. For example, Wool Red is obviously stiffer than Viscose White, shown in the ground-truth. Likewise, it is also observed in our BDP's simulation at the bottom.

4.5.2 Comparison

Alternative Fabric Models To our best knowledge, there is no similar method designed for exactly the same setting as ours, we therefore adopt the closest methods as baselines. We adopt (Liang et al. 2019) because their method can also learn fabric physical parameters, albeit only taking simulated motion as input. Since they only model homogeneous material, we refer to their model as HOMO. Further, we augment their model by making the fabric physical parameter learning element-wise so as to enable learning heterogeneous physical parameters. We call this model HETER for short. Both HOMO and HETER are *deterministic* models. So we use MSE as the loss function: $\mathcal{L}(\tau) = \sum_{i=1}^{L} \sum_{j=1}^{L} (\hat{\mathcal{I}}_{ij} - \mathcal{I}_{ij})^2$, to learn the physical parameters from one single silhouette.

For comparison, we train HOMO and HETER with the same data (5 for training and the rest 7 for testing) as BDP. However, as in (Liang et al. 2019), HOMO and HETER can only fit data, but cannot predict on unseen data. We use their simulation results averaged over all 5 training samples as prediction for the rest 7 testing samples. For BDP, we sample the parameters and run multiple simulations, then select the best result to calculate the average MSE and H. Dis. As shown in Table 4.1, our BDP is better than the HOMO and HETER in generalization.

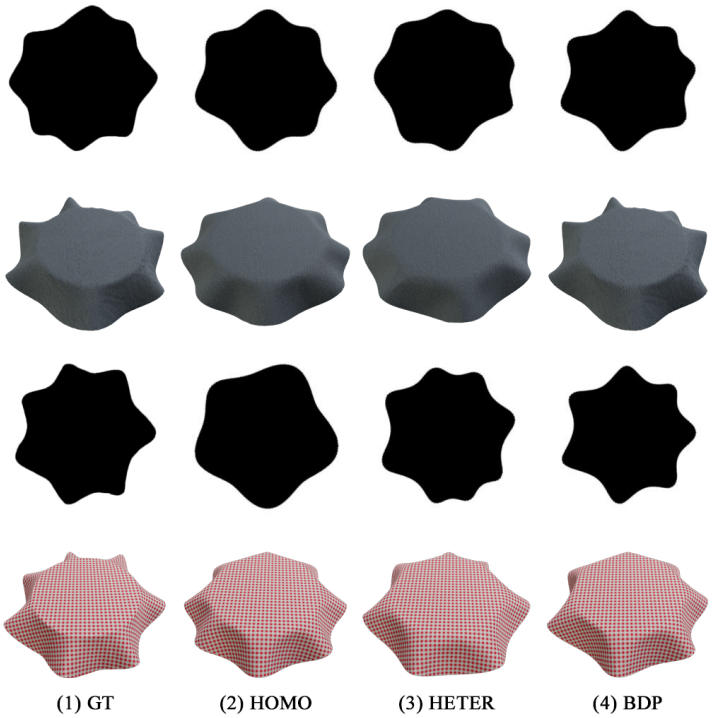


Figure 4.9: We select the most similar silhouettes (minimal MSE) and meshes (minimal H. Dis) to a testing fabric sample learned by HOMO, HETER, and BDP. In the figure, it is obvious that the BDP outperforms the HOMO and HETER

In addition, Fig. 4.9 also shows that only our BDP can simulate the drape shapes that are closer to the testing samples. This is not surprising because HOMO and HETER lack in the capacity to generalize to unseen samples, even when the testing samples are from the same type of fabrics. To further show the importance of modeling material heterogeneity across the same fabric sample, we compare HETER with HOMO in Fig. 4.10. It highlights that HETER can fit data much better than HOMO and demonstrates the necessity of modeling fabric material heterogeneity in learning from fabric Cusick drape.

Alternative Learning Methods While our method is built on differentiable physics models for learning, i.e., derivative-based optimization, the traditional methods widely used in material science and physics are usually based on derivative-free optimization. So we also compare different learning strategies.

Bayesian Optimization (BO) is a representative derivative-free optimizer which usually uses Gaussian Process to approximate an unknown optimized objective function (Brochu et al. 2010; Frazier 2018). However, the performance of vanilla BO drops drastically when the number of parameters are above 20 (Eriksson and Jankowiak 2021; Letham et al. 2020). There are over

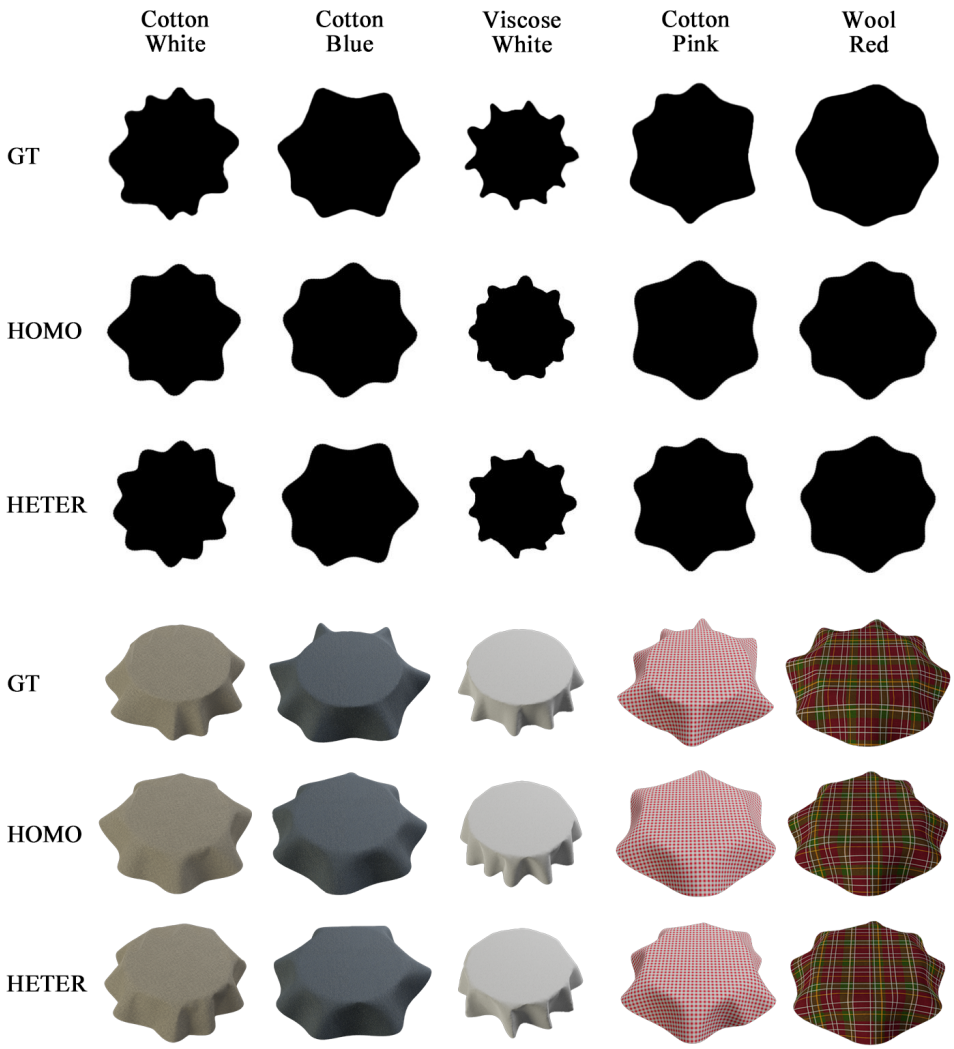


Figure 4.10: The first three rows show the ground truth silhouettes , the silhouettes learned by the HOMO and HETER respectively. The three rows below show the corresponding 3D meshes. It is obvious the learned drape by the HETER is closer to the ground truth across different fabrics.

200,000 parameters in our fabric model. So we employ Random Embedding Bayesian Optimization (Wang et al. 2016) and Hashing-enhanced Subspace Bayesian Optimization (Nayebi et al. 2019) as baselines. For short, we refer them as to REMBO and HeSBO, respectively. We use BoTorch's (Balandat et al. 2020) REMBO and HeSBO implementation and compare them with our method. Given the same drape silhouette, we run our method 100 epochs, and the REMOB and HeSBO for 500 trials. Table 4.2 shows that, our derivative-based method is better with fewer optimization steps. Fig. 4.11 also illustrates the drape shapes learned by our derivative-based method are closer to the ground truth.

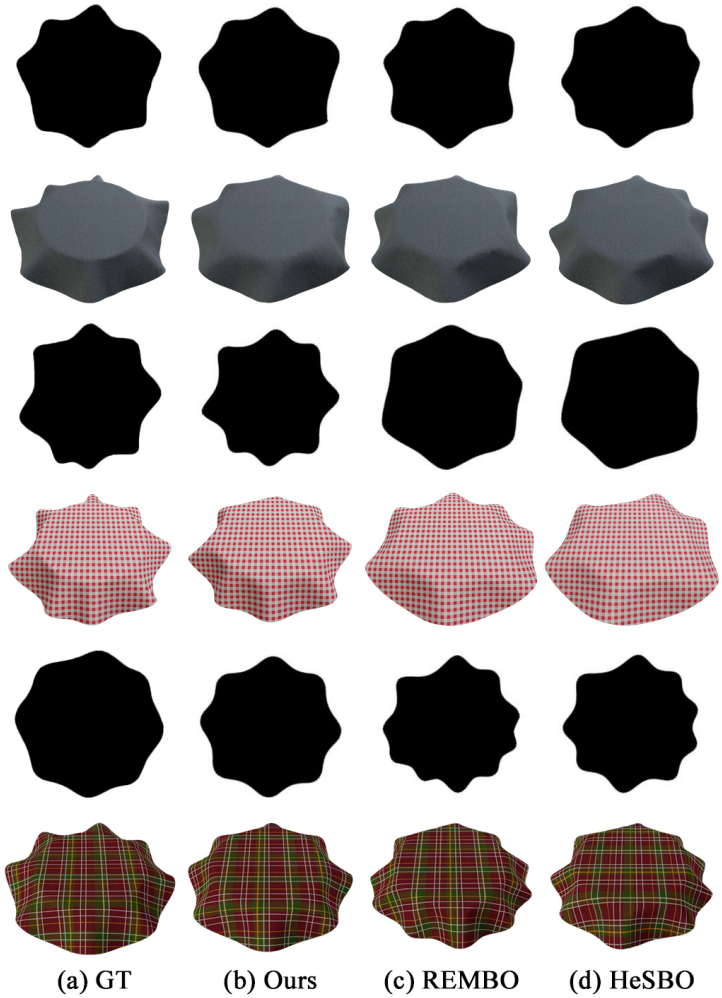


Figure 4.11: Comparison of ground truth draped shape and the simulated drape shapes learned by ours(b)(derivative-based), REMBO(c)and HeSBO(d)(derivative-free) optimization.

4.5.3 Material Heterogeneity

Our model attributes the within-sample shape variation to fabric material heterogeneity. To demonstrate it, we use the HETER (deterministic heterogeneous model) to learn the drape shapes of six samples (three from Cotton Blue and three from Viscose White). The ground-truth (GT) rows in Fig. 4.12 and Fig. 4.13 show that samples from the same fabric are obviously different. The learned (LR) rows in Fig. 4.12 and Fig. 4.13 demonstrate that the HETER can accurately learn from the different drape shapes. The heat maps in the last row illustrate the learned distributions of the fabric physical parameters across the meshes. This result demonstrates that within-sample drape shape variation can be accounted for by fabric material heterogeneity and varied physical parameter distributed across the mesh in different samples.

Table 4.2: The average MSE and H. Dis of ours, REMBO, and HeSBO optimizer. The results shows that our gradient-based optimization achieves better results.

Metrics	Ours	REMBO	HeSBO
Avg MSE	3.27×10^{-2}	5.40×10^{-2}	5.46×10^{-2}
Avg H. Dis	1.55×10^{-2}	3.15×10^{-2}	3.32×10^{-2}

Table 4.3: Comparing the run time between (Liang et al. 2019) (collision handling off) and our model in forward simulation and backward gradient computation. Our implementation is much faster.

Test (sec/step)	Mesh 1	Mesh 2	Mesh 3
(Liang et al. 2019) Forward	0.643	2.793	3.861
(Liang et al. 2019) Backward	1.400	7.379	26.212
Ours Forward	0.038	0.073	0.178
Ours Backward	0.062	0.228	0.660

4.5.4 Performance

Our implementation of the differentiable fabric model is highly inspired by (Liang et al. 2019). But ours is more GPU-friendly, and there runs fast. We compare the forward simulation and the backward gradient computation, which are two major time-consuming operations for differentiable physics models, between our implementation and (Liang et al. 2019). We use three meshes with different resolutions, consisting of 279, 1205, and 2699 vertices respectively. Table 4.3 shows the significant performance gain by our vectorization and GPU parallel computing. The test is conducted on a PC with an Intel Xeon E5-1650 v4 3.60GHz CPU and an NVIDIA TITAN Xp GPU.

4.5.5 Stretching in Cusick Drapes

Although the dominant deformation in Cusick drape is bending, the stretching also significantly affects the drape shapes (Petrak et al. 2021). To demonstrate the influence of stretching, we show simulation results of three fabric samples with different stretching stiffness: $\mathbf{C}1$, $\mathbf{C}2$, and $\mathbf{C}3$. Fig. 4.14 shows that fabric drape shapes are obviously different even if the only difference is their stretching stiffness $\mathbf{C}$. Therefore, $\mathbf{C}$ can be reflected in Cusick drape shapes and should be included in the trained parameters.

4.6 Discussion and Conclusion

We have proposed a new method for estimating detailed fabrics physical properties. To our best knowledge, this is the first Bayesian differentiable fabric model which can work seamlessly with standard Cusick drape data. Our model has been proven to be highly accurate and generalizable. Compared with black-box deep learning methods, our limitation is that it requires prior knowledge of the underlying physics and cannot simply *plug and play* on data. However, we argue that this is a reasonable trade-off when the model needs to be explainable for applications like parameter estimation. In future, we will capture more diversified data via Cusick drape testing and establish a larger benchmark dataset. Also, we will explore fabric dynamic drape and garment drape. Apart from the parameter randomness across geometries, we will study other types of uncertainties in fabric mechanical behaviors.

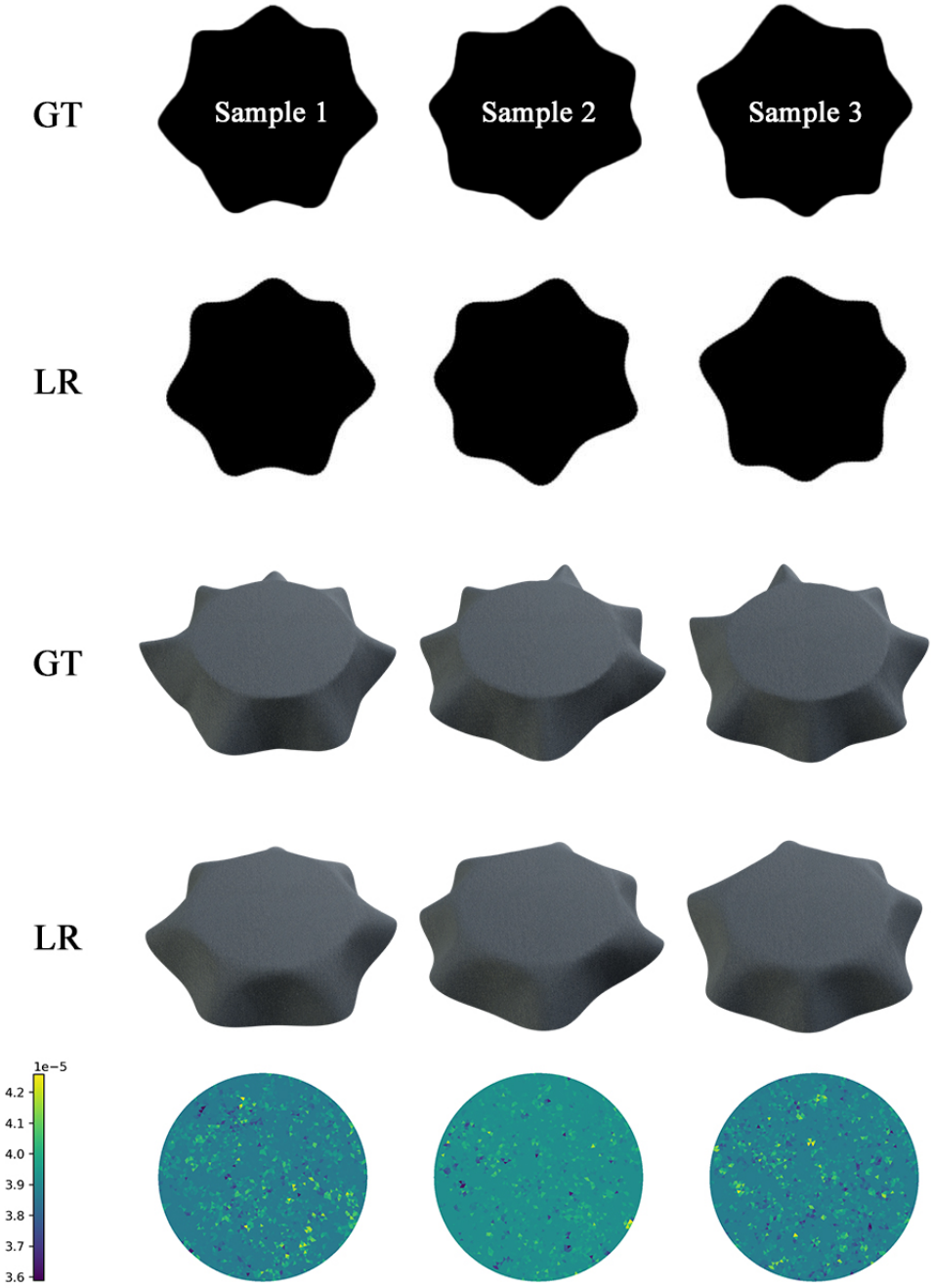


Figure 4.12: Learning from drape shapes of the 3 samples cut from fabric Cotton Blue. The GT rows show the within-sample variations in drape shapes. The LR rows show the HETER model accurately learns these different drape shapes. The heat maps (bottom) show distributions the per-element bending stiffness $\mathbf{B}$ over the mesh.

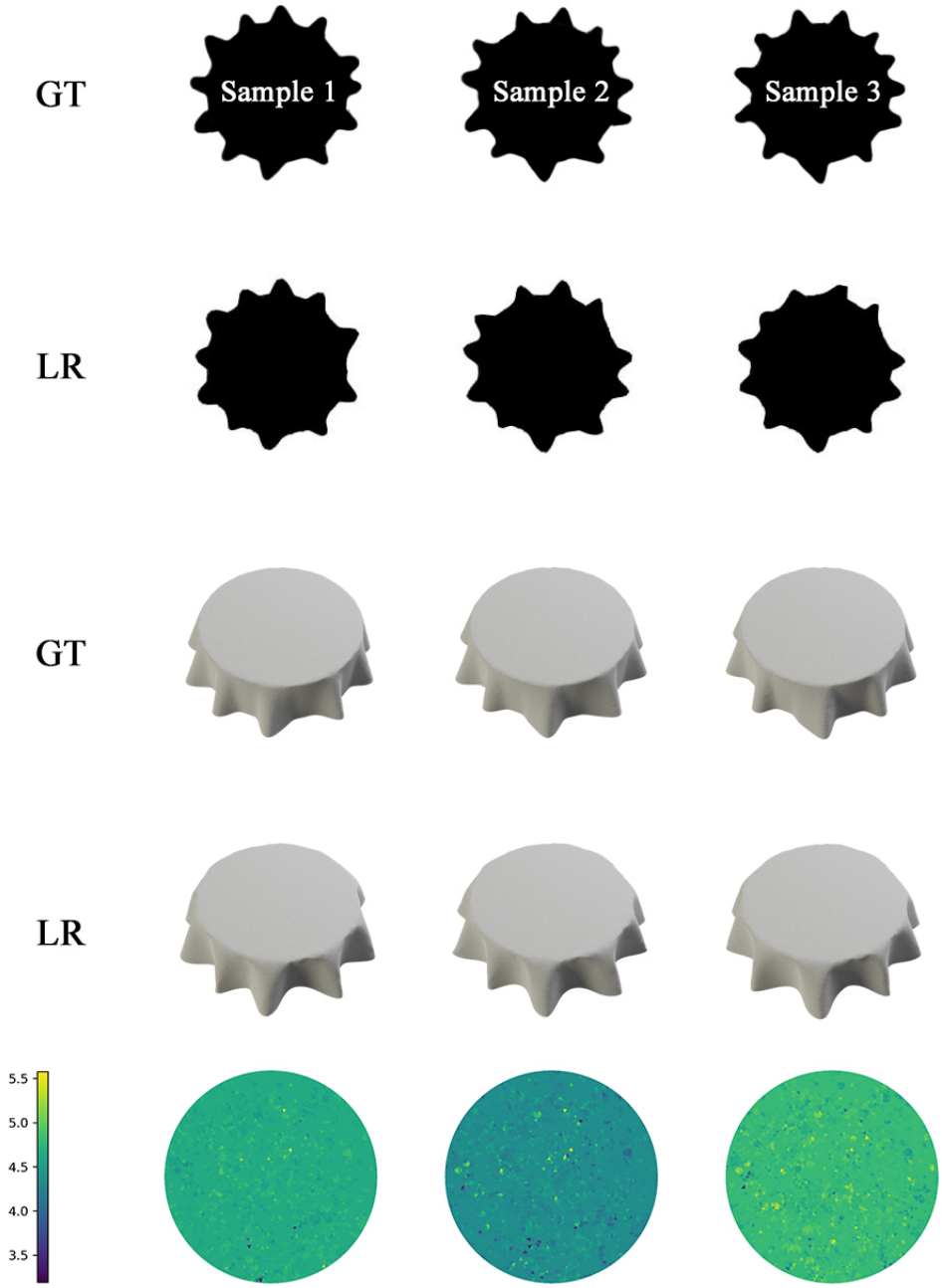


Figure 4.13: Learning from drape shapes of the 3 samples cut from fabric Viscose White. The GT rows show the within-sample varied draped shapes. The LR rows show the HETER model accurately learns these different drape shapes. The heat maps (bottom) show distributions the per-element stretching stiffness $\mathbf{C}$ over the mesh.

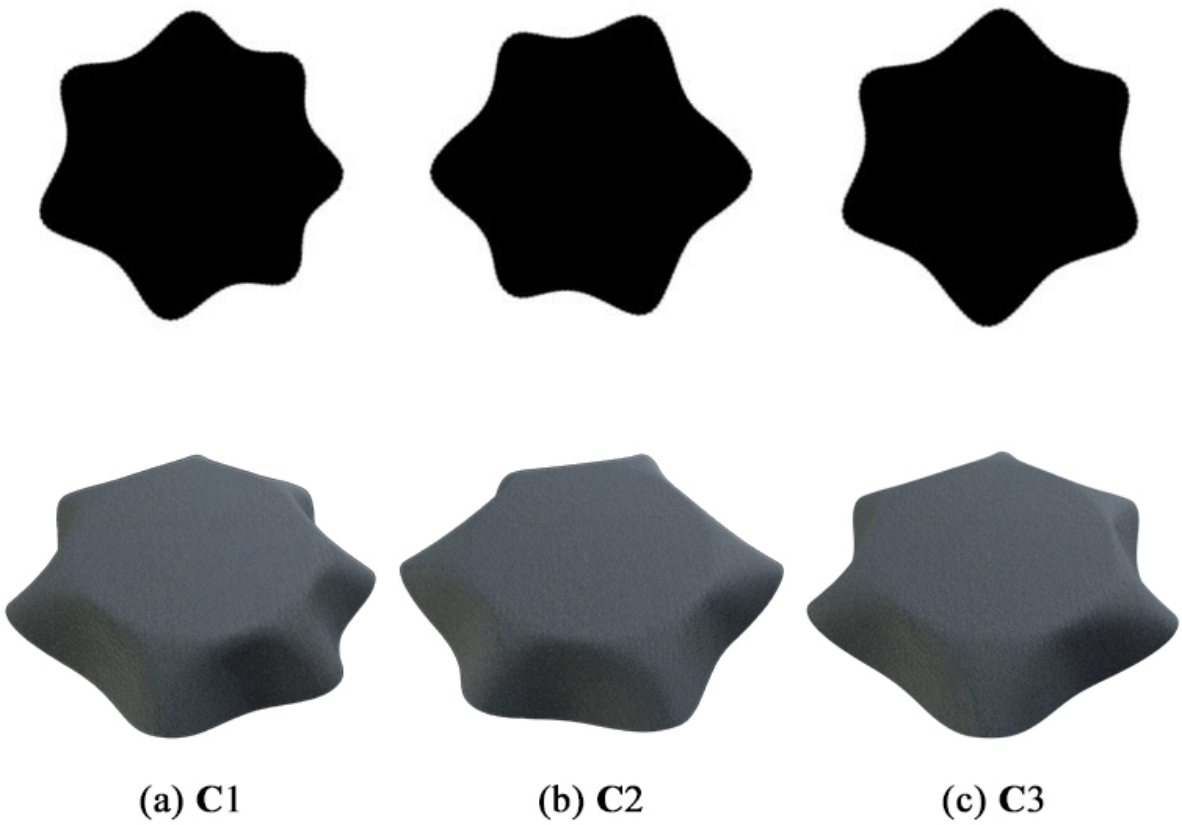


Figure 4.14: The drape shapes simulated by our differentiable physical fabric simulator with different stretching stiffness **C**1, **C**2, and **C**3. The other fabric physical parameters are the same. The result demonstrates that stretching stiffness plays a key role and affect the drape shape image.

Chapter 5

Time-dependent Persistent Wrinkles for Cloth Simulation

Apart from cloth yarn-level mechanical behaviors and material heterogeneity, cloth fine-grained physics also affects cloth persistent wrinkles which are essentially a combinational effect of the frictional behaviors between contacted yarns and fibers, and the yarns' and fibers' plastic deformations. Persistent wrinkles are often observed on garments when they are crumpled for some time, e.g., the wrinkles around the knees after sitting for a while. Such wrinkles can sometimes be easily recovered while being persistent at other times. Since they are vital to the visual realism of cloth animation, we aim to simulate realistic persistent wrinkles. To this end, we present a physically-based fine-grained wrinkle model. Different from existing methods, we recognize the importance of the interplay between internal friction and plasticity during wrinkle formation. Furthermore, we take into full consideration the *time-dependence* of persistent wrinkles. Our model is capable of not only simulating realistic wrinkle patterns as a result of deformation but also their time-dependent changes according to how long the deformation is maintained. Through extensive experiments, we show that our model is effective in simulating realistic spatial and temporal varying wrinkles, versatile in simulating different materials, and capable of generating more fine-grained wrinkles than the state-of-the-art.

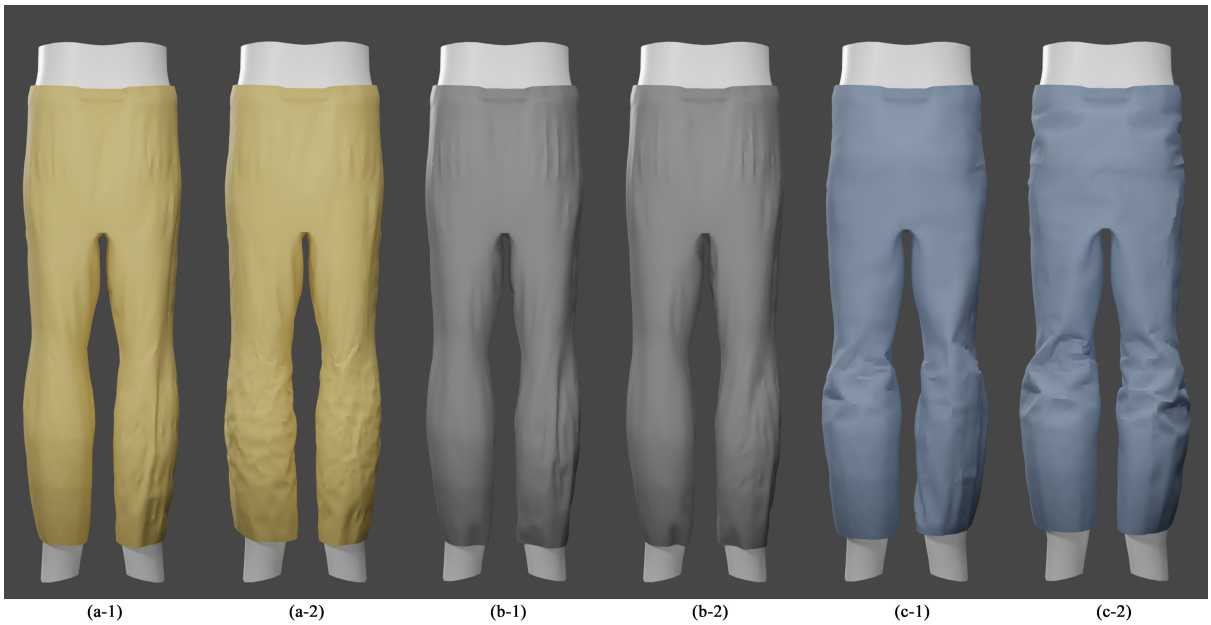


Figure 5.1: Wrinkles with time-dependency. After sitting for different duration (short: x-1; long: x-2), the wrinkles appear on the trousers made from three types of fabrics: Cotton (a-x), Polyester (b-x), and Tannin(c-x). Cotton is soft and thin. Sitting causes small wrinkles (a-1) which are more obvious after sitting for a long time (a-2). Polyester has little plasticity and rarely generates hard wrinkles. Therefore, the wrinkles on the grey polyester trousers are less noticeable (b-1 & b-2). Tannin is stiff and thick. Sitting causes big folds (c-1) which becomes more obvious after sitting for a long time (c-2).

5.1 Introduction

High-fidelity cloth simulation has been actively studied in many fields (Volino and Magnenat-Thalmann 2000). Within computer graphics, a core research topic is to generate physically valid and visually realistic cloth geometries and motions, which are required in animation, fashion design, games, etc. (Stuyck 2022). In this work, we investigate a key aspect of cloth simulation: the formation and time evolvement of wrinkles. Wrinkles at times can be seen as secondary geometries to the overall cloth dynamics, but significantly affect the visual realism (Bridson et al. 2003). Despite decades of research in cloth simulation in graphics, their underlying physics are still under-explored.

Existing methods for wrinkle simulation can be broadly divided into three categories: rule-based, data-driven and physics-based. Early attempts focus on designing rules that can map deformations into wrinkle geometries (Cutler et al. 2005). However, to achieve visual realism, it needs a lengthy trial-and-error process, heavily relying on the user experience and manual labor. Later, data-driven methods are proposed (Wang et al. 2010a) to generate wrinkles from data. While this alleviates the manual labor, it is hard to synthesize physically-valid wrinkles

outside the distribution of the data, which is particularly problematic if further simulation is needed. As an alternative solution, physics-based methods investigate the underlying physics of wrinkle formation (Rohmer et al. 2010), which provides a promising avenue for capturing the complex underlying dynamics of wrinkles.

Existing physics-based methods tend to attribute wrinkles to either plasticity or internal friction. When treating cloth as a continuum elastoplastic membrane, wrinkles can be seen as a consequence of material plasticity (Narain et al. 2013) where large deformations cause permanent material changes hence the geometric change. But this simplification neglects the impact of the internal friction between the micro-level yarns/fibers, which prevents cloths from unraveling and also leads to wrinkles (Skelton 1968). The relative sliding between yarns and the internal friction have been proven crucial in the overall dynamics (Cirio et al. 2014; Ngo and Boivin 2004). Further as argued in (Miguel et al. 2013), relative sliding is easy to happen than yarns/fiber plastic deformation, which is demonstrated to also contribute to wrinkles.

While being in favor of physics-based methods for wrinkle simulation, we argue that existing methods both simplify its formation process and completely overlook its *time-dependence*. In the formation process, the interplay between the internal friction and the plasticity plays a key role in wrinkles (Prevorsek et al. 1975). The former prevents the cloth from recovering after deformation, while the latter causes permanent material property changes (especially in large deformations). Intuitively, wrinkles caused by the internal friction are softer than the ones caused by plasticity, but they are not independent factors in wrinkle formation. Instead, the internal friction and the plasticity jointly affect the wrinkle formation, with their individual dominance dynamically changing as the deformation magnitude changes. This dictates that both factors need to be considered, which previous research has largely neglected. The Dahl's model (Ngo and Boivin 2004; Miguel et al. 2013) cannot simulate wrinkles due to the lack of stick friction and does not consider plasticity, while plasticity-based wrinkles (Narain et al. 2013) do not consider the internal friction.

Furthermore, wrinkles are time-dependent, i.e., the longer the deformation is kept, the more obvious the wrinkles tend to be (Levison et al. 1962). An intuitive real-world example is a crumbled shirt being pressed for a long time tends to form firm wrinkles with sharp edges. This suggests that there is a process that is similar to solidification when wrinkles are formed, which needs to be modeled in the plasticity and the internal friction especially for persistent wrinkles.

This time-dependence has not been studied in existing work, to our best knowledge.

To fill the research gap, we propose a new cloth simulator that is capable of generating high-fidelity wrinkles. It captures the interplay between the internal friction and the plasticity during wrinkle formation, and also explicitly models its time-dependence. While the friction model is designed to simulate wrinkles in small deformation caused by the inter-yarn sliding, the plasticity model accounts for the wrinkles caused by large deformation. Together, intensifying deformation causes wrinkles to change from friction-dominance, to mixed friction and plasticity, then finally to plasticity-dominance. Furthermore, both models are designed to be time-dependent, so that longer deformation duration causes sharper, more persistent and hard-to-recover wrinkles. Through thorough experiments, we first demonstrate that our simulator can simulate different wrinkles generated in small/large deformations by combining the friction and the plastic model. Next, we also simulate and compare the wrinkles generated with different deformation duration. The results show our simulator can generate time-dependent wrinkles, which is visually realistic as the wrinkles evolve in time. Further, we show our simulator can simulate a wide range of cloth types, from materials that are prone to hard and firm wrinkles in space and time, to materials that resist them. Finally, we demonstrate that our simulator is ready for applications by simulating wrinkles on garments caused by body motions. Through comparisons with previous work, we demonstrate that our simulator can generate more visually plausible wrinkles, with the key time-dependence that is commonly observed in daily life but largely missed by current research.

Formally, our contributions can be summarized as below:

- We propose the first cloth simulator that can generate complex persistent wrinkles with time-dependence, to our best knowledge.
- We propose a new physics model that considers both the internal friction and plasticity for simulating cloth persistent wrinkles.
- We propose a new time-dependent friction model for cloth simulation.
- We propose a new time-dependent plasticity model for cloth simulation.

5.2 Related Work

Wrinkle Simulation Wrinkles are a natural result of cloth deformation and therefore have been implicitly considered in the material modeling in the aforementioned research. However, since it greatly affects visual realism but is hard to simulate (e.g., due to mesh resolution, material properties, etc.), dedicated effort has been made towards generating realistic wrinkles. Wrinkles can be separated into *dynamic wrinkles* and *static/persistent wrinkles* (Larboulette and Cani 2004). The former refers to the fine geometrical details and the folds dynamically appear with the cloth's motions. Conversely, the latter refers to the permanent deformations with which the cloth can no longer return to its original shape after releasing external forces.

The *dynamic wrinkle* simulation method can be categorized into: rule-based method, data-driven method, and physics-based method. The rule-based method allows users to determine when and where wrinkles should appear on a cloth according to its deformation. Hadap et al. (1999) propose a wrinkling algorithm for merging multiple user-defined wrinkle patterns and then attaching the merged pattern to the cloth mesh. (Cutler et al. 2005)'s method is additionally able to interpolate and animate the varying garment wrinkles with limited artist-defined wrinkle patterns on the given reference poses. (Müller and Chentanez 2010) adopts a coarse-fine mesh paradigm where the coarse mesh is used to simulate cloth dynamics and the fine mesh encodes the wrinkles. The wrinkles on the fine mesh are generated by the predefined rule with considering the coarse geometry. Although it is flexible and simple, it is extremely unintuitive and labor-intensive to find appropriate rules for realistic wrinkles as it is not directly based on physics but geometry, which then requires a large number of iterations of trial-and-error. Alternatively, wrinkles can be pre-computed by simulation or modelers then a model can learn to generate similar wrinkles by learning a mapping between the underlying deformation to wrinkles (Wang et al. 2010a). Instead of adding wrinkles by altering geometry, Lähner et al. (2018) train a Generative Adversarial Network(GAN) by capturing real data to generate normal maps for visually adding wrinkles in the rendering pipeline.This method can be seen as a data-driven mesh refinement of low-resolution simulation. Although the data-driven method avoids defining the rules manually, it usually performs poorly when simulating wrinkles outside of the distribution of the training data, as it is difficult to gather wrinkle data to cover all possible underlying deformation. Finally, building on the physical laws, the physics-based methods exhibit superior realism and flexibility in wrinkle simulation, where the formation of wrinkles is

a natural result of the relevant material property (e.g., bending properties) explicitly modeled in the physics model. The early work (Kunii and Gotoda 1990) leverages the singularity theory to define the cloth wrinkle formation process in cloth motions. Kang et al. (2001) use their wrinkled cubic spline curve to simulate detailed wrinkles with smoothing cloth geometry. Bridson et al. (2003) propose a post-processing method to keep the wrinkles around the areas of collision. Moreover, its bending model allows artists to design persistent wrinkles by setting a non-zero rest bending angle. Compared with other two method, the physics-based method normally increases the computation time, but with modern hardware for parallel computing, the extra computation load can be largely mitigated (Wang 2021).

By contrast, the works about the formation of cloth *static/persistent wrinkles* are sparse. Narain et al. (2013) and Guo et al. (2018) use plastic deformation to account for persistent wrinkles and adopts the hardening plastic model (Grinspun 2008) to simulate them. Kim et al. (2011) simulate permanent wrinkles by changing cloth rest shape and material stiffness parameters with deformations. However, they cannot simulate time-dependent wrinkles (observed in real cloths (Levison et al. 1962)) and ignore another important reason for cloth wrinkles: internal friction.

Cloth Internal Friction Internal friction is an important cloth mechanical property which causes many uniquely physical behaviors in clothes. For example, it is widely used to account for the energy loss (hysteresis) observed in cloth load-deformation curves (Lahey 2002). On the other hand, it is an important reason for cloth persistent wrinkles (Chapman and Hearle 1972; Brenner and Chen 1964; Prevorsek et al. 1975; Chapman 1975). In graphics research, Ngo and Boivin (2004) use the Dahl's friction model to fit the load-deformation curves measured in the real cloths. Miguel et al. (2013) re-parameterize Dahl's friction model to make it more feasible for cloth simulation and shows that internal friction can also simulate wrinkles. Wong et al. (2013) introduce a mathematical model to simulate cloth internal friction and fit cloth bending hysteresis behaviors. Likewise, their friction model can also simulate persistent wrinkles. In the yarn-level cloth simulators (Cirio et al. 2014), the internal friction force is modeled as the sliding friction force between contacted yarns at the cross-overs which prevents cloths from unraveling and the shearing friction force which can simulate shearing wrinkles. However, none of these internal friction models are time-dependent. In addition, as discovered by Prevorsek et al. (1975), cloth persistent wrinkles are collectively caused by: (1) frictional fabric bending; (2)

Table 5.1: Notations

Symbol	Mean
$\mathcal{S}_t$	State at time step t
v	Number of mesh vertices
$\mathbf{x}_t$	Nodal position at time step t
$\dot{\mathbf{x}}_t$	Nodal velocity at time step t
$\mathbf{M}$	Mass matrix
$\mathbf{F}_t$	Global net force vector at time step t
h	Time step size
W_s	Stretching Energy (scalar)
$\mathbf{K}_s$	Stretching stiffness (matrix)
k_b	Bending stiffness (scalar)
$\boldsymbol{\varepsilon_s}$	Stretching strain (vector)
$\boldsymbol{\sigma_s}$	Stretching stress (vector)
$\mathbf{f}_s$	Local stretching force (vector R^9)
W_b	Bending Energy (scalar)
ε_b	Bending strain (scalar)
σ_b	Bending stress (scalar)
θ	Bending Dihedral angle
$\bar{\theta}$	Rest bending Dihedral angle
$\mathbf{f}_b$	Local bending force (vector R^{12})
A_s	The rest area of a mesh face
A_b	The rest area associated with a bending edge
k_h	Handle stiffness (scalar)
$\mathbf{f}_h$	Local handle force (vector R^3)

friction yarn bending, and (3) permanent bending of filaments (can be modeled by plastic deformations). However, in graphics, simulating cloth's persistent wrinkles by combining internal friction and plasticity has been rarely explored so far.

5.3 Basic Formulation

Notations are given in Table 5.1. The cloth is discretized into a triangle mesh, whose state is represented by the positions and velocities of mesh vertices. Specially, given a cloth mesh with v vertices, we denote its state at time t by $\mathcal{S}_t = \{\mathbf{x}_t, \dot{\mathbf{x}}_t\}$, where $\mathbf{x} \in \mathrm{R}^{3v}$ and $\dot{\mathbf{x}} \in \mathrm{R}^{3v}$ denote the nodal position and velocity vector respectively. Given the initial state $\mathcal{S}_0$, the cloth motion is governed by Newton's second law, $\mathbf{F} = \mathbf{M}\ddot{\mathbf{x}}$, where $\mathbf{M} \in \mathbf{R}^{3v\times 3v}$ is the lumped mass matrix (Logan 2022) and $\mathbf{F} \in \mathrm{R}^{3v}$ is the net force vector: the combined force of the internal and the external forces at vertices. To solve $\mathbf{F} = \mathbf{M}\ddot{\mathbf{x}}$, we employ an implicit Euler formulation (Baraff and Witkin 1998) for stability under large time steps and the governing

equation:

$$\left(\mathbf{M} - h^2\frac{\partial \mathbf{F}}{\partial \mathbf{x}} - h\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{x}}}\right)\Delta\dot{\mathbf{x}}_t = h\left(\mathbf{F}_{t-1} + h\frac{\partial \mathbf{F}}{\partial \mathbf{x}}\dot{\mathbf{x}}_{t-1}\right) \tag{5.1}$$

which can be solved by an iterative solver, e.g., Conjugate Gradient (Shewchuk 1994).

In linear elasticity, the internal force is determined by a linear relationship between strain, ε, (average deformation) and stress, σ, (average response force): $\sigma = k\varepsilon$ where k is the material stiffness. When applied to cloth simulation, a cloth, modelled as a thin shell, deformations mainly consist of in-plane stretching and out-of-plane bending, generating the stretching and the bending force respectively, where these internal forces tend to keep the cloth in its rest state. Under a triangular mesh discretization, the stretching and the bending strain/stress are calculated per triangle and every two adjacent triangles (as illustrated in Fig. 4.5 Left and Middle). For stretching, we adopt the Green-Lagrange strain tensor to define the triangle deformation and, in Voigt Notation, it can be denoted by the vector $\boldsymbol{\varepsilon}_s = (\varepsilon_{uu}, \varepsilon_{vv}, \varepsilon_{uv})$. Cloths, e.g., woven fabrics, usually exhibit distinctive stretching mechanical properties in the warp and the weft direction, and they are usually modeled as orthotropic materials (Boisse et al. 2001). Therefore, ε_{uu}, ε_{vv}, and ε_{uv} represent the tensile strains along the warp, the weft, and the diagonal direction (shearing strain), respectively. The stretching constitutive equation is:

$$\boldsymbol{\sigma}_s = \begin{bmatrix}\sigma_{uu}\\ \sigma_{vv}\\ \sigma_{uv}\end{bmatrix} = \begin{bmatrix}k_{11} & k_{12} & 0\\ k_{12} & k_{22} & 0\\ 0 & 0 & k_{33}\end{bmatrix}\begin{bmatrix}\varepsilon_{uu}\\ \varepsilon_{vv}\\ \varepsilon_{uv}\end{bmatrix} = \mathbf{K}_s\boldsymbol{\varepsilon}_s \tag{5.2}$$

where $\mathbf{K}_s$ is the stretching stiffness matrix in which k_{11}, k_{22}, k_{33}, and k_{12} are the warp/weft/shear stretching stiffness and Poisson's ratio (Wang et al. 2011). The nodal stretching force is the partial derivative of the stretching energy w.r.t. the vertex position:

$$W_s = A_s\int \boldsymbol{\sigma}_s \mathrm{d}\boldsymbol{\varepsilon}_s = \frac{1}{2}A_s\boldsymbol{\sigma}_s\boldsymbol{\varepsilon}_s \tag{5.3}$$

$$\mathbf{f}_s = -\frac{\partial W_s}{\partial \mathbf{x}_s} = \frac{1}{2}A_s\boldsymbol{\sigma}_s\frac{\partial \boldsymbol{\varepsilon}_s}{\partial \mathbf{x}_s} \tag{5.4}$$

where W_s is the stretching energy, A_s is the rest area of a triangle and $\mathbf{x}_s \in \mathcal{R}^9$ is the position vector of the triangle's vertices. Please refer to work proposed by Volino et al. (2009) for $\frac{\partial \boldsymbol{\varepsilon}_s}{\partial \mathbf{x}_s}$ and the force Jacobians.

For bending, we adopt the method in the work proposed by Grinspun et al. (2003) which uses the mean-curvature to define the bending strain: $\varepsilon_b = 3\frac{\theta-\bar{\theta}}{e}$ where θ is the Dihedral angle between two adjacent triangles and $\bar{\theta}$ is the rest Dihedral angle. e is the average height of the two triangles. Therefore, the elastic bending stress is $\sigma_b = K_b\varepsilon_b$ where K_b is the bending stiffness. Similarly, the nodal bending force is the partial derivative of the bending energy w.r.t. the vertex position:

$$W_b = A_b \int \sigma_b \mathrm{d}\varepsilon_b = \frac{1}{2} A_b \sigma_b \varepsilon_b \tag{5.5}$$

$$\mathbf{f}_b = -\frac{\partial W_b}{\partial \mathbf{x}_b} = -\frac{1}{2} A_b \sigma_b \frac{\partial \varepsilon_b}{\partial \mathbf{x}_b} = -\frac{3}{2}\frac{A_b \sigma_b}{e}\frac{\partial \theta}{\partial \mathbf{x}_b} \tag{5.6}$$

where W_b is the bending energy, $A_b = \frac{1}{3} le$ where l is the rest length of the common edge shared by the two triangles. $\mathbf{x}_b \in \mathcal{R}^{12}$ is the vertex position vector of the two triangles. Please refer to (Tamstorf and Grinspun 2013) for the derivatives and the Jacobians of θ.

Other than the internal forces, we also consider external forces which include gravity, collision, external friction, and handle force. Gravity is a body force and is applied to all lump masses. In addition, we adopt (Bridson et al. 2002) to handle collision detection and collision response. In every simulation step, it uses boundary volume hierarchies (BVH) and continues collision detection (CCD) to detect edge-edge and vertex-face collisions which are then merged into rigid impact zones. The collision response combines repulsion force method and geometric method to avoid the wrong intersections. The repulsion force method adds repulsive force to push the edge-edge and vertex-face in collisions apart when the distance between two edges or a vertex and a face is smaller than a given threshold. The repulsion force method may fail when, for example, the cloth geometry changes extremely fast. In this case, the geometric method will correct the penetrations by directly altering vertices positions. Moreover, if there is relative sliding between elements, external friction forces are introduced to prohibit relative motions. We refer the reader to the original paper for more details. Finally, the handle force is intended for controlling the cloth and derives from a penalty-based method to pin vertices at specified locations:

$$\mathbf{f}_h = k_h(\mathbf{x} - \mathbf{x}_h) \tag{5.7}$$

where k_h is the handle stiffness and $\mathbf{x}_h$ specifies the anchor positions in space.

5.3.1 Wrinkles

Wrinkle formation is mainly dictated by the bending deformation (Wong et al. 2013; Wong 2014; Fan and Hunter 2009). Given a flat undeformed cloth, all the Dihedral angles are $\bar{\theta} = \pi$ which are the rest angles. As shown in Eq. (5.6), the bending force always tends to keep a Dihedral angle at its rest angle $\bar{\theta}$. However, when wrinkles are formed due to forces that counter-balance this bending force, there are two sources of such impact: (1) another force preventing the cloth from recovering to the rest angle $\bar{\theta}$; or/and (2) the $\bar{\theta}$ being changed after deformation so that the cloth cannot return to the initial rest angle without external forces. These two sources correspond the internal friction and the plasticity respectively. The wrinkles that are mainly caused by the internal friction are usually soft and easy to recover (e.g., after stretching), while the ones that are mainly caused by the plasticity are often firm, persistent and even unrecoverable. For simplicity, we refer to the former as *friction wrinkles* and the latter as *plastic wrinkles*. Note the names here are for simplicity rather than suggesting wrinkles are dictated by a single factor.

5.3.2 Internal Friction Model

Shifting Anchor Friction Friction forces can prevent relative sliding between two contacting objects, which in cloth is the relative sliding between contacting yarns/fibers. The internal friction can cause wrinkles because, at the microscopic level, a bending causes both the yarns/-fibers deformation and the relative sliding between yarns/fibers (Lin et al. 2012). So when the internal bending force tends to recover the cloth to its rest state, it causes the opposite relative sliding and the internal friction prevents this relative sliding. We model this friction by the variation of the bending strain, ε_b:

$$\sigma_{friction} = K_{friction} \Delta\varepsilon_b \tag{5.8}$$

where $K_{friction}$ is the friction coefficient. $\Delta\varepsilon_b$ is the bending strain variation that measures the difference between the current bending strain and an anchor bending strain $\bar{\varepsilon}_b$.

$$\Delta\varepsilon_b = \varepsilon_b - \bar{\varepsilon}_b \tag{5.9}$$

While $\sigma_{friction}$ is proportional to $\Delta\varepsilon_b$, it cannot grow unbounded as $\bar{\varepsilon}_b$ is not fixed. This is because when the bending deformation becomes larger, the internal friction can merely counterbalance the bending force up to a threshold before sliding happens, as observed in stick-slip friction (Al-Bender et al. 2004). To model this, we update $\bar{\varepsilon}_b$ when $\Delta\varepsilon_b$ is greater than a predefined stick-slip transition threshold, ε_{thres}:

$$\bar{\varepsilon}_b = \begin{cases} \bar{\varepsilon}_b & \text{, if } |\varepsilon_b - \bar{\varepsilon}_b| < \varepsilon_{thres} \\ \bar{\varepsilon}_b + (\varepsilon_b - \varepsilon_{thres}) & \text{, otherwise} \end{cases} \tag{5.10}$$

When stick friction happens ($\bar{\varepsilon}_b$ not updated), the friction keeps the cloth around the anchor bending strain; otherwise the slide friction occurs ($\bar{\varepsilon}_b$ updated), the friction prevents relative motions and the anchor bending strain will move toward the current strain until $|\varepsilon_b - \bar{\varepsilon}_b| < \varepsilon_{thres}$. Then, it reverts back to the stick friction and keeps the cloth around the new anchor bending strain. Overall, this allows the internal friction to generate small and large wrinkles, while allowing these wrinkles to recover under external forces, e.g., flatten the cloth by stretching.

Time dependence One common but under-explored phenomenon in cloth internal friction is the dwell effect: the longer the stick friction state is maintained, the more difficult it is for slide friction to happen (Al-Bender et al. 2004). To this end, we propose a new friction component to make wrinkles time-dependent. As the duration of a friction wrinkle increases, the wrinkle becomes more difficult to recover.

As ε_{thres} delimits the stick-slip transition, our friction model varies this threshold with time to simulate this time-dependent behavior:

$$t_{stick} = \begin{cases} t_{stick} + h, & \text{if } |\varepsilon_b - \bar{\varepsilon}_b| < \varepsilon_{thres} \\ 0, & \text{otherwise} \end{cases}$$
$$\text{where } \varepsilon_{thres} = \varepsilon_{inf} - (\varepsilon_{inf} - \varepsilon_0) \times e^{-\frac{t_{stick}}{\tau_f}} \tag{5.11}$$

where we introduce a variable t_{stick} for the duration of the stick state such that ε_{thres} can vary within the inteval $[\varepsilon_0, \varepsilon_{inf}]$. Concretely, in every step, t_{stick} increases by the time step size h if slip friction does not occur ($|\varepsilon_b - \bar{\varepsilon}_b| < \varepsilon_{thres}$). In turn, this will increase ε_{thres} toward ε_{inf}, and consequently the slip friction becomes more difficult to happen. On the other hand, if slip

friction occurs, t_{stick} is reset to zero and ε_{thres} is then reduced to ε_0. As a result, the slip friction resets the dwell effect. In addition, the parameter τ_f allows users to control the increase rate of ε_{thres} with t_{stick}.

Stribeck and Varying Break-away Forces In addition to the time-dependence, the relative sliding can frequently happen which corresponds to frequent transitions between the stick state and the slip state, which is also crucial for wrinkle formation. Therefore, we also introduce a stick-slip transition model for $K_{friction}$ in Eq. (5.8).

In the simplified Coulomb's models, $K_{friction}$ is assumed to be a constant or to vary slightly between stick and slip friction (Pennestrì et al. 2016). However, it has been observed this is a massive simplification and there is a non-linear relation between the stick and slip friction known as the Stribeck effect and varying break-away force (De Wit et al. 1995; Swevers et al. 2000; Gong et al. 2022).

Stribeck effect refers to the phenomena that the friction stress decreases as the friction transits to the slip state from the stick state (Armstrong-Helouvry 2012). Moreover, increasing the slip velocity further decreases the slip friction stress. To simulate the Stribeck effect, we vary $K_{friction}$ with an anchor strain variation rate $\dot{\varepsilon}$.

$$K_{friction} = K_c + (K_s - K_c)e^{-|\dot{\bar{\varepsilon}}_b|/\varepsilon_s} \tag{5.12}$$

where K_s and K_c denote the stick and the slip friction stiffness respectively. ε_s denotes the threshold from which the $K_{friction}$ starts to change from K_s to K_c as $\dot{\bar{\varepsilon}}_b$ increases. As the stick friction stress is usually greater than the slip friction stress, K_c is smaller than K_s. In this way, $K_{friction} = K_s$ in the stick state where $\dot{\bar{\varepsilon}}_b = 0$. By contrast, in the slip state where $\dot{\bar{\varepsilon}}_b \neq 0$, $K_{friction}$ is smaller than K_s. As $\dot{\bar{\varepsilon}}_b$ increases, $K_{friction}$ will further deceases toward K_c.

Break-away force refers to the needed force for switching from stick friction to slip friction. The varying break-away force is the phenomenon that the higher the increase rate of the applied force is, the smaller the break-away force tends to be (Richardson and Nolle 1976). In cloth simulation, a high force increase rate means a large force increase between two consecutive steps, i.e., $|\mathbf{F}_t| - |\mathbf{F}_{t-1}|$. This leads to a large $\Delta\dot{\mathbf{x}}$ (Eq. (4.6)) and a large strain variation rate. Therefore, our friction model realizes the varying break-away effect by varying the stick friction

stiffness K_s with the strain variation rate $\dot{\varepsilon}$:

$$K_s = C_0 + C_1 e^{-C_2|\dot{\varepsilon}_b|} \tag{5.13}$$

where C_0 is the minimal stick friction stiffness. The parameters C_1 and C_2 control the additional stick stiffness and how fast K_s decreases as $\dot{\varepsilon}_b$ increases. Therefore, the higher $\dot{\varepsilon}_b$ is, the smaller K_s is, which leads to a smaller break-away force $K_s\varepsilon_{thres}$.

Deriving the Friction Force Having defined all the components needed, we derive the nodal force due to the internal friction. Similarly to Eq. (5.6), we first define a friction energy on the bending edge:

$$\begin{aligned} W &= A_b \int_{\bar{\varepsilon}}^{\varepsilon} \sigma_{friction} d\varepsilon_b = A_b \int_{\bar{\varepsilon}}^{\varepsilon} K_{friction} \Delta\varepsilon_b d\varepsilon_b \\ &= A_b (\frac{1}{2} K_{friction} \varepsilon_b^2 - K_{friction} \bar{\varepsilon}_b \varepsilon_b)|_{\bar{\varepsilon}}^{\varepsilon} \end{aligned} \tag{5.14}$$

Therefore, the corresponding friction forces on the four vertices around the bending edge are:

$$\mathbf{F} = -\frac{\partial W}{\partial \mathbf{x}} = -\frac{\partial W}{\partial \varepsilon_b}\frac{\partial \varepsilon_b}{\partial \mathbf{x}} = -A_b K_{friction}(\varepsilon_b - \bar{\varepsilon}_b)\frac{\partial \varepsilon_b}{\partial \mathbf{x}} \tag{5.15}$$

To solve Eq. (5.1), we need the force Jacobian w.r.t the vertex positions:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{x}} = -A_b K_{friction}\left(\frac{\partial^2 \varepsilon_b}{\partial \mathbf{x}^2} + \frac{\partial \varepsilon_b}{\partial \mathbf{x}}\frac{\partial \varepsilon_b}{\partial \mathbf{x}}^\top\right) \tag{5.16}$$

where the bending strain's second-order derivative, $\frac{\partial^2 \varepsilon_b}{\partial \mathbf{x}^2} = 0$. We also need the force Jacobian w.r.t. the velocity:

$$\begin{aligned} \frac{\partial \mathbf{F}}{\partial \dot{\mathbf{x}}} &= -A_b(\varepsilon_b - \bar{\varepsilon}_b)\frac{\partial K_{friction}}{\partial \dot{\mathbf{x}}}\frac{\partial \varepsilon_b}{\partial \mathbf{x}}^\top \\ &= -A_b(\varepsilon_b - \bar{\varepsilon}_b)\left(\frac{\partial K_S}{\partial \dot{\mathbf{x}}} e^{-|\dot{\bar{\varepsilon}}_b|/\varepsilon_s}\right)\frac{\partial \varepsilon_b}{\partial \mathbf{x}}^\top \\ &= A_b sign(\dot{\varepsilon}_b)(\varepsilon_b - \bar{\varepsilon}_b) C_1 C_2 e^{-C_2|\dot{\varepsilon}_b| - |\dot{\bar{\varepsilon}}_b|/\varepsilon_s}\frac{\partial \dot{\varepsilon}_b}{\partial \dot{\mathbf{x}}}\frac{\partial \varepsilon_b}{\partial \mathbf{x}}^\top \end{aligned} \tag{5.17}$$

Looking closely, Eq. (5.17) is often dominated by $e^{-|\dot{\bar{\varepsilon}}_b|/\varepsilon_s}$ as it is very small. This is because the difference between K_S and K_C is usually small as observed in Stribeck. Empirically, we find no visible changes when setting $\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{x}}} = 0$. Note the time-dependence, the Stribeck and varying

breaking-way forces are still largely maintained by Eq. (5.16). Therefore, we safely remove $\frac{\partial \mathbf{F}}{\partial \hat{\mathbf{x}}}$ in our experiments. The overall algorithm for the internal friction is shown in Algorithm 2.

Algorithm 2 Time-dependent Friction

procedure FRICTION($\varepsilon_n, \varepsilon_{n-1}, \bar{\varepsilon}$)
  $\dot{\varepsilon} \leftarrow \frac{\varepsilon_n - \varepsilon_{n-1}}{h}$
  $\varepsilon_{thres} = \varepsilon_{inf} - (\varepsilon_{inf} - \varepsilon_0)e^{-t_{stick}/\tau_f}$
  $\Delta\varepsilon = \varepsilon_n - \bar{\varepsilon}$
  if $|\Delta\varepsilon| > \varepsilon_{thres}$ **then** ▷ Slip Friction
    $\bar{\varepsilon} \leftarrow \bar{\varepsilon} + sign(\Delta\varepsilon) * (|\Delta\varepsilon| - \varepsilon_{thres})$ ▷ Eq.5.10
    $|\dot{\bar{\varepsilon}}_b| = (|\Delta\varepsilon| - \varepsilon_{thres})/h$
    $K_s = C_0 + C_1 e^{-C_2|\dot{\varepsilon}_b|}$ ▷ Eq.5.13
    $K_{friction} = K_c + (K_s - K_c)e^{-|\dot{\bar{\varepsilon}}_b|/\varepsilon_s}$ ▷ Eq.5.12
    Compute $\mathbf{F}$, $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$ (Eq. 5.15 and 5.16)
  else ▷ Stick Friction
    $\mathbf{F} = \mathbf{0}$, $\frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \mathbf{0}$
  end if
  return $\mathbf{F}$, $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$
end procedure

5.3.3 Plastic Model

An intuitive way of modeling plastic wrinkles is through the change of the rest Dihedral angles. When a large deformation happens and is maintained for long duration, the rest Dihedral angle changes hence the change of the cloth rest state, so that the internal bending stress does not attempt to recover the cloth to the previous rest state. To this end, we employ an elastoplastic constitutive law so the bending strain can be decoupled to an elastic part and a plastic part (O'Brien et al. 2002):

$$\varepsilon_b = \varepsilon_e + \varepsilon_p \tag{5.18}$$

and the bending stress is proportional to the elastic part

$$\sigma = K_b \varepsilon_e = K_b(\varepsilon_b - \varepsilon_p). \tag{5.19}$$

Given a flat wrinkle-free cloth with no plastic strain ($\varepsilon_p = 0$), the bending stress tends to keep it wrinkle-free. Otherwise, its rest shape is changed to $\varepsilon_b - \varepsilon_p = 0$. In this case, the bending stress tends to keep the cloth in its new rest shape which is no longer flat such that plastic wrinkles appear.

Since a plastic deformation does not occur until the deformation reaches a certain threshold, e.g., a large bending deformation exceeding certain threshold, which is usually referred to as

yield strain (we denote it by ε_Y). The simplest plastic model is what is called the perfect plastic model where all of the elastic strain ε_e that exceeds ε_Y is treated as the plastic strain (Chaves 2013):

$$\varepsilon_p = \begin{cases} \varepsilon_e - \varepsilon_Y, & \text{if } \varepsilon_e > \varepsilon_Y \\ \varepsilon_p, & \text{otherwise} \end{cases} \tag{5.20}$$

However, this simplified plastic model does not capture the time-dependence nature observed on real cloths (Marchesi et al. 2012). In reality, there is a hardening process which is dependent on time (Benusiglio et al. 2012). To this end, we let the plastic deformation gradually increase with time so that plastic wrinkles can become sharper by keeping a plastic deformation for a long period. We introduce a time-dependent hardening plastic model which can alter the yield strain, ε_Y, with time so that the plastic strain $\varepsilon_p = \varepsilon_e - \varepsilon_Y$ becomes time-dependent. Further, we also assume that this dependence on time is related to the material, i.e., some cloths quickly form plastic wrinkles while others take longer time, which is empirically observed. To our best knowledge, there is no generally accepted parameteric form to describe this relation for various materials. But we do notice there is an overall decelerated hardening along with time across different materials (Benusiglio et al. 2012). This inspires us to propose the following model. When plasticity occurs, i.e., $\varepsilon_e > \varepsilon_Y$, it changes according to:

$$K_h = K_{h0}(1 - g(1 - e^{(-t_{plastic}/\tau_p)})) \tag{5.21}$$

$$\varepsilon_{hp} \leftarrow \varepsilon_{hp} + \frac{K_b}{K_b + K_h}(|\varepsilon_e| - \varepsilon_Y) \tag{5.22}$$

$$\varepsilon_p \leftarrow \varepsilon_p + sign(\varepsilon_e)\frac{K_b}{K_b + K_h}(|\varepsilon_e| - \varepsilon_Y) \tag{5.23}$$

$$\varepsilon_Y = \varepsilon_{Y0} + \varepsilon_{hp}\frac{K_h}{K_b} \tag{5.24}$$

where K_h and K_{h0} are the hardening parameter and initial hardening parameter respectively. $g \in (0, 1)$ controls the lower bound of the hardening parameters and τ_p decides K_h variation rate. The variable $t_{plastic}$ times how long the plastic deformation has been kept. In every

simulation step, it is updated according to

$$t_{plastic} = \begin{cases} t_{plastic} + h & \text{, if } sign(\varepsilon_e) = sign(\varepsilon_p) \text{ and } \varepsilon_e > \varepsilon_Y \\ 0 & \text{, otherwise} \end{cases} \tag{5.25}$$

Therefore, $t_{plastic}$ is increased by h in every step if the plastic deformation flows to the overall deformation, or otherwise reduced to zero. ε_{hp} denotes hardening plastic strain which is used for accumulating the plastic deformation and controlling the plastic hardening effect. Also, ε_p denotes the actual plastic deformation that is used in Eq. (5.19). To simulate plastic hardening, the yield strain ε_Y is affected by the plastic deformation and it can increase from a given initial yield strain ε_{Y0}.

To further understand how hardening is simulated by Eq.5.21-5.25, we describe a scenario where a plastic deformation first appears and then develops with time. When a plastic deformation first appears, $t_{plastic}$ is increased which decreases K_h (Eq. (5.21)). This hardens the cloth as ε_{hp} is increased (Eq. (5.22)), and so does the yield strain ε_Y eventually (Eq. (5.24)). Moreover, only part of the bending strain that exceeds the yield strain is treated as plastic strain. If the current deformation is kept, $t_{plastic}$ will gradually increases and K_h reduces from K_{h0} toward $K_{h0}(1-g)$. Consequently, ε_{hp} and ε_Y further increase, and cloth is hardened. Meanwhile, the plastic strain ε_p also gradually accounts more for the over all strain/stress so that the elastic strain/stress $\varepsilon_e = \varepsilon - \varepsilon_p$ reduces. This procedure is repeated as the simulation runs until $\varepsilon_e \leq \varepsilon_Y$. Overall, the longer the deformation is maintained, the larger the plastic strain is and also the more obvious the plastic wrinkles are. The algorithm of our time-dependent hardening plastic model is shown in Algorithm 3.

After integrating the plastic model, the bending nodal force defined in Eq. (5.6) becomes

$$\mathbf{f}_b = -\frac{1}{2} A_b \sigma_b \frac{\partial \varepsilon_b}{\partial \mathbf{x}} = -\frac{1}{2} A_b K_b (\varepsilon_b - \varepsilon_p) \frac{\partial (\varepsilon_b - \varepsilon_p)}{\partial \mathbf{x}} \tag{5.26}$$

The plastic hardening is usually very slow which can take minutes or even hours to observe noticeable changes. As one simulation step which is usually between 0.01s and 0.001s for stability, the plastic strain variation is usually small so that we can approximate $\frac{\partial \varepsilon_p}{\partial \mathbf{x}} \approx \mathbf{0}$ and bending nodal force is

$$\mathbf{f}_b = -\frac{1}{2} A_b K_b (\varepsilon_b - \varepsilon_p) \frac{\partial \varepsilon_b}{\partial \mathbf{x}} \tag{5.27}$$

Algorithm 3 Time-dependent Hardening Plastic Model

procedure PLASTIC_BENDING($\varepsilon_n, \varepsilon_p, \varepsilon_Y$)
  $\varepsilon_e = \varepsilon_n - \varepsilon_p$
  if $|\varepsilon_e| > \varepsilon_Y$ **then** ▷ Plastic Deformation
    if $sign(\varepsilon_e) = sign(\varepsilon_p)$ **then**
      $t_{plastic} \leftarrow t_{plastic} + h$
    else
      $t_{plastic} \leftarrow 0$
    end if
    $K_h = K_{h0}(1 - g(1 - e^{(-t_{plastic}/\tau_p)}))$ ▷ Eq. 5.21
    $\varepsilon_{hp} \leftarrow \varepsilon_{hp} + \frac{K_b}{K_b+K_h}(|\varepsilon_e| - \varepsilon_Y)$ ▷ Eq. 5.22
    $\varepsilon_p \leftarrow \varepsilon_p + sign(\varepsilon_e)\frac{K_b}{K_b+K_h}(|\varepsilon_e| - \varepsilon_Y)$ ▷ Eq. 5.23
    $\varepsilon_Y = \varepsilon_{Y0} + \varepsilon_{hp}\frac{K_h}{K_b}$ ▷ Eq. 5.24
    $\varepsilon_e \leftarrow \varepsilon_Y$
  end if
  Compute $\mathbf{F}$, $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$ (Eq.5.6 and (Grinspun et al. 2003))
  return $\mathbf{F}$, $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$
end procedure

where we avoid computing complex derivatives and do not observe any negative impact on the simulation stability in practice.

5.4 Friction and Plasticity in Tensile

Internal friction and plastic deformation also exist in cloth in-plane tensile deformations. The internal friction prevents a stretched cloth from returning to its rest length. Overly stretching a cloth can cause plastic tensile deformations so a new rest state (longer than the original one) is established and the cloth can only recover to its new rest length.

Our friction and plastic model can also be applied to model cloth tensile internal friction and plasticity. Similarly to the elastic tensile model, the tensile internal friction is also orthotropic. Therefore, the tensile friction stress of a mesh triangle is:

$$\boldsymbol{\sigma}_{fri} = \begin{bmatrix} \sigma_{fri-uu} \\ \sigma_{fri-vv} \\ \sigma_{fri-uv} \end{bmatrix} = \begin{bmatrix} K_{11} & 0 & 0 \\ 0 & K_{22} & 0 \\ 0 & 0 & K_{33} \end{bmatrix} \begin{bmatrix} \Delta\varepsilon_{fri-uu} \\ \Delta\varepsilon_{fri-vv} \\ \Delta\varepsilon_{fri-uv} \end{bmatrix} \tag{5.28}$$

where K_{11}, K_{22}, and K_{33} are the internal friction stiffness along the warp, the weft, and the diagonal direction respectively. The tensile internal friction strains/stresses along these directions are independent and their internal friction stiffness are updated independently in the same way

as introduced in Section 5.3.2.

Similarly, the tensile plastic strains in the three directions are independent as well:

$$\begin{bmatrix} \varepsilon_{uu} \\ \varepsilon_{vv} \\ \varepsilon_{uv} \end{bmatrix} = \begin{bmatrix} \varepsilon_{e\_uu} \\ \varepsilon_{e\_vv} \\ \varepsilon_{e\_uv} \end{bmatrix} + \begin{bmatrix} \varepsilon_{p\_uu} \\ \varepsilon_{p\_vv} \\ \varepsilon_{p\_uv} \end{bmatrix} \tag{5.29}$$

and their plastic hardening effects are also updated independently following Section 5.3.3.

5.5 Implementation

Our implementation is in MSVC(10.0.19041.0) C++ with OpenMP for CPU parallel computing. We use the Eigen library (Guennebaud, Jacob, et al. 2010) for matrix calculation and solving the governing equation. Our experiments are conducted on a PC with an Intel i7-12700H 2.3GHz CPU and 16GB 4800MHz RAM.

In addition, we use the Dihedral angle θ to define the slide friction threshold ε_{thres} and the yield strain ε_Y, rather than the bending strain ε_b. This is because ε_b is affected by specific discretization, e.g., the size of triangles. $\varepsilon_b = 3\frac{\theta-\bar{\theta}}{e}$ where the average height $e = \frac{A_{s1}+A_{s2}}{4l}$ is decided by two adjacent triangles' rest areas, A_{s1} and A_{s2}, and the bending edge's rest length l. When tuning the parameters, it is counter-intuitive if the slide friction thresholds ε_{thres} and ε_Y vary according to the cloth mesh discretization. As the bending strain ε_b essentially is a scaled θ, we use θ as a surrogate to measure deformation, and define the ε_{thres} and ε_Y accordingly. Consequently, using the Dihedral angle to define the thresholds is more intuitive for users.

The parameters τ_f and τ_p allow us to define the variation rate of ε_{thres} and K_h with respect to the deformation duration. Remember ε_{thres} and K_h control the friction dwell effect and plastic hardening effect respectively which further decide how fast the friction/plastic wrinkles appear in time. To show the time-dependence, theoretically our simulator can run for a long time to simulate e.g., slow hardening processes. However, this might not be ideal in applications. Therefore, we tune the parameters given a fixed deformation duration (i.e., how long a deformation is kept) for different materials. In particular, given the same deformation duration, the smaller the τ's are, the more obvious the wrinkles tend to be. Therefore, we can flexibly determine a deformation duration that is long enough for the friction/plastic wrinkles to develop,

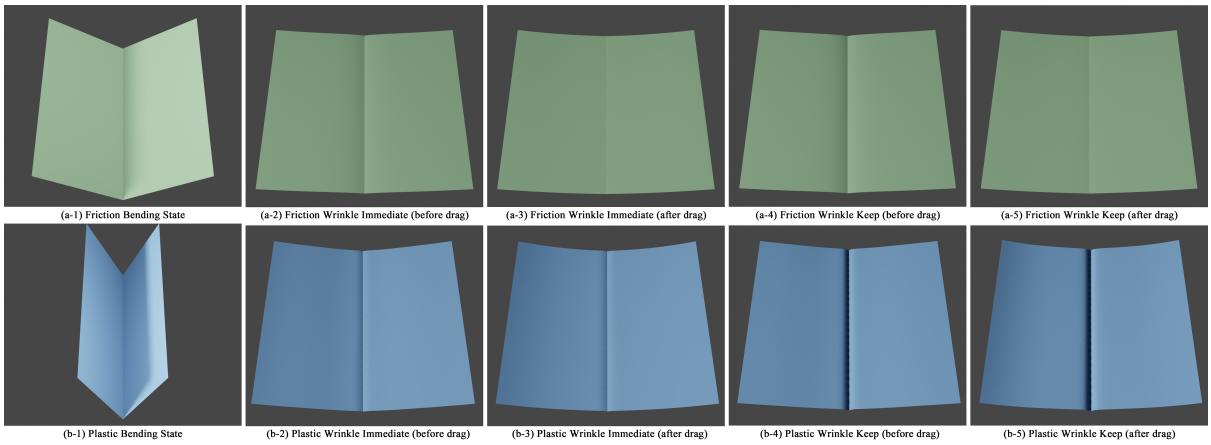


Figure 5.2: The friction (top) and the plastic (bottom) wrinkles separately simulated by the internal friction model and the plastic model. After the initial bending (a-1, b-1), we either immediately release it (a-2, b-2) or keep it for a while (a-4, b-4), after which we stretch the cloth trying to reverse the wrinkle (a-3, a-5, b-3, b-5). The wrinkles tend to be more recoverable after the immediate release (b, e) than being kept for a while (c, f). Also, plastic wrinkles are harder to recover than friction wrinkles.

as long as that duration is much greater than the τ's. In our experiments, we set the τ's to 30 seconds and the long deformation duration to 500 seconds which is sufficient to observe the wrinkle time-dependence. In addition, we can increase t_{stick} and $t_{plastic}$ by larger time steps to avoid running the simulation for too long. For instance, in our experiments, we increase t_{stick} and $t_{plastic}$ by 10 seconds in every simulation step when keeping the cloth's deformation. This enables us to only run 50 steps before t_{stick} and $t_{plastic}$ reach 500 seconds.

5.6 Experiments

5.6.1 Ablation Study

Friction/Plastic Wrinkles To verify that both the internal friction and the plasticity can cause wrinkles, we show them separately by disabling one factor and simulate cloth using the other. We design a simple one-wrinkle experiment shown in Fig. 5.2, where a small deformation (Fig. 5.2 a-1) is used to generate a pure friction wrinkle. After the deformation, we release it either immediately or after keeping the deformation for a long period. After releasing it, we slightly stretch the cloth by pulling the left and the right side in opposite directions, to attempt to flatten the cloth (testing if the wrinkle is reversible). With the immediate release (Fig. 5.2 a-2 and a-3), the wrinkle is largely reversible; while if the deformation is kept for a while, it is less so (Fig. 5.2 a-4 and a-5).

We also show the same experiment when only plasticity is considered. A large initial deformation (Fig. 5.2 b-1) is followed by an immediate release (Fig. 5.2 b-2 and b-3) or a delayed release (Fig. 5.2 b-4 and b-5). Similar to the friction wrinkle, the wrinkle under the immediate release tends to be more reversible. Moreover, when comparing the friction wrinkle and the plastic wrinkle, it is easy to see that the plastic wrinkle is more persistent in all situations. This experiment demonstrates that our model can successfully generate time-dependent friction and plastic wrinkles, and the high-fidelity is also shown in the varying reversibility of the wrinkles according to their causes and duration. In addition, we refer the reader to the appended video for viewing cloth dynamics.

5.6.2 Tensile Internal Friction and Plasticity

In addition to bending induced wrinkles, our model can also simulate wrinkles caused by tensile deformation (Section 5.4). We design an experiment where mainly in-plane elongation is induced. We fix the four edges of a cloth and press the central area downwards to cause small and large deformation (Fig. 5.3 (a) and (b)). Again, we then either release it immediately (Fig. 5.3 (c, d)) or keep the deformation for a while (Fig. 5.3 (e, f)), then compare the wrinkles.

With small deformation, the wrinkles are mainly caused by the friction; otherwise the plasticity. Again, the friction wrinkles (Fig. 5.3 (c, e)) are not as sharp as the plastic wrinkles (Fig. 5.3 (d, e)), regardless whether the deformation is kept or not. Furthermore, within each type of wrinkles, the longer the deformation is kept, the harder it is for the wrinkles to recover (Fig. 5.3 (g, h)). This experiment demonstrates that our simulator is also effective in simulating wrinkles induced by tensile deformation.

5.6.3 Stribeck and Varying Break-away Force

As we explicitly model the Stribeck and the varying break-away force/stress phenomena, our friction model can simulate them by varying the friction stiffness $K_{friction}$ according to the strain variation rate. We demonstrate this by applying our friction model to simulate the tensile internal friction (plastic model is switched off). As shown in Fig. 5.4 (a) and (b), we apply a gradually increasing force to drag the top edge of a square cloth along its weft direction (up in the figure) while fixing the bottom edge. The experiments are conducted twice with the stretching force increasing at two different rates.

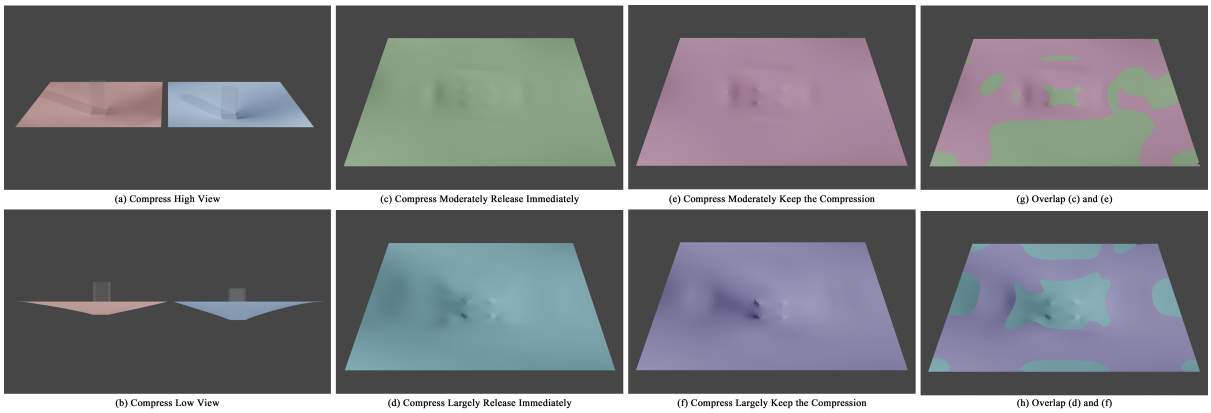


Figure 5.3: We press a square area of two cloths, whose edges are fixed, to different depth (shown in (a) and (b)). The cloths are persistently stretched due to the stretching friction and plastic deformation (shown in (c-f)). Moreover, (g) and (h) show that pressing the cloth for a long time makes the persistent stretching deformation more obvious

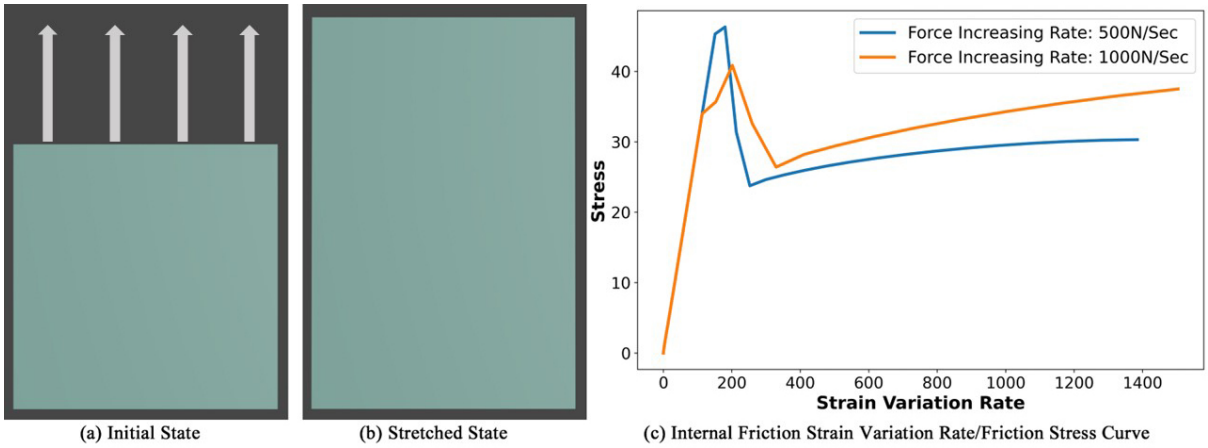


Figure 5.4: A square cloth (a) is stretched along its weft direction by an external force whose magnitude gradually increases (b). (c) comparing the stress-strain variation rate curve ($\dot{\varepsilon}_{vv}$-σ_{fri_vv}) when the magnitude of the dragging force increases at two different rates.

Fig. 5.4 (c) shows two curves where the vertical axis is the friction stress and the horizontal axis is the stretching strain variation rate. First, the two curves share the overall dynamics in common. Both curves first linearly go up (where mainly elasticity happens) until they peak. The peaks are where the stick-slip transaction occurs when the friction stress reaches the maximum stick friction stress, i.e., break-away stress. After the transition, the friction stress starts to fall after the slip friction takes place until it starts to slightly increase again. This is the well-known Stribeck effect.

When comparing the two curves, the major difference is the two forces increases at different rates: 500N/sec (blue) and 1000N/sec (orange). As a result, the peak of the blue curve is higher than that of the orange one, hence the varying break-away forces. This shows that our

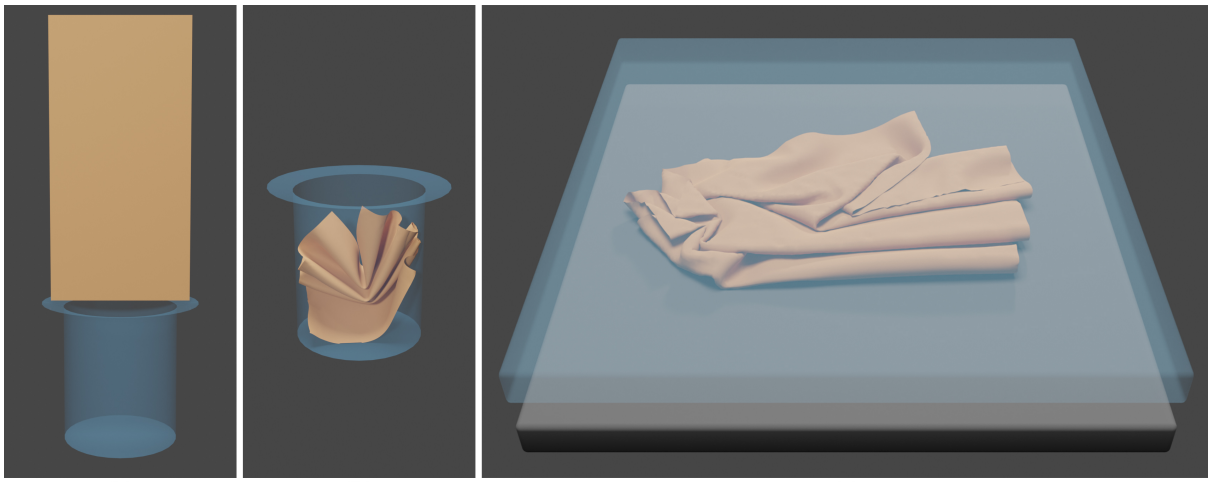


Figure 5.5: The scenarios for simulating friction and plastic wrinkles on cloth. (a) An originally wrinkleless rectangle cloth falls into a cylindrical container due to its self-weight; (b) The cloth is folded moderately due to collision; (c) To cause extreme deformations, we compress the cloth after it falls on the ground.

simulator can simulate varying break-away forces based on the force increase rate: the larger the increase rate is, the smaller the break-away force is, which is widely observed (Richardson and Nolle 1976).

5.6.4 Combined Friction and Plasticity

Combining internal friction and plasticity model allows our simulator to simulate different soft-/sharp wrinkles in small/large deformation. We design an experiment of crumbled cloths. We first drop a wrinkleless cloth into a cylindrical container (Fig. 5.5 (a)) and let the cloth to fold by gravity and collision (Fig. 5.5 (b)). Then our experiment bifurcates into two scenarios. In the first scenario, we remove the container and lift the cloth by picking up its two corners. In the second scenario, we remove the container then add a heavy weight (the transparent object in Fig. 5.5 (c)) and let it fall onto the cloth. Within each scenario, to show the time-dependency, we either immediately lift the cloth or keep it for a while before lifting. After lifting, the cloth will be hung only under the influence of its weight. The experiment aims to mimic everyday scenarios e.g., clothes dropped onto a sofa and left there, sometimes with people sitting on them.

We show the results in Fig. 5.6. First, even under self weight, we notice all scenarios generate wrinkles that are not reversible by gravity. Looking closely and comparing the four results, Fig. 5.6 (c, d) generate sharper wrinkles than Fig. 5.6 (a, b). This is understandable and

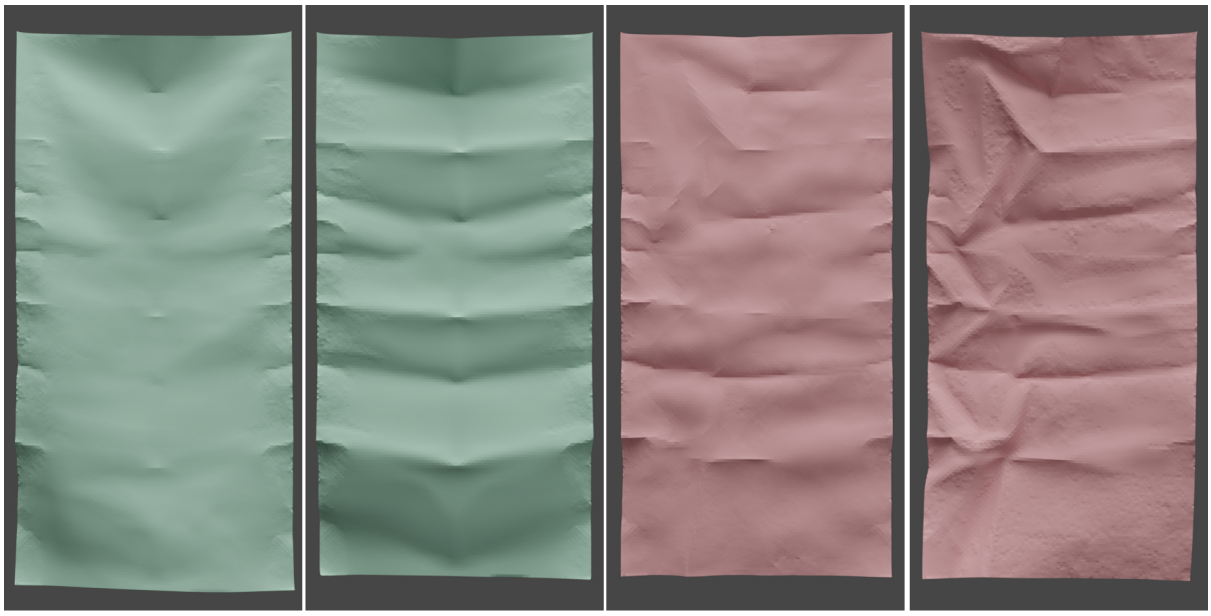


Figure 5.6: The wrinkles on the cloth after lifting it. (a) Immediately lift the cloth after folding moderately; (b)Lift the cloth after keeping the moderate deformation for a long time; (c) Immediately lift the cloth after being compressed by the heavy weight; (d) Lift the cloth after compressing it for a long time.

intuitive because there is no heavy weight placed on the cloth for Fig. 5.6 (a, b). The weight forces larger deformation and therefore makes plasticity prominent in the wrinkle formation. However, the effect of friction is also visible in all cases by soft wrinkles. Overall, combining the internal friction and the plasticity gives a visually realistic combination of both soft and hard wrinkles.

Time-dependence Next, within each scenario (with and without weight), we can clearly see the time-dependency. When there is no weight and the cloth is immediately lifted (Fig. 5.6 (a)), the deformation does not leave many wrinkles as the process of lifting and hanging leads to stretching under self weight, which can reverse some of the soft wrinkles caused by the internal friction. Comparatively, when the cloth is left for a while Fig. 5.6 (b)), the friction wrinkles start to harden under self weight. Wrinkles become more visible and resist recovery. Note that most of the hardened wrinkles here are still friction wrinkles. Similarly, when there is a weight, even for a short period of time, plastic wrinkles are still formed (Fig. 5.6 (c)). But since the time is short, some of the wrinkles are partially recovered after lifting before they harden completely. If the weight is left for a while, then sharp and irreversible wrinkles appear (Fig. 5.6 (d)). Overall, this experiment shows both the internal friction and the plasticity, when combined, are still

Table 5.2: Simulation parameters of simulate cotton (plain woven), tannin (cotton twill), and polyester cloths.

Cloth	ρ	k_b	k_{11}	k_{22}	k_{12}	k_{33}	C_0	C_1	C_2
Cotton	0.1	1e-6	200.0	200.0	0.2	20.0	3e-6	3e-6	0.8
Tannin	0.2	3e-5	200.0	200.0	0.2	150.0	6e-6	3e-5	0.8
Polyester	0.15	1e-6	100.0	100.0	0.2	20.0	1e-6	1e-6	0.8
Cloth	k_c	ε_0	ε_{inf}	ε_s	τ_f	K_{h0}	g	τ_p	ε_{Y0}
Cotton	3e-6	0.1	1.2	0.1	30.0	1e-6	0.99	30.0	1.5
Tannin	5e-5	0.2	1.2	0.1	30.0	3e-5	0.99	30.0	1.2
Polyester	5e-7	0.1	0.1	0.1	30.0	1e-6	0.99	30.0	3.1

time-dependent and can generate highly plausible wrinkles.

Friction vs Plasticity Comparing Fig. 5.6 (b) and (c), we see obvious wrinkles in both scenarios. But looking closely, there are steeply sharp wrinkles with high curvature at the peak of the wrinkle ridges in Fig. 5.6 (c) which are not presented in Fig. 5.6 (b). This suggests that the internal friction, even after dwell, is less likely to form wrinkles as sharp as plastic wrinkles. This is physically correct and visually intuitive. The combination of the internal friction and the plasticity enables the simulator to respond to all kinds of deformation and external forces realistically.

5.6.5 Model Versatility

Cloths made from different materials usually exhibit distinctive wrinkles even after the same deformation, due to different material properties, e.g., bending stiffness, inter-yarns/fibers friction, and plasticity (Fan and Hunter 2009). Our model comes with parameters governing the material properties, which can simulate a wide range of materials when set properly. Below, we show three types of materials: Cotton (plain woven), Tannin (twill cotton) and Polyester. We choose these types as they are typical fabrics and common in clothes. Also, they can generate visually distinguishable wrinkles. Finally, we do not claim that they are truly the cotton, tannin and polyester material from the perspective of physics, as not all parameters we use can be easily measured in experiments, hence no direct verification. We refer to them mainly based on their visual appearances.

The parameters are shown in Table 5.2. Cotton cloths are used to make table cloths. It is soft (low bending stiffness), but difficult to be elongated by stretching (large stretching stiffness).

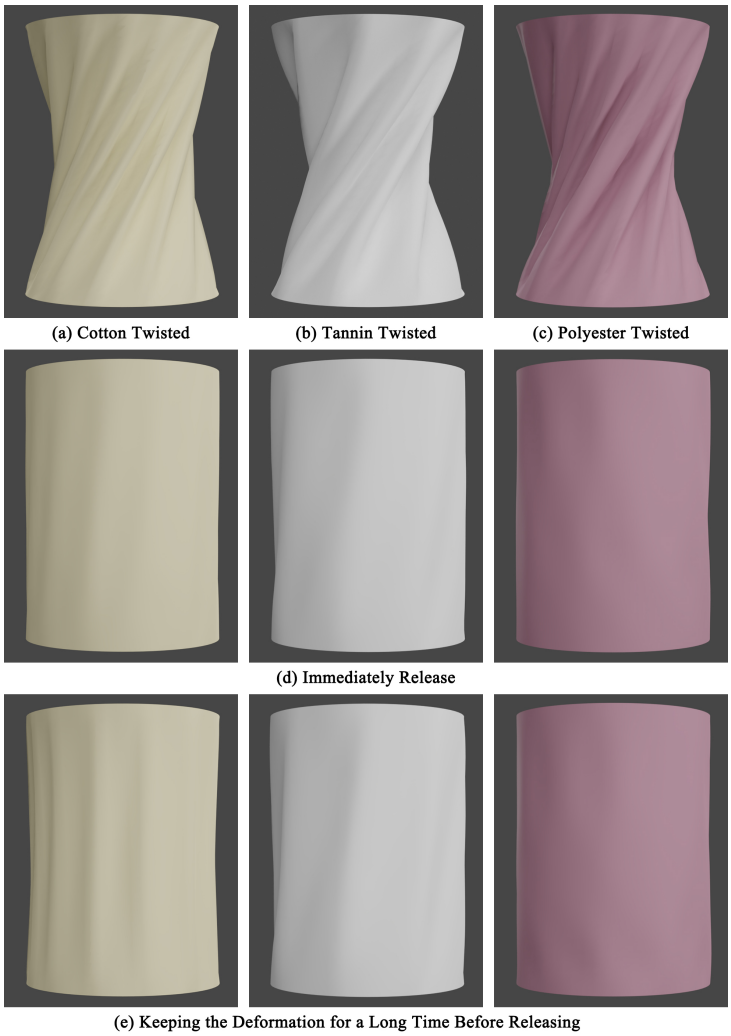


Figure 5.7: Wrinkles on the twisted cloths made from different materials: (a) Cotton; (b) Tannin; (c) Polyester. Cotton and Tannin are more likely to generate wrinkles than Polyester. Moreover, keeping the cloths in the twisted state (e) makes the wrinkles more obvious than those on the cloths that are released immediately (d).

Wrinkles are likely to appear on plain woven cotton, so we give this cloth a low ε_{thres} and ε_Y. Tannin is thick, solid and durable, and is usually used to make jeans. It has high bending stiffness and density because it is usually thick and heavy. Wrinkles also commonly appear on heavily used tannin cloths. We also give this cloth a low ε_{thres} and ε_Y. Conversely, polyester is a very light material and commonly used to make sports outfits because it can be deformed extremely without wrinkles. It exhibits good wrinkle recovery and is difficult to have plastic deformations. Therefore, we give this cloth a small density/bending stiffness/stretching stiffness. To simulate good wrinkle recovery, we give polyester a low ε_0 and ε_{inf}. Also, we set its ε_{Y0} high so that it almost cannot have plastic wrinkles.

As stretching and bending have been shown before, we use twisting in this experiment. The

Table 5.3: Consumed time per simulation step (seconds/step) when using different resolution mesh. Turning on our friction and plastic model only trivially slow down the simulation.

Model	2K Mesh	7K Mesh	27K Mesh
Friction/Plastic on	0.279	1.319	12.064
Friction/Plastic off	0.256	1.286	11.518

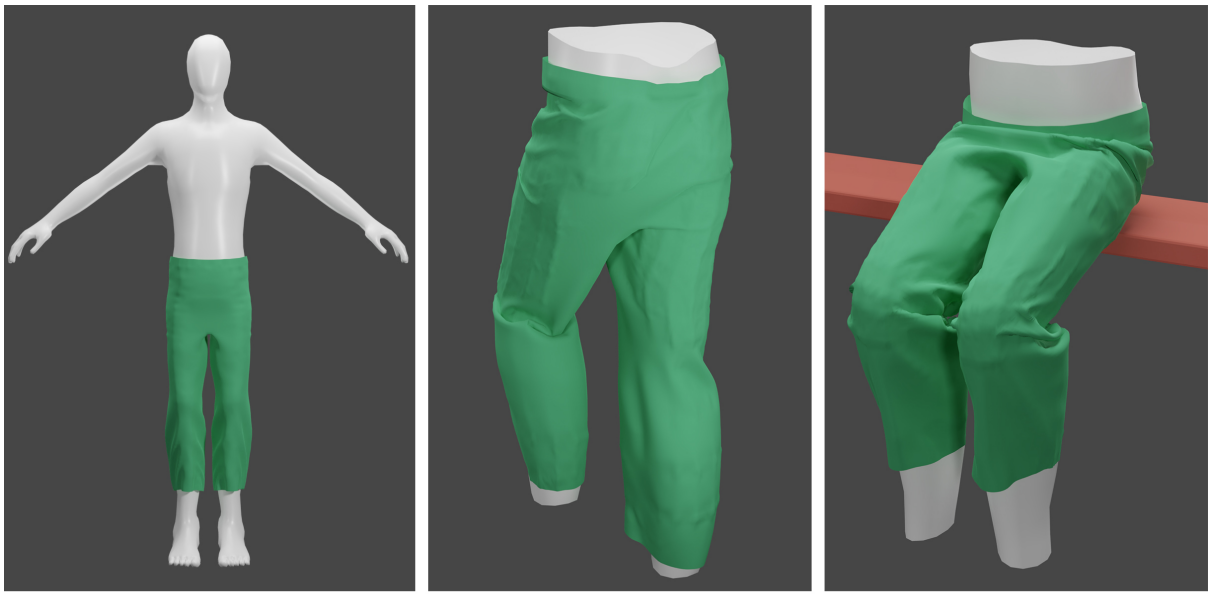
cloths are made into cylinders. We twist the top of every cloth by 0.25π in the clockwise direction (from the top view) along its central axis with its bottom fixed. Then, we twist in the counter-clockwise direction by 0.25π to return its initial state and observe the wrinkles caused by the deformation. The results are shown in Fig. 5.7. As expected, Cotton gives the most obvious wrinkles and Polyester gives the least. Moreover, keeping the deformation for a long time makes the wrinkles more obvious, which is shown in both Cotton and Tannin. Polyester is less affected by deformation duration as there is little internal friction and plasticity. Overall, our simulator is capable of simulating high-fidelity wrinkles that reflect cloth materials. We refer the reader to the appended video for viewing cloth dynamics.

5.6.6 Performance

We use the twisting experiments to show the performance of our simulator. This is to mainly measure the time needed for the internal friction and the plasticity, in the absence of other factors such as collisions and external frictions. Table 5.3 shows the time per simulation step when the cloth mesh has different resolutions: approximately 2K vertices, 7K vertices, and 27K vertices. Note our implementation is not optimized particularly for speed, hardware or parallel computing. The result demonstrates our friction and plastic models only cause a trivial addition to the computing and therefore they can be incorporated easily in simulation applications.

5.6.7 Wrinkles on Garments

Finally, we demonstrate that our simulator can be used for garment simulation under complex human motions, which is crucial for many applications, from fashion design to animation. We simulate trousers with 20k vertices on a human body (Fig. 5.8 (a)), under deformation caused by two motions: lifting the left leg (Fig. 5.8 (b)) and sitting down (Fig. 5.8 (c)). Both motions are commonly seen in daily life. Lifting leg is a good example of moderate deformations asymmetrically distributed between two legs, while sitting down involves multiple regions of large deformations around the pelvis and knees.

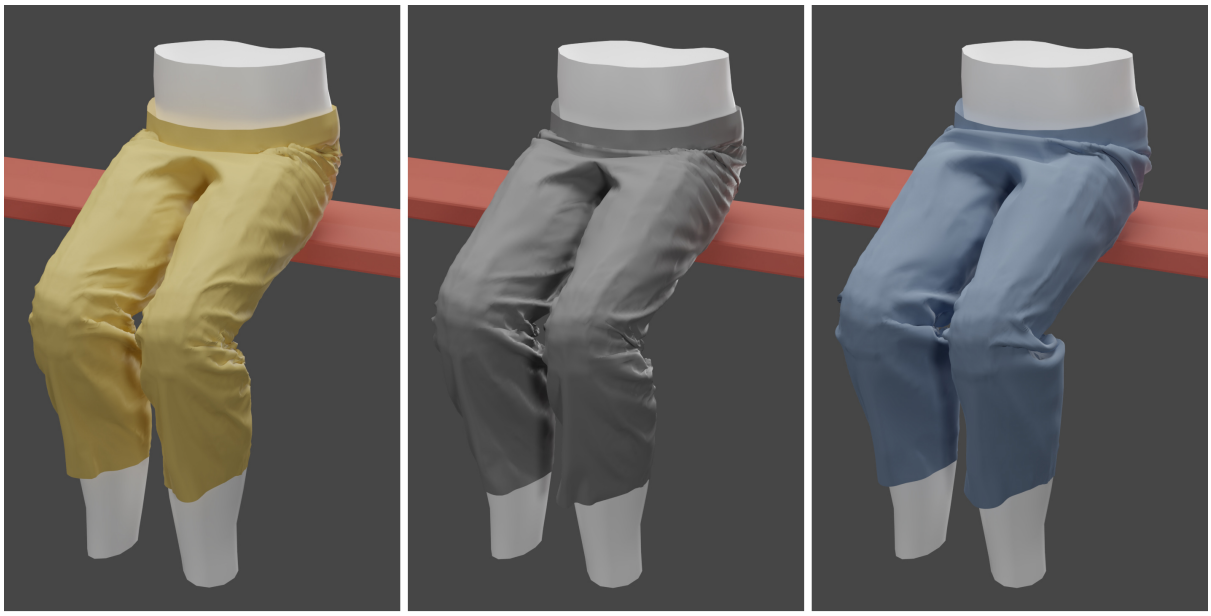


(a) Garment on body initial state (b) Lifting Leg (c) Sitting

Figure 5.8: Trousers simulation. (a) The trousers on A-pose human body does not have wrinkles; (b) Lifting a leg will deform the trousers moderately; (c) Sitting down will cause larger deformations.

We first show results in Tannin trousers in Fig. 5.10. In each motion, we keep the lifting or sitting for a short and a long period of time. The leg lifting mainly causes wrinkles behind the left knee, with some mild wrinkles behind the bottom. As expected, the wrinkles after long deformation (Fig. 5.10 (b)) are sharper behind the knee and the bottom than (Fig. 5.10 (a)). Since leg lifting causes moderate wrinkles, the visual difference between short and long deformation is visible but subtle. Comparatively, the difference is more visible in Sitting where more sharp wrinkles are formed in Fig. 5.10 (d) than (Fig. 5.10 (c). The differences are especially noticeable not only in the areas behind the bottom and around the knees, but also near the crotch.

Next, we also show the trousers without the human body in Fig. 5.10 (e-h). This is to show some wrinkles are generated and maintained mainly due to the collisions between the body and the trousers. Without the human body, the trousers are more stretched under gravity and some wrinkles disappear. However, these disappeared wrinkles tend to be the ones that are soft and mainly caused by the internal friction. These wrinkles are more easily reversible. Finally, we show trousers made from different materials from behind when draping in Fig. 4.1, and from the front when body-cloth collisions are present in Fig. 5.9. We also show another example of a T-shirt from T-pose to arm-bending (Fig. 5.11), with short and long duration of deformation (Fig. 5.12).



(a) Plain Woven Cotton (b) Polyester (c) Tannin

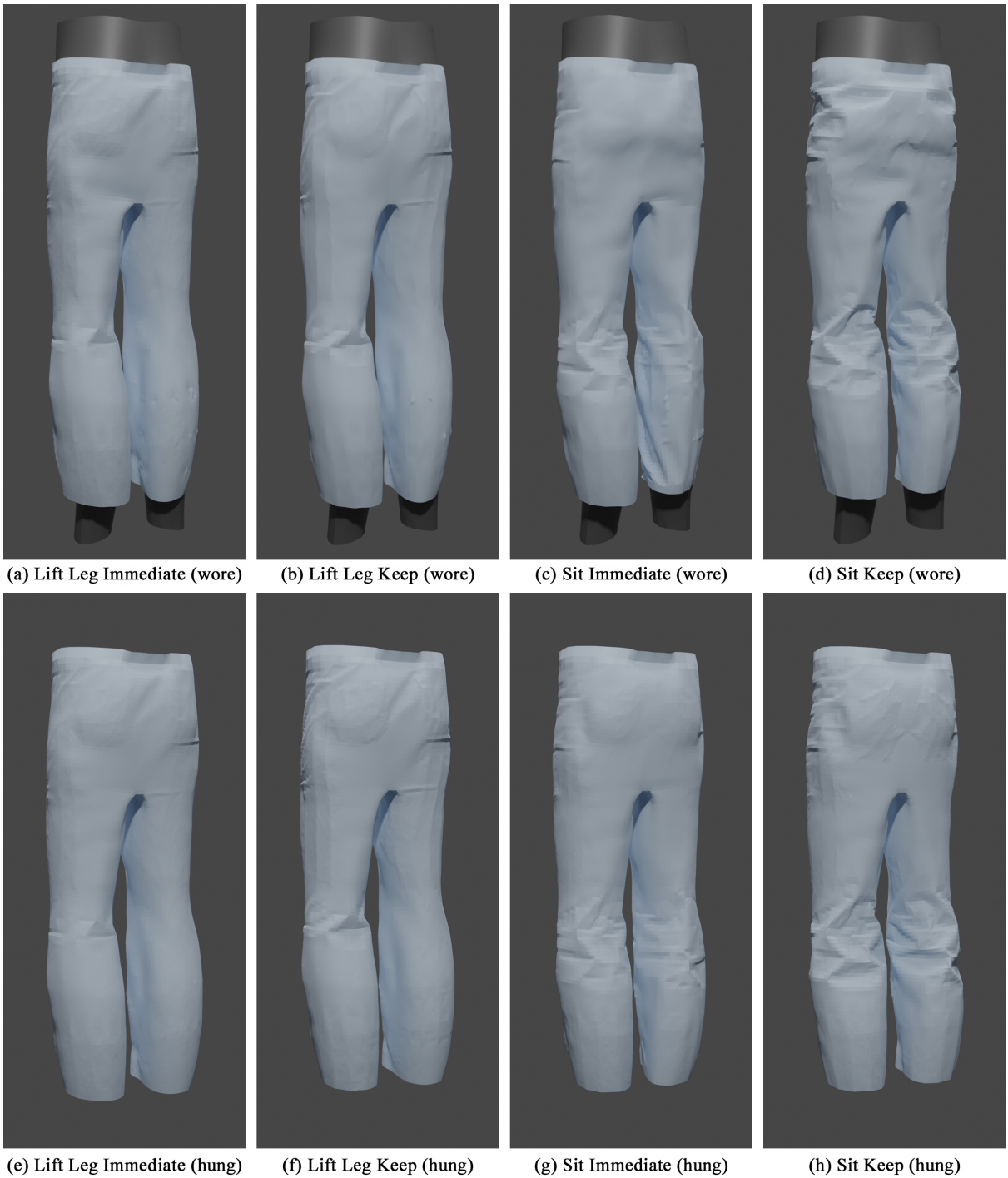
Figure 5.9: Front view of sitting where trousers are made from different fabrics.

5.6.8 Comparison Experiments

To our best knowledge, there is no cloth simulator that combines internal friction and plasticity. Further, there is no cloth simulator that models wrinkle time-dependence. So we choose two baseline methods that are closest to ours. One considers the internal friction only based on Dahl's Model (Miguel et al. 2013). The other considers plasticity only (Narain et al. 2013; Grinspun 2008). Note neither models time-dependence. We refer the reader to the appended video for viewing cloth dynamics.

Compare with Dahl's Model

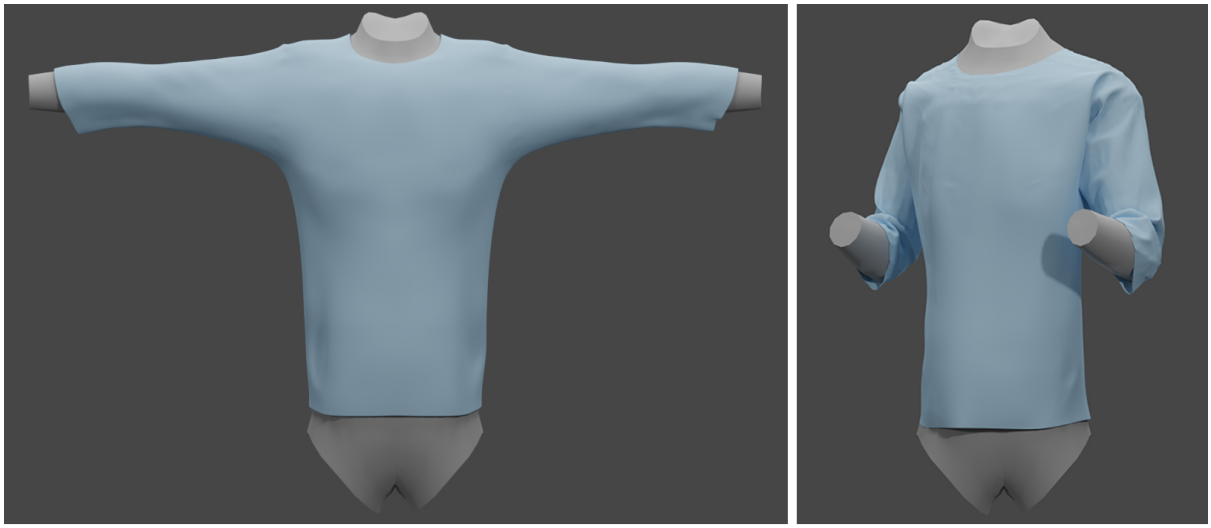
Dahl's friction to is used to model cloth internal friction in (Miguel et al. 2013). Their experiments show that the friction model can also form cloth wrinkles. However, without considering the dwell effect, it cannot simulate time-dependent friction wrinkles. We use the simulation scenario in Fig. 5.5 (b) (without weight) and use Dahl's friction model to simulate the friction wrinkles. Fig. 5.13 (a) and (b) show, when using Dahl's friction model, the wrinkles are identical even though their deformation duration is different. Therefore, Dahl's friction model cannot simulate time-dependent friction wrinkles. In addition, due to the lack of stick friction, Dahl's friction model is not feasible for simulating wrinkles. To simulate the obvious wrinkles as shown in Fig. 5.13, we have to follow (Miguel et al. 2013)'s suggestion: using large friction force, slowing down the motion, and even frequently set velocity to zero to prevent wrinkles



(a) Lift Leg Immediate (wore) (b) Lift Leg Keep (wore) (c) Sit Immediate (wore) (d) Sit Keep (wore)

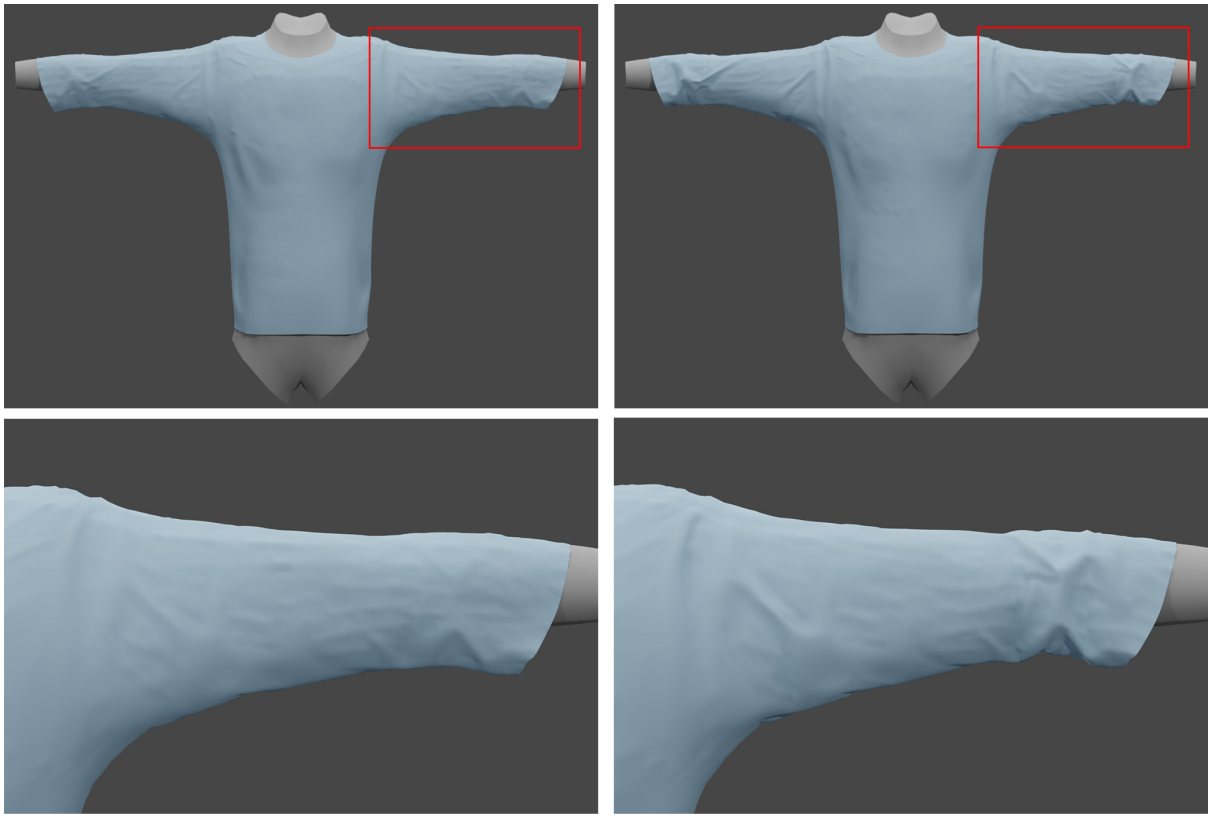
(e) Lift Leg Immediate (hung) (f) Lift Leg Keep (hung) (g) Sit Immediate (hung) (h) Sit Keep (hung)

Figure 5.10: (a-d) and (e-h) are with and without human body. The wrinkles caused by sitting (a,b,e,f) are more obvious that those caused by lifting leg (c,d,g,h) because sitting causes larger deformations. Moreover, the wrinkles on (b, d, f, h) are sharper and deeper than those on (a, c, e, h). Therefore, keeping deformations for a long time makes the wrinkles more obvious.



(a) Initial State (b) Deformed State

Figure 5.11: Top garment simulation.(a)The top on a T-pose human body does not have wrinkles; (b) Bending the arm folds the elbow areas and the underarm area of the sleeves.



(a) Wrinkles without keeping the deformation (b) Wrinkles after keeping the deformation

Figure 5.12: The wrinkles on the top garment after returning to the initial t-pose. Keeping the pose of a long time makes the wrinkles(b) more obvious that the wrinkles which formed by immediate returning to the t-pose(a).

from disappearing. Otherwise, wrinkles would look very subtle. We refer the reader to the appended video for viewing cloth dynamics.

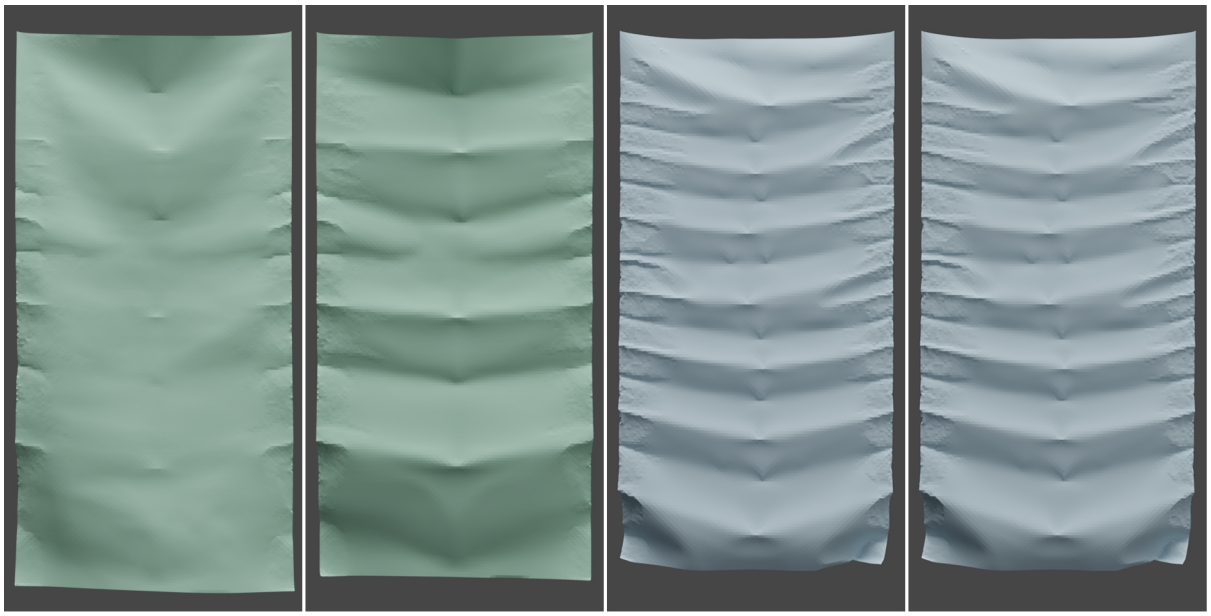


Figure 5.13: The friction wrinkles simulated by the Dahl's friction model (Miguel et al. 2013) with different deformation duration: (a-1) lift immediately; (a-2) lift after long-time keeping. Due to the lack of time-dependency, the simulated wrinkles simulated by the Dahl's model do not vary with time. By contrast, our simulator can simulate time-dependent wrinkles(b-1:lift immediately; b-2:lift after long-time keeping).

Compare with Hardening Plastic Model.

Plastic models are commonly used for simulating cloth wrinkles in graphics. We choose the hardening plastic model as the baseline which has been used in (Narain et al. 2013; Grinspun 2008). We twist the two edges of a rectangular cloth to cause plastic deformation (Fig. 5.14 (a-b)). Our model and the baseline use the same yield strain, ε_Y. As shown in Fig. 5.14 (c) and (d), keeping the deformation for different duration affects the wrinkles when using our model. Conversely, as shown in Fig. 5.14(e) and (f), the wrinkles simulated by the baseline method are identical no matter how long the deformation is kept. Moreover, due to the lack of internal friction, the cloths in Fig. 5.14 (e) and (f) have much fewer wrinkles because plastic wrinkles only appears in large deformations. We refer the reader to the appended video for viewing cloth dynamics.

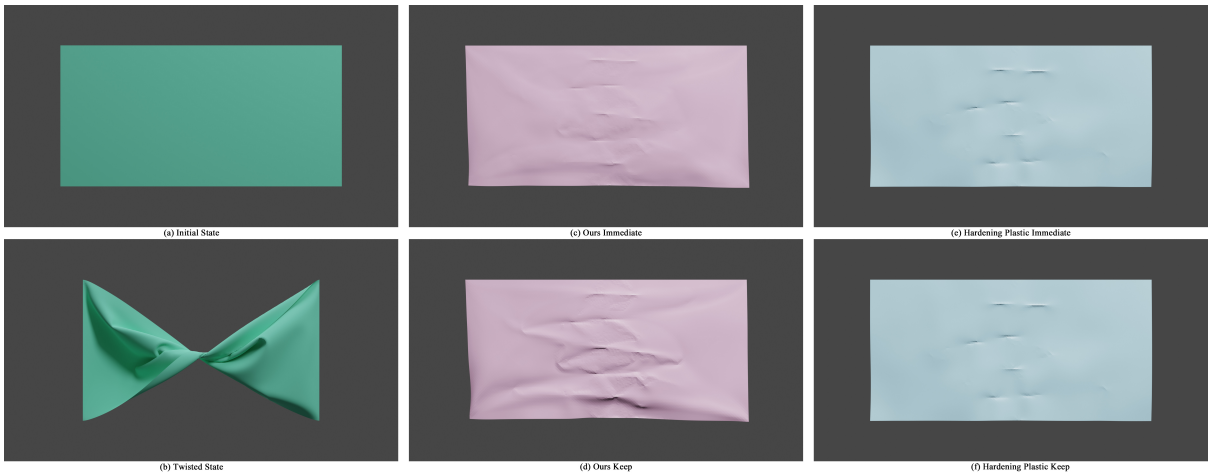


Figure 5.14: We twist the two edges of a initially flat rectangular cloth (a) in opposite direction by $\frac{\pi}{2}$ to cause wrinkles (b). After flattening the twisted cloth, the wrinkles simulated by our model (pink) and the hardening plastic model (blue) (Narain et al. 2013) with different deformation duration: (c, d) Immediately flattened after twisting; (e, f) Twisted for a while before flattening.

5.7 Discussion and Conclusion

We have introduced a new cloth simulator that is capable of generating realistic wrinkles. To our best knowledge, our method is the first one that investigates the temporal aspect of wrinkle formation. To this end, we proposed a new physics model combining time-dependent internal friction and plasticity. Through experiments, we have demonstrated our method achieves high visual fidelity by introducing more complicated cloth wrinkle mechanics: (1) the interplay between internal friction and plastic deformation; (2) time-dependence.

In this work, our simulator is not optimized for speed in comparison with the state-of-art (e.g., real-time). In future, we will refactor our algorithms to leverage the parallel computing capability of GPUs, so it becomes performance ready for applications such as games and virtual try-on. In addition, our friction and plastic models inevitably introduce more simulation parameters which further increases the difficulty of the parameter tuning. We plan to introduce a differentiable cloth simulation based on this work so that it can estimate the simulation parameters from data in our future work. Finally, theoretically, our friction model and plastic model can be used in a plug-and-play manner and transferred to other cloth simulators, e.g., yarn-level cloth simulators. We will explore how to integrate them into various cloth simulators.

Chapter 6

Conclusion

This thesis has reported our three works on improving physics-based cloth simulation realism by diving into the fine-grained physics in cloth and accurately estimating cloth physical parameters. By modeling cloth micro-mechanics at yarn-level, our first work (Chapter 3) introduces a new differentiable cloth simulator that captures individual yarns' mechanical behaviors and their interactions. Thanks to our novel friction and shear force models, our simulator is fully differentiable so that can conduct gradient-based optimization to estimate cloth physical parameters with outstanding sample efficiency. Compared with an existing sheet-level differentiable cloth simulator, it can more accurately estimate cloth physical parameters and then more realistically reproduce the observed cloth dynamics. Moreover, it can also naturally embed woven patterns and handle blend woven cloth. In addition, cloth micro-mechanics determines its macro-level physical properties: Cloths usually exhibit material heterogeneity where a cloth's physical properties randomly vary across its geometry. This results from the uneven physical properties at their micro-level, e.g., yarns cannot be perfectly identical because of production errors. Our second work (Chapter 4) proposes a Bayesian differentiable cloth simulator which models cloth physical parameters as probability distributions such that is can encode cloth material heterogeneity. We apply our simulator to estimate cloth physical properties from Cusick drape testing because it is accurate, widely acknowledged, and reproduciable. More importantly, it can obviously reflects cloth material heterogeneity, i.e., varied drape shapes of cloth samples from the same fabric and asymmetric drape shape. This work also releases a Cusick drape dataset consists of 660 samples which includes both 2D cloth drape silhouettes and reconstructed 3D meshes. Compared with homogeneous cloth models, our Bayesian differentiable cloth simula-

tor can more accurately estimate cloth physical properties and then more realistically simulate cloths with reflecting material heterogeneity. Furthermore, cloth fine-grained physics also has a significant influence on cloth persistent wrinkles which are the combinational effect of the friction between yarns and fibers, and yarns' and fibers' plastic deformations. Our third work (Chapter 5) proposes our novel time-dependent friction model and plastic model, and integrates them into our physics-based cloth simulator to simulate cloth persistent wrinkles. Compared with previous works, our simulator not only can more realistically simulate the differentiable wrinkles due to internal friction or plasticity according to cloth deformation, but can also naturally simulate time-dependent persistent wrinkles. In conclusion, the results demonstrate that fine-grained physics is indispensable to realistic cloth simulation and would be an important trend of further research in pursuing cloth simulation realism.

Nevertheless, there is still room for further improvement in our current work. First, simulating cloth micro-mechanics is computationally expensive. In computer graphics, the simulation efficiency is also an important concern. Our current implementations mainly rely on the single machine CPU parallel to speed up the simulation and the differentiable simulation. Their running speeds are acceptable in our experiment settings but would slow down when simulating more complicated scenarios or integrating finer cloth mechanics. To gain a more efficient implementation, it would be more feasible to leverage the GPU(s) parallel, mix CPU/GPU, or distribute the simulation to a cluster. Second, the same as most of the state-of-the-art differentiable cloth simulators, our differentiable cloth simulators are implemented by PyTorch's (Paszke et al. 2019) C++ extension to leverage its automatic gradient computation functionality and well-implemented gradient-based optimizers. However, as PyTorch is mainly designed for building machine/deep learning models, its efficiency will dramatically degrade when being applied to implement differentiable cloth simulators because the vectorization is usually broken. In addition, increasing the number of simulation steps will complicate the gradient graph and makes back-propagation intractable. Unfortunately, there is no automatic differentiation framework designed for the differentiable cloth simulation. Therefore, developing a new framework for differentiable cloth simulation would be a promoting research direction in the future. Third, it is laborious, costly, and time-consuming to test real cloths and build a cloth database. The number of samples in our Cusick drape database is not very great. In theory, our Bayesian differentiable cloth simulator would achieve better results if there were more training samples. Currently, there is rare public real cloth dataset, and building a benchmark database would be

a great contribution to the cloth simulation community. Last but not least, our time-dependent friction model and plastic model introduce more simulation parameters for simulating persistent wrinkles. Tuning these parameters to realistically simulate the observed cloth by trial and error is tedious and slow. As these models involve indifferentiable functions, they cannot be simply converted into a differentiable cloth simulator by integrating automatic differentiation.

We plan to resolve these problems in the future work. An efficient and user-friendly coding framework for the differentiable cloth simulation would be a solid base for implementing new ideas and demonstrating hypotheses. Our next work is going to build a differentiable cloth simulation framework that can leverage CPU/GPU to efficiently run simulation and conduct gradient optimization. In addition, a real cloth dataset is indispensable to the research on cloth parameter estimation and cloth simulation. We am going to build a benchmark database that contains comprehensive textile measurements of various daily used cloths. Finally, we will introduce a differentiable version of the internal friction model and plastic model such that our persistent cloth wrinkles simulator can be converted into a differentiable cloth simulation and estimate the simulation parameters by observing the target cloths.

References

Alter, Ethan (2008). "Alvin and the Chipmunks". In: *Film Journal International* 111.2, pp. 40–41.

Amos, Brandon and Kolter, J. Zico (2017). "OptNet: differentiable optimization as a layer in neural networks". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, pp. 136–145. URL: https://proceedings.mlr.press/v70/amos17a/amos17a.pdf.

Armstrong-Helouvry, Brian (2012). *Control of machines with friction*. Vol. 128. Springer Science & Business Media. ISBN: 978-1-4615-3972-8. DOI: 10.1007/978-1-4615-3972-8. URL: https://doi.org/10.1007/978-1-4615-3972-8.

Ascher, Uri M and Boxerman, Eddy (2003). "On the modified conjugate gradient method in cloth simulation". In: *The Visual Computer* 19, pp. 526–531. ISSN: 1432-2315. DOI: 10.1007/s00371-003-0220-4. URL: https://doi.org/10.1007/s00371-003-0220-4.

Awrejcewicz, J. (Sept. 1988). "Chaotic motion in a nonlinear oscillator with friction". en. In: *KSME Journal* 2.2, pp. 104–109. ISSN: 1738-494X. DOI: 10.1007/BF02953669. URL: https://doi.org/10.1007/BF02953669 (visited on 04/22/2021).

Balandat, Maximilian, Karrer, Brian, Jiang, Daniel R., Daulton, Samuel, Letham, Benjamin, Wilson, Andrew Gordon, and Bakshy, Eytan (2020). *BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization*. DOI: 10.48550/arXiv.1910.06403. arXiv: 1910.06403 [cs.LG]. URL: https://doi.org/10.48550/arXiv.1910.06403.

Baraff, David (1994). "Fast contact force computation for nonpenetrating rigid bodies". In: *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*.

SIGGRAPH '94. New York, NY, USA: Association for Computing Machinery, pp. 23–34. ISBN: 0897916670. DOI: 10.1145/192161.192168. URL: https://doi.org/10.1145/192161.192168.

Baraff, David and Witkin, Andrew (1998). "Large steps in cloth simulation". In: *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. New York, NY, USA: Association for Computing Machinery, pp. 43–54. ISBN: 0897919998. DOI: 10.1145/280814.280821. URL: https://doi.org/10.1145/280814.280821.

Bear, Daniel M., Wang, Elias, Mrowca, Damian, Binder, Felix J., Tung, Hsiao-Yu Fish, Pramod, R. T., Holdaway, Cameron, Tao, Sirui, Smith, Kevin, Sun, Fan-Yun, Fei-Fei, Li, Kanwisher, Nancy, Tenenbaum, Joshua B., Yamins, Daniel L. K., and Fan, Judith E. (2022). *Physion: Evaluating Physical Prediction from Vision in Humans and Machines*. DOI: 10.48550/arXiv.2106.08261. arXiv: 2106.08261 [cs.AI]. URL: https://doi.org/10.48550/arXiv.2106.08261.

Beck, Christian, Hutzenthaler, Martin, Jentzen, Arnulf, and Kuckuck, Benno (2023). "An overview on deep learning-based approximation methods for partial differential equations". In: *Discrete and Continuous Dynamical Systems - B* 28.6, pp. 3697–3746. ISSN: 1531-3492. DOI: 10.3934/dcdsb.2022238. URL: https://www.aimsciences.org/article/id/639b367cb2114e413cb39c48.

Belbute-Peres, Filipe de A., Smith, Kevin A., Allen, Kelsey R., Tenenbaum, Joshua B., and Kolter, J. Zico (2018). "End-to-end differentiable physics for learning and control". In: NIPS'18. Montréal, Canada: Curran Associates Inc., pp. 7178–7189. DOI: 10.5555/3327757.3327820. URL: https://dl.acm.org/doi/10.5555/3327757.3327820.

Al-Bender, Farid, Lampaert, Vincent, and Swevers, Jan (2004). "A novel generic model at asperity level for dry friction force dynamics". In: *Tribology Letters* 16, pp. 81–93. ISSN: 1573-2711. DOI: 10.1023/B:TRIL.0000009718.60501.74. URL: https://doi.org/10.1023/B:TRIL.0000009718.60501.74.

Bender, Jan, Koschier, Dan, Charrier, Patrick, and Weber, Daniel (Nov. 2014a). "Position-based simulation of continuous materials". In: *Comput. Graph.* 44.C, pp. 1–10. ISSN: 0097-8493. DOI: 10.1016/j.cag.2014.07.004. URL: https://doi.org/10.1016/j.cag.2014.07.004.

Bender, Jan, Müller, Matthias, and Macklin, Miles (2015). "Position-Based Simulation Methods in Computer Graphics". In: *EG 2015 - Tutorials*. Ed. by M. Zwicker and C. Soler. The Eurographics Association. DOI: 10.2312/egt.20151045. URL: http://dx.doi.org/10.2312/egt.20151045.

Bender, Jan, Müller, Matthias, Otaduy, Miguel A., Teschner, Matthias, and Macklin, Miles (2014b). "A Survey on Position-Based Simulation Methods in Computer Graphics". In: *Computer Graphics Forum* 33.6, pp. 228–251. DOI: https://doi.org/10.1111/cgf.12346. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12346.

Benusiglio, Adrien, Mansard, Vincent, Biance, Anne-Laure, and Bocquet, Lydéric (2012). "The anatomy of a crease, from folding to ironing". In: *Soft Matter* 8 (12), pp. 3342–3347. DOI: 10.1039/C2SM07151G. URL: http://dx.doi.org/10.1039/C2SM07151G.

Bertiche, H., Madadi, M., Tylson, E., and Escalera, S. (Oct. 2021a). "DeePSD: Automatic Deep Skinning And Pose Space Deformation For 3D Garment Animation". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 5451–5460. DOI: 10.1109/ICCV48922.2021.00542. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00542.

Bertiche, Hugo, Madadi, Meysam, and Escalera, Sergio (Dec. 2021b). "PBNS: physically based neural simulation for unsupervised garment pose space deformation". In: *ACM Trans. Graph.* 40.6. ISSN: 0730-0301. DOI: 10.1145/3478513.3480479. URL: https://doi.org/10.1145/3478513.3480479.

Bertiche, Hugo, Madadi, Meysam, and Escalera, Sergio (Nov. 2022). "Neural Cloth Simulation". In: *ACM Trans. Graph.* 41.6. ISSN: 0730-0301. DOI: 10.1145/3550454.3555491. URL: https://doi.org/10.1145/3550454.3555491.

Bertsimas, Dimitris and Tsitsiklis, John (1993). "Simulated Annealing". In: *Statistical Science* 8.1, pp. 10–15. DOI: 10.1214/ss/1177011077. URL: https://doi.org/10.1214/ss/1177011077.

Bhat, Kiran S., Twigg, Christopher D., Hodgins, Jessica K., Khosla, Pradeep K., Popović, Zoran, and Seitz, Steven M. (2003). "Estimating cloth simulation parameters from video". In:

Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '03. San Diego, California: Eurographics Association, pp. 37–51. ISBN: 1581136595. DOI: 10.5555/846276.846282. URL: https://dl.acm.org/doi/10.5555/846276.846282.

Blundell, Charles, Cornebise, Julien, Kavukcuoglu, Koray, and Wierstra, Daan (2015). "Weight uncertainty in neural networks". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, pp. 1613–1622. DOI: 10.5555/3045118.3045290. URL: https://proceedings.mlr.press/v37/blundell15.pdf.

Boisse, P, Gasser, A, and Hivet, G (2001). "Analyses of fabric tensile behaviour: determination of the biaxial tension–strain surfaces and their use in forming simulations". In: *Composites Part A: Applied Science and Manufacturing* 32.10, pp. 1395–1414. ISSN: 1359-835X. DOI: https://doi.org/10.1016/S1359-835X(01)00039-2. URL: https://www.sciencedirect.com/science/article/pii/S1359835X01000392.

Bonet, Javier and Wood, Richard D. (2008). *Nonlinear Continuum Mechanics for Finite Element Analysis*. 2nd ed. Cambridge University Press. DOI: 10.1017/CBO9780511755446. URL: https://doi.org/10.1017/CBO9780511755446.

Bouaziz, Sofien, Martin, Sebastian, Liu, Tiantian, Kavan, Ladislav, and Pauly, Mark (July 2014). "Projective dynamics: fusing constraint projections for fast simulation". In: *ACM Trans. Graph.* 33.4. ISSN: 0730-0301. DOI: 10.1145/2601097.2601116. URL: https://doi.org/10.1145/2601097.2601116.

Bouman, Katherine L., Xiao, Bei, Battaglia, Peter, and Freeman, William T. (2013). "Estimating the Material Properties of Fabric from Video". In: *2013 IEEE International Conference on Computer Vision*, pp. 1984–1991. DOI: 10.1109/ICCV.2013.455. URL: https://doi.org/10.1109/ICCV.2013.455.

Breen, David E., House, Donald H., and Getto, Phillip H. (1992). "A physically-based particle model of woven cloth". In: *The Visual Computer* 8, pp. 264–277. ISSN: 1432-2315. DOI: 10.1007/BF01897114. URL: https://doi.org/10.1007/BF01897114.

Breen, David E., House, Donald H., and Wozny, Michael J. (1994a). "A Particle-Based Model for Simulating the Draping Behavior of Woven Cloth". In: *Textile Research Journal* 64.11, pp. 663–685. DOI: 10.1177/004051759406401106. URL: https://doi.org/10.1177/004051759406401106.

Breen, David E., House, Donald H., and Wozny, Michael J. (1994b). "Predicting the drape of woven cloth using interacting particles". In: SIGGRAPH '94. New York, NY, USA: Association for Computing Machinery, pp. 365–372. ISBN: 0897916670. DOI: 10.1145/192161.192259. URL: https://doi.org/10.1145/192161.192259.

Brenner, F.C. and Chen, C.S. (1964). "The Mechanical Behavior of Fabrics: Part I: Wrinkling". In: *Textile Research Journal* 34.6, pp. 505–517. DOI: 10.1177/004051756403400605. URL: https://doi.org/10.1177/004051756403400605.

Bridson, R., Marino, S., and Fedkiw, R. (2003). "Simulation of clothing with folds and wrinkles". In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '03. San Diego, California: Eurographics Association, pp. 28–36. ISBN: 1581136595. DOI: 10.5555/846276.846281. URL: https://dl.acm.org/doi/10.5555/846276.846281.

Bridson, Robert, Fedkiw, Ronald, and Anderson, John (July 2002). "Robust treatment of collisions, contact and friction for cloth animation". In: *ACM Trans. Graph.* 21.3, pp. 594–603. ISSN: 0730-0301. DOI: 10.1145/566654.566623. URL: https://doi.org/10.1145/566654.566623.

British Standards Institute (1973). *BS 5058:1973: Method for the assessment of drape of fabrics*. eng. URL: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000000087095.

British Standards Institute (1998). *BS EN ISO 9073-9:1998*. eng. URL: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000001522897.

British Standards Institute (2008). *BS EN ISO 9073-9:2008: Textiles. Test methods for nonwovens: Determination of drapability including drape coefficient*. eng. URL: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000030128826.

Brochu, Eric, Cora, Vlad M, and De Freitas, Nando (2010). "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: *arXiv preprint arXiv:1012.2599*. DOI: https://doi.org/10.48550/arXiv.1012.2599. URL: 10.48550/arXiv.1012.2599.

Brochu, Tyson, Edwards, Essex, and Bridson, Robert (July 2012). "Efficient geometrically exact continuous collision detection". In: *ACM Trans. Graph.* 31.4. ISSN: 0730-0301. DOI: 10.1145/2185520.2185592. URL: https://doi.org/10.1145/2185520.2185592.

Cao, J., Akkerman, R., Boisse, P., Chen, J., Cheng, H.S., de Graaf, E.F., Gorczyca, J.L., Harrison, P., Hivet, G., Launay, J., Lee, W., Liu, L., Lomov, S.V., Long, A., de Luycker, E., Morestin, F., Padvoiskis, J., Peng, X.Q., Sherwood, J., Stoilova, Tz., Tao, X.M., Verpoest, I., Willems, A., Wiggers, J., Yu, T.X., and Zhu, B. (2008). "Characterization of mechanical behavior of woven fabrics: Experimental methods and benchmark results". In: *Composites Part A: Applied Science and Manufacturing* 39.6, pp. 1037–1053. ISSN: 1359-835X. DOI: https://doi.org/10.1016/j.compositesa.2008.02.016. URL: https://www.sciencedirect.com/science/article/pii/S1359835X08000572.

Carignan, Michel, Yang, Ying, Thalmann, Nadia Magnenat, and Thalmann, Daniel (July 1992). "Dressing animated synthetic actors with complex deformable clothes". In: *SIGGRAPH Comput. Graph.* 26.2, pp. 99–104. ISSN: 0097-8930. DOI: 10.1145/142920.134017. URL: https://doi.org/10.1145/142920.134017.

Casafranca, Juan J., Cirio, Gabriel, Rodríguez, Alejandro, Miguel, Eder, and Otaduy, Miguel A. (2020). "Mixing Yarns and Triangles in Cloth Simulation". In: *Computer Graphics Forum* 39.2, pp. 101–110. DOI: 10.1111/cgf.13915. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13915.

Chapman, B. M. and Hearle, J. W.S. (1972). "27—THE BENDING AND CREASING OF MULTICOMPONENT VISCOELASTIC FIBRE ASSEMBLIES. PART I: GENERAL CONSIDERATION OF THE PROBLEM". In: *The Journal of The Textile Institute* 63.7, pp. 385–403. DOI: 10.1080/00405007208630241. URL: https://doi.org/10.1080/00405007208630241.

Chapman, B.M. (1975). "The Importance of Interfiber Friction in Wrinkling". In: *Textile Research Journal* 45.12, pp. 825–829. DOI: 10.1177/004051757504501201. URL: https://doi.org/10.1177/004051757504501201.

Chaves, Eduardo WV (2013). *Notes on continuum mechanics*. Springer Science & Business Media.

Chen, Tianqi, Xu, Bing, Zhang, Chiyuan, and Guestrin, Carlos (2016). *Training Deep Nets with Sublinear Memory Cost*. DOI: 10.48550/arXiv.1604.06174. arXiv: 1604.06174 [cs.LG]. URL: https://doi.org/10.48550/arXiv.1604.06174.

Chen, Wenzheng, Gao, Jun, Ling, Huan, Smith, Edward J., Lehtinen, Jaakko, Jacobson, Alec, and Fidler, Sanja (2019). *Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer*. DOI: 10.48550/arXiv.1908.01210. arXiv: 1908.01210 [cs.CV]. URL: https://doi.org/10.48550/arXiv.1908.01210.

Choi, Kwang-Jin and Ko, Hyeong-Seok (July 2002). "Stable but responsive cloth". In: *ACM Trans. Graph.* 21.3, pp. 604–611. ISSN: 0730-0301. DOI: 10.1145/566654.566624. URL: https://doi.org/10.1145/566654.566624.

Choi, Kwang-Jin and Ko, Hyeong-Seok (2005). "Research problems in clothing simulation". In: *Computer-aided design* 37.6, pp. 585–592. ISSN: 0010-4485. DOI: 10.1016/j.cad.2004.11.002. URL: https://doi.org/10.1016/j.cad.2004.11.002.

Chu, Chauncey C., Cummings, Clinton L., and Teixeira, Newton A. (1950). "Mechanics of Elastic Performance of Textile Materials: Part V: A Study of the Factors Affecting the Drape of Fabrics—The Development of a Drape Meter". In: *Textile Research Journal* 20.8, pp. 539–548. DOI: 10.1177/004051755002000802. URL: https://doi.org/10.1177/004051755002000802.

Chung, Junyoung, Gulcehre, Caglar, Cho, KyungHyun, and Bengio, Yoshua (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. DOI: 10.48550/arXiv.1412.3555. arXiv: 1412.3555 [cs.NE]. URL: https://doi.org/10.48550/arXiv.1412.3555.

Cirio, Gabriel, Lopez-Moreno, Jorge, Miraut, David, and Otaduy, Miguel A. (Nov. 2014). "Yarn-level simulation of woven cloth". In: *ACM Trans. Graph.* 33.6. ISSN: 0730-0301. DOI: 10.1145/2661229.2661279. URL: https://doi.org/10.1145/2661229.2661279.

Cirio, Gabriel, Lopez-Moreno, Jorge, and Otaduy, Miguel A. (2015). "Efficient simulation of knitted cloth using persistent contacts". In: *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA '15. Los Angeles, California: Association for Computing Machinery, pp. 55–61. ISBN: 9781450334969. DOI: 10.1145/2786784.2786801. URL: https://doi.org/10.1145/2786784.2786801.

Cirio, Gabriel, Lopez-Moreno, Jorge, and Otaduy, Miguel A. (2017). "Yarn-Level Cloth Simulation with Sliding Persistent Contacts". In: *IEEE Transactions on Visualization and Computer Graphics* 23.2, pp. 1152–1162. ISSN: 1077-2626. DOI: 10.1109/TVCG.2016.2592908. URL: https://doi.org/10.1109/TVCG.2016.2592908.

CLOTH3D (2018). https://www.clo3d.com. Accessed: 2023-09-10.

Clyde, David, Teran, Joseph, and Tamstorf, Rasmus (2017). "Modeling and data-driven parameter estimation for woven fabrics". In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA '17. Los Angeles, California: Association for Computing Machinery. ISBN: 9781450350914. DOI: 10.1145/3099564.3099577. URL: https://doi.org/10.1145/3099564.3099577.

Collier, Billie J. (1991). "Measurement of Fabric Drape and Its Relation to Fabric Mechanical Properties and Subjective Evaluation". In: *Clothing and Textiles Research Journal* 10.1, pp. 46–52. DOI: 10.1177/0887302X9101000107. URL: https://doi.org/10.1177/0887302X9101000107.

Collier, Billie J., Paulins, Virginia A., and Collier, John R. (1989). "Effects of Interfacing Type on Shear And Drape Behavior of Apparel Fabrics". In: *Clothing and Textiles Research Journal* 7.3, pp. 51–56. DOI: 10.1177/0887302X8900700308. URL: https://doi.org/10.1177/0887302X8900700308.

Cordier, Frederic and Magnenat-Thalmann, Nadia (2005). "A Data-Driven Approach for Real-Time Clothes Simulation†". In: *Computer Graphics Forum* 24.2, pp. 173–183. DOI: https:

//doi.org/10.1111/j.1467-8659.2005.00841.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2005.00841.x.

Coumans, Erwin and Bai, Yunfei (2016). *PyBullet, a Python module for physics simulation for games, robotics and machine learning*. http://pybullet.org.

Cusick, G. E. (1965). "46—THE DEPENDENCE OF FABRIC DRAPE ON BENDING AND SHEAR STIFFNESS". In: *Journal of the Textile Institute Transactions* 56.11, T596–T606. DOI: 10.1080/19447026508662319. URL: https://doi.org/10.1080/19447026508662319.

Cusick, G. E. (1968). "21—THE MEASUREMENT OF FABRIC DRAPE". In: *The Journal of The Textile Institute* 59.6, pp. 253–260. DOI: 10.1080/00405006808659985. URL: https://www.tandfonline.com/doi/abs/10.1080/00405006808659985.

Cusick, G.E. (1961). "30—The Resistance of Fabrics to Shearing Forces". In: *Journal of the Textile Institute Transactions* 52.9, T395–T406. DOI: 10.1080/19447027.1961.10750513. URL: https://doi.org/10.1080/19447027.1961.10750513.

Cutler, Lawrence D., Gershbein, Reid, Wang, Xiaohuan Corina, Curtis, Cassidy, Maigret, Erwan, Prasso, Luca, and Farson, Peter (2005). "An art-directed wrinkle system for CG character clothing". In: *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '05. Los Angeles, California: Association for Computing Machinery, pp. 117–125. ISBN: 1595931988. DOI: 10.1145/1073368.1073384. URL: https://doi.org/10.1145/1073368.1073384.

Dahl, Phil R (1968). "A solid friction model". In: *The Aerospace Corporation* 18.1, pp. 1–24. URL: https://apps.dtic.mil/sti/tr/pdf/ADA041920.pdf.

Daviet, Gilles, Bertails-Descoubes, Florence, and Boissieux, Laurence (Dec. 2011). "A hybrid iterative solver for robustly capturing coulomb friction in hair dynamics". In: *ACM Trans. Graph.* 30.6, pp. 1–12. ISSN: 0730-0301. DOI: 10.1145/2070781.2024173. URL: https://doi.org/10.1145/2070781.2024173.

De Aguiar, Edilson, Sigal, Leonid, Treuille, Adrien, and Hodgins, Jessica K (July 2010). "Stable spaces for real-time clothing". In: *ACM Trans. Graph.* 29.4. ISSN: 0730-0301. DOI: 10.1145/1778765.1778843. URL: https://doi.org/10.1145/1778765.1778843.

De Wit, C Canudas, Olsson, Henrik, Astrom, Karl Johan, and Lischinsky, Pablo (1995). "A new model for control of systems with friction". In: *IEEE Transactions on Automatic Control* 40.3, pp. 419–425. DOI: 10.1109/9.376053. URL: https://doi.org/10.1109/9.376053.

Degrave, Jonas, Hermans, Michiel, Dambre, Joni, and wyffels, Francis (2019). "A Differentiable Physics Engine for Deep Learning in Robotics". In: *Frontiers in Neurorobotics* 13. ISSN: 1662-5218. DOI: 10.3389/fnbot.2019.00006. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2019.00006.

Delavari, Koorosh and Dabiryan, Hadi (2021). "Mathematical and numerical simulation of geometry and mechanical behavior of sandwich composites reinforced with 1× 1-Rib-Gaiting weft-knitted spacer fabric; compressional behavior". In: *Composite Structures* 268, p. 113952. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2021.113952. URL: https://www.sciencedirect.com/science/article/pii/S0263822321004128.

Desbrun, Mathieu, Schröder, Peter, and Barr, Alan (1999). "Interactive animation of structured deformable objects". In: *Proceedings of the 1999 Conference on Graphics Interface '99*. Kingston, Ontario, Canada: Morgan Kaufmann Publishers Inc., pp. 1–8. ISBN: 1558606327. DOI: 10.5555/351631.351638. URL: http://www.multires.caltech.edu/pubs/GI99.pdf.

Deul, Crispin, Charrier, Patrick, and Bender, Jan (2016). "Position-based rigid-body dynamics". In: *Computer Animation and Virtual Worlds* 27.2, pp. 103–112. DOI: https://doi.org/10.1002/cav.1614. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.1614.

Donahue, Jeff, Hendricks, Lisa Anne, Rohrbach, Marcus, Venugopalan, Subhashini, Guadarrama, Sergio, Saenko, Kate, and Darrell, Trevor (2017). "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4, pp. 677–691. DOI: 10.1109/TPAMI.2016.2599174. URL: https://doi.org/10.1109/TPAMI.2016.2599174.

Eberhardt, B., Weber, A., and Strasser, W. (1996). "A fast, flexible, particle-system model for cloth draping". In: *IEEE Computer Graphics and Applications* 16.5, pp. 52–59. ISSN: 0272-1716. DOI: 10.1109/38.536275. URL: https://doi.org/10.1109/38.536275.

Eischen, Jeffrey W, Deng, Shigan, and Clapp, Timothy G (1996). "Finite-element modeling and control of flexible fabric parts". In: *IEEE Computer Graphics and Applications* 16.5, pp. 71–80. ISSN: 0272-1716. DOI: 10.1109/38.536277. URL: https://doi.org/10.1109/38.536277.

El Messiry, Magdi and El-Tarfawy, Shaimaa (2020). "Investigation of fabric drape-flexural rigidity relation: modified fabric drape coefficient". In: *The Journal of The Textile Institute* 111.3, pp. 416–423. DOI: 10.1080/00405000.2019.1634976. URL: https://doi.org/10.1080/00405000.2019.1634976.

English, Elliot and Bridson, Robert (Aug. 2008). "Animating developable surfaces using nonconforming elements". In: *ACM Trans. Graph.* 27.3, pp. 1–5. ISSN: 0730-0301. DOI: 10.1145/1360612.1360665. URL: https://doi.org/10.1145/1360612.1360665.

Eriksson, David and Jankowiak, Martin (2021). *High-Dimensional Bayesian Optimization with Sparse Axis-Aligned Subspaces*. DOI: 10.48550/arXiv.2103.00349. arXiv: 2103.00349 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2103.00349.

Fan, Jintu and Hunter, Lawrance (2009). *Engineering apparel fabrics and garments*. Woodhead Publishing in textiles ; no. 96. Cambridge, U.K: Woodhead Publishing in association with the Textile Institute. ISBN: 978-1-84569-134-9.

Feng, Wei-Wen, Yu, Yizhou, and Kim, Byung-Uck (July 2010). "A deformation transformer for real-time cloth animation". In: *ACM Trans. Graph.* 29.4. ISSN: 0730-0301. DOI: 10.1145/1778765.1778845. URL: https://doi.org/10.1145/1778765.1778845.

Feng, Xudong, Huang, Wenchao, Xu, Weiwei, and Wang, Huamin (Nov. 2022). "Learning-Based Bending Stiffness Parameter Estimation by a Drape Tester". In: *ACM Trans. Graph.* 41.6. ISSN: 0730-0301. DOI: 10.1145/3550454.3555464. URL: https://doi.org/10.1145/3550454.3555464.

Feynman, Carl Richard (1986). "Modeling the appearance of cloth". PhD thesis. Massachusetts Institute of Technology. URL: http://hdl.handle.net/1721.1/14924.

Frazier, Peter I. (2018). *A Tutorial on Bayesian Optimization*. DOI: 10.48550/arXiv.1807.02811. arXiv: 1807.02811 [stat.ML]. URL: https://doi.org/10.48550/arXiv.1807.02811.

Al-Gaadi, Bidour, Göktepe, Fatma, and Halász, Marianna (2012). "A new method in fabric drape measurement and analysis of the drape formation process". In: *Textile Research Journal* 82.5, pp. 502–512. DOI: 10.1177/0040517511420760. URL: https://doi.org/10.1177/0040517511420760.

Gao, Ziyue and Chen, Li (2021). "A review of multi-scale numerical modeling of three-dimensional woven fabric". In: *Composite Structures* 263, p. 113685. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2021.113685. URL: https://www.sciencedirect.com/science/article/pii/S026382232100146X.

Ghith, Adel, Hamdi, Thouraya, and Fayala, Faten (2015). "Prediction of Drape Coefficient by Artificial Neural Network". In: *Autex Research Journal* 15.4, pp. 266–274. DOI: doi:10.1515/aut-2015-0045. URL: https://doi.org/10.1515/aut-2015-0045.

Goldenthal, Rony, Harmon, David, Fattal, Raanan, Bercovier, Michel, and Grinspun, Eitan (July 2007). "Efficient simulation of inextensible cloth". In: *ACM Trans. Graph.* 26.3, 49–es. ISSN: 0730-0301. DOI: 10.1145/1276377.1276438. URL: https://doi.org/10.1145/1276377.1276438.

Gong, Deshan, Zhu, Zhanxing, Bulpitt, Andrew J., and Wang, He (2022). *Fine-grained differentiable physics: a yarn-level model for fabrics*. DOI: 10.48550/arXiv.2202.00504. arXiv: 2202.00504 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2202.00504.

Grinspun, Eitan (2008). "A Discrete Model of Thin Shells". In: *Discrete Differential Geometry*. Ed. by Alexander I. Bobenko, John M. Sullivan, Peter Schröder, and Günter M. Ziegler. Basel: Birkhäuser Basel, pp. 325–337. ISBN: 978-3-7643-8621-4. DOI: 10.1007/978-3-7643-8621-4_17. URL: https://doi.org/10.1007/978-3-7643-8621-4_17.

Grinspun, Eitan, Hirani, Anil N., Desbrun, Mathieu, and Schröder, Peter (2003). "Discrete shells". In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '03. San Diego, California: Eurographics Association, pp. 62–67. ISBN: 1581136595. DOI: 10.5555/846276.846284. URL: https://dl.acm.org/doi/10.5555/846276.846284.

Guan, Peng, Reiss, Loretta, Hirshberg, David A., Weiss, Alexander, and Black, Michael J. (July 2012). "DRAPE: DRessing Any PErson". In: *ACM Trans. Graph.* 31.4. ISSN: 0730-0301. DOI: 10.1145/2185520.2185531. URL: https://doi.org/10.1145/2185520.2185531.

Guennebaud, Gaël, Jacob, Benoît, et al. (2010). *Eigen v3*. http://eigen.tuxfamily.org.

Gundogdu, Erhan, Constantin, Victor, Parashar, Shaifali, Seifoddini, Amrollah, Dang, Minh, Salzmann, Mathieu, and Fua, Pascal (Jan. 2022). "GarNet++: Improving Fast and Accurate Static 3D Cloth Draping by Curvature Loss". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.01, pp. 181–195. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.3010886. URL: https://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.3010886.

Gundogdu, Erhan, Constantin, Victor, Seifoddini, Amrollah, Dang, Minh, Salzmann, Mathieu, and Fua, Pascal (Nov. 2019). "GarNet: A Two-Stream Network for Fast and Accurate 3D Cloth Draping". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 8738–8747. DOI: 10.1109/ICCV.2019.00883. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00883.

Guo, Qi, Han, Xuchen, Fu, Chuyuan, Gast, Theodore, Tamstorf, Rasmus, and Teran, Joseph (July 2018). "A material point method for thin shells with frictional contact". In: *ACM Trans. Graph.* 37.4. ISSN: 0730-0301. DOI: 10.1145/3197517.3201346. URL: https://doi.org/10.1145/3197517.3201346.

Hadap, S., Bongarter, E., Volino, P., and Magnenat-Thalmann, N. (1999). "Animating wrinkles on clothes". In: *Proceedings Visualization '99 (Cat. No.99CB37067)*, pp. 175–523. DOI: 10.1109/VISUAL.1999.809885. URL: https://doi.org/10.1109/VISUAL.1999.809885.

Han, Jiequn, Jentzen, Arnulf, and Weinan, E (2018). "Solving high-dimensional partial differential equations using deep learning". In: *Proceedings of the National Academy of Sciences* 115.34, pp. 8505–8510. DOI: 10.1073/pnas.1718942115. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1718942115.

Harmon, David, Vouga, Etienne, Tamstorf, Rasmus, and Grinspun, Eitan (Aug. 2008). "Robust treatment of simultaneous collisions". In: *ACM Trans. Graph.* 27.3, pp. 1–4. ISSN: 0730-0301. DOI: 10.1145/1360612.1360622. URL: https://doi.org/10.1145/1360612.1360622.

Hauth, Michael, Etzmuß, Olaf, and Straßer, Wolfgang (2003). "Analysis of numerical methods for the simulation of deformable models". In: *The Visual Computer* 19, pp. 581–600. ISSN: 1432-2315. DOI: 10.1007/s00371-003-0206-2. URL: https://doi.org/10.1007/s00371-003-0206-2.

Hecht, F. (2012). "New development in FreeFem++". In: *J. Numer. Math.* 20.3-4, pp. 251–265. ISSN: 1570-2820. URL: https://freefem.org/.

Heiden, Eric, Millard, David, Zhang, Hejia, and Sukhatme, Gaurav S. (May 2020). "Interactive Differentiable Simulation". In: *arXiv:1905.10706 [cs, stat]*. arXiv: 1905.10706. URL: http://arxiv.org/abs/1905.10706 (visited on 04/22/2021).

Hinds, BK and McCartney, J (1990). "Interactive garment design". In: *The Visual Computer* 6, pp. 53–61. ISSN: 1432-2315. DOI: 10.1007/BF01901066. URL: https://doi.org/10.1007/BF01901066.

Hoffman, Matthew D, Blei, David M, Wang, Chong, and Paisley, John (2013). *Stochastic Variational Inference*. DOI: 10.48550/arXiv.1206.7051. arXiv: 1206.7051 [stat.ML]. URL: https://doi.org/10.48550/arXiv.1206.7051.

Holl, Philipp, Koltun, Vladlen, Um, Kiwon, and Thuerey, Nils (2020). "phiflow: A differentiable pde solving framework for deep learning via physical simulations". In: *NeurIPS Workshop*. URL: https://montrealrobotics.ca/diffcvgp/assets/papers/3.pdf.

Hu, Jinlian. (2008). *Fabric testing*. eng. Woodhead publishing in textiles ; no. 76. Boca Raton, Fla: CRC Press. ISBN: 9781845692971. URL: https://www.sciencedirect.com/book/9781845692971/fabric-testing.

Hu, Xinrong, Bai, Yan, Cui, Shuqin, Du, Xiaoqin, and Deng, Zhongmin (2009). "Review of cloth modeling". In: *2009 ISECS International Colloquium on Computing, Communication, Control, and Management*. Vol. 4, pp. 338–341. DOI: 10.1109/CCCM.2009.5267684. URL: https://doi.org/10.1109/CCCM.2009.5267684.

Hu, Yuanming, Anderson, Luke, Li, Tzu-Mao, Sun, Qi, Carr, Nathan, Ragan-Kelley, Jonathan, and Durand, Frédo (2020). "DiffTaichi: Differentiable Programming for Physical Simulation".

In: *arXiv preprint arXiv:1910.00935*. DOI: 10.48550/arXiv.1910.00935. URL: https://doi.org/10.48550/arXiv.1910.00935.

Hu, Yuanming, Liu, Jiancheng, Spielberg, Andrew, Tenenbaum, Joshua B., Freeman, William T., Wu, Jiajun, Rus, Daniela, and Matusik, Wojciech (2019). "ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics". In: *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, QC, Canada: IEEE Press, pp. 6265–6271. DOI: 10.1109/ICRA.2019.8794333. URL: https://doi.org/10.1109/ICRA.2019.8794333.

Huang, Zhiao, Hu, Yuanming, Du, Tao, Zhou, Siyuan, Su, Hao, Tenenbaum, Joshua B., and Gan, Chuang (2021). *PlasticineLab: A Soft-Body Manipulation Benchmark with Differentiable Physics*. DOI: 10.48550/arXiv.2104.03311. arXiv: 2104.03311 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2104.03311.

James, Doug L. and Fatahalian, Kayvon (July 2003a). "Precomputing Interactive Dynamic Deformable Scenes". In: *ACM Trans. Graph.* 22.3, pp. 879–887. ISSN: 0730-0301. DOI: 10.1145/882262.882359. URL: https://doi.org/10.1145/882262.882359.

James, Doug L. and Fatahalian, Kayvon (July 2003b). "Precomputing interactive dynamic deformable scenes". In: *ACM Trans. Graph.* 22.3, pp. 879–887. ISSN: 0730-0301. DOI: 10.1145/882262.882359. URL: https://doi.org/10.1145/882262.882359.

Jatavallabhula, Krishna Murthy, Macklin, Miles, Golemo, Florian, Voleti, Vikram, Petrini, Linda, Weiss, Martin, Considine, Breandan, Parent-Levesque, Jerome, Xie, Kevin, Erleben, Kenny, Paull, Liam, Shkurti, Florian, Nowrouzezahrai, Derek, and Fidler, Sanja (2021). "gradSim: Differentiable simulation for system identification and visuomotor control". In: *arXiv preprint*. DOI: 10.48550/arXiv.2104.02646. arXiv: 2104.02646 [cs.CV]. URL: https://doi.org/10.48550/arXiv.2104.02646.

Jatavallabhula, Krishna Murthy, Smith, Edward, Lafleche, Jean-Francois, Tsang, Clement Fuji, Rozantsev, Artem, Chen, Wenzheng, Xiang, Tommy, Lebaredian, Rev, and Fidler, Sanja (2019). "Kaolin: A pytorch library for accelerating 3d deep learning research". In: *arXiv preprint arXiv:1911.05063*. DOI: 10.48550/arXiv.1911.05063. URL: https://doi.org/10.48550/arXiv.1911.05063.

Jawed, M Khalid, Novelia, Alyssa, and O'Reilly, Oliver M (2018). *A primer on the kinematics of discrete elastic rods*. Springer. DOI: 10.1007/978-3-319-76965-3. URL: https://doi.org/10.1007/978-3-319-76965-3.

Jeong, Y. J. (1998). "A Study of Fabric-drape Behaviour with Image Analysis Part I: Measurement, Characterisation, and Instability". In: *The Journal of The Textile Institute* 89.1, pp. 59–69. DOI: 10.1080/00405009808658597. URL: https://doi.org/10.1080/00405009808658597.

Jeong, Y. J. and D. G., Phillips (1998). "A Study of Fabric-drape Behaviour with Image Analysis. Part II: The Effects of Fabric Structure and Mechanical Properties on Fabric Drape". In: *The Journal of The Textile Institute* 89.1, pp. 70–79. DOI: 10.1080/00405009808658598. URL: https://doi.org/10.1080/00405009808658598.

Johannes, V. I., Green, M. A., and Brockley, C. A. (June 1973). "The role of the rate of application of the tangential force in determining the static friction coefficient". en. In: *Wear* 24.3, pp. 381–385. ISSN: 0043-1648. DOI: 10.1016/0043-1648(73)90166-X. URL: https://www.sciencedirect.com/science/article/pii/004316487390166X (visited on 04/22/2021).

Jojic, Nebojsa and Huang, Thomas S. (Jan. 1997). "Estimating Cloth Draping Parameters from Range Data". In: *International Workshop on Synthetic-Natural Hybrid Coding and 3-D Imaging*. URL: https://citeseerx.ist.psu.edu/pdf/25e99a6c0ecac9f40f428b44bec4023015e75e25.

Jones, Donald R, Schonlau, Matthias, and Welch, William J (1998). "Efficient global optimization of expensive black-box functions". In: *Journal of Global optimization* 13.4, pp. 455–492. ISSN: 1573-2916. DOI: 10.1023/A:1008306431147. URL: https://doi.org/10.1023/A:1008306431147.

Ju, Eunjung and Choi, Myung Geol (2020). "Estimating Cloth Simulation Parameters From a Static Drape Using Neural Networks". In: *IEEE Access* 8, pp. 195113–195121. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3033765. URL: https://doi.org/10.1109/ACCESS.2020.3033765.

Ju, Eunjung, Kim, Kwang-yun, Lee, Jaehoon, Yoon, Sungjin, and Choi, Myung Geol (2022). "Interactive exploration of drapes by simulation parameters". In: *Computer Animation and*

Virtual Worlds 33.3-4, e2058. DOI: 10.1002/cav.2058. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.2058.

Jung, Il-Hoe, Lee, Sang-Bin, Kim, Jong-Jun, Ryu, Han-Na, and Ko, Hyeong-Seok (2016). "Modeling the non-elastic stretch deformation of cloth based on creep analysis". In: *Textile Research Journal* 86.3, pp. 245–255. DOI: 10.1177/0040517515588268. URL: https://doi.org/10.1177/0040517515588268.

Kaldor, Jonathan M., James, Doug L., and Marschner, Steve (2008). "Simulating knitted cloth at the yarn level". In: *ACM SIGGRAPH 2008 Papers*. SIGGRAPH '08. Los Angeles, California: Association for Computing Machinery. ISBN: 9781450301121. DOI: 10.1145/1399504.1360664. URL: https://doi.org/10.1145/1399504.1360664.

Kaldor, Jonathan M., James, Doug L., and Marschner, Steve (July 2010). "Efficient yarn-based cloth with adaptive contact linearization". In: *ACM Trans. Graph.* 29.4. ISSN: 0730-0301. DOI: 10.1145/1778765.1778842. URL: https://doi.org/10.1145/1778765.1778842.

Kang, Young-Min, Choi, Jeong-Hyeon, Cho, Hwan-Gue, and Lee, Do-Hoon (2001). "An efficient animation of wrinkled cloth with approximate implicit integration". In: *The Visual Computer* 17, pp. 147–157. ISSN: 1432-231. DOI: 10.1007/s003710100103. URL: https://doi.org/10.1007/s003710100103.

Karniadakis, George Em, Kevrekidis, Ioannis G., Lu, Lu, Perdikaris, Paris, Wang, Sifan, and Yang, Liu (2021). "Physics-informed machine learning". In: *Nature Reviews Physics* 3, pp. 422–440. ISSN: 2522-5820. DOI: 10.1038/s42254-021-00314-5. URL: https://doi.org/10.1038/s42254-021-00314-5.

Kavan, Ladislav, Gerszewski, Dan, Bargteil, Adam W., and Sloan, Peter-Pike (2011). "Physics-inspired upsampling for cloth simulation in games". In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH '11. Vancouver, British Columbia, Canada: Association for Computing Machinery. ISBN: 9781450309431. DOI: 10.1145/1964921.1964988. URL: https://doi.org/10.1145/1964921.1964988.

Kawabata, Sueo (1980). "The Standardization and Analysis of Hand Evaluation". In: *The Textile Machinery Society Japan*.

Kenkare, Narahari, Lamar, Traci A. M., Pandurangan, Pradeep, and Eischen, Jeffrey (2008). "Enhancing accuracy of drape simulation. Part I: Investigation of drape variability via 3D scanning". In: *The Journal of The Textile Institute* 99.3, pp. 211–218. DOI: 10.1080/00405000701489222. URL: https://doi.org/10.1080/00405000701489222.

Kim, Byung-Cheol, Oh, Seungwoo, and Wohn, Kwangyun (Jan. 2011). "Persistent Wrinkles and Folds of Clothes". In: *International Journal of Virtual Reality* 10.1, pp. 61–66. DOI: 10.20870/IJVR.2011.10.1.2803. URL: https://ijvr.eu/article/view/2803.

Kim, Doyub, Koh, Woojong, Narain, Rahul, Fatahalian, Kayvon, Treuille, Adrien, and O'Brien, James F. (July 2013). "Near-exhaustive precomputation of secondary cloth effects". In: *ACM Trans. Graph.* 32.4. ISSN: 0730-0301. DOI: 10.1145/2461912.2462020. URL: https://doi.org/10.1145/2461912.2462020.

Kim, Sungmin (2011). "Determination of fabric physical properties for the simulation of Cusick drapemeter". In: *Fibers and Polymers* 12.1, pp. 132–136. ISSN: 1875-0052. DOI: 10.1007/s12221-011-0132-2. URL: https://doi.org/10.1007/s12221-011-0132-2.

Kim, Tae-Yong and Vendrovsky, Eugene (2008). "DrivenShape: a data-driven approach for shape deformation". In: *ACM SIGGRAPH 2008 Talks*. SIGGRAPH '08. Los Angeles, California: Association for Computing Machinery. ISBN: 9781605583433. DOI: 10.1145/1401032.1401121. URL: https://doi.org/10.1145/1401032.1401121.

King, Michael James, Jearanaisilawong, Petch, and Socrate, Simona (2005). "A continuum constitutive model for the mechanical behavior of woven fabrics". In: *International Journal of Solids and Structures* 42.13, pp. 3867–3896. ISSN: 0020-7683. DOI: 10.1016/j.ijsolstr.2004.10.030. URL: https://www.sciencedirect.com/science/article/pii/S0020768304006225.

Kingma, Diederik P and Welling, Max (2022). *Auto-Encoding Variational Bayes*. DOI: 10.48550/arXiv.1312.6114. arXiv: 1312.6114 [stat.ML]. URL: https://doi.org/10.48550/arXiv.1312.6114.

Kuijpers, Sandra, LuibleTBär, Christiane, and Gong, Hugh (2020). "THE MEASUREMENT OF FABRIC PROPERTIES FOR VIRTUAL SIMULATION—A CRITICAL REVIEW". En-

glish. In: *IEEE Standards Association, Industry Connections Report*, pp. 1–43. URL: https://standards.ieee.org/wp-content/uploads/import/governance/iccom/3DBP-Measurement_Fabric_Properties-Virtual_Simulation.pdf.

Kunii, Tosiyasu L. and Gotoda, Hironobu (1990). "Modeling and Animation of Garment Wrinkle Formation Processes". In: *Computer Animation '90*. Ed. by Nadia Magnenat-Thalmann and Daniel Thalmann. Tokyo: Springer Japan, pp. 131–147. ISBN: 978-4-431-68296-7.

Lahey, Timothy (2002). "Modelling hysteresis in the bending of fabrics". MA thesis. University of Waterloo. URL: http://hdl.handle.net/10012/941.

Lähner, Zorah, Cremers, Daniel, and Tung, Tony (2018). "DeepWrinkles: Accurate and Realistic Clothing Modeling". In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss. Cham: Springer International Publishing, pp. 698–715. ISBN: 978-3-030-01225-0.

Larboulette, C. and Cani, M.-P. (2004). "Real-time dynamic wrinkles". In: *Proceedings Computer Graphics International, 2004.* Pp. 522–525. DOI: 10.1109/CGI.2004.1309258. URL: https://doi.org/10.1109/CGI.2004.1309258.

Leaf, Jonathan, Wu, Rundong, Schweickart, Eston, James, Doug L., and Marschner, Steve (Dec. 2018). "Interactive design of periodic yarn-level cloth patterns". In: *ACM Trans. Graph.* 37.6. ISSN: 0730-0301. DOI: 10.1145/3272127.3275105. URL: https://doi.org/10.1145/3272127.3275105.

Letham, Ben, Calandra, Roberto, Rai, Akshara, and Bakshy, Eytan (2020). "Re-examining linear embeddings for high-dimensional Bayesian optimization". In: *Advances in neural information processing systems* 33, pp. 1546–1558. DOI: https://doi.org/10.48550/arXiv.2001.11659. URL: 10.48550/arXiv.2001.11659.

Levison, R., C. Bostwick, B. Behre, and Kärrholm, M. (1962). "SOME FUNDAMENTAL THEORETICAL AND EXPERIMENTAL ASPECTS OF FABRIC CREASING". In: *Journal of the Textile Institute Proceedings* 53.1, P116–P134. DOI: 10.1080/19447016208688649. URL: https://doi.org/10.1080/19447016208688649.

Li, Cheng, Tang, Min, Tong, Ruofeng, Cai, Ming, Zhao, Jieyi, and Manocha, Dinesh (Nov. 2020). "P-cloth: interactive complex cloth simulation on multi-GPU systems using dynamic matrix assembly and pipelined implicit integrators". In: *ACM Trans. Graph.* 39.6. ISSN: 0730-0301. DOI: 10.1145/3414685.3417763. URL: https://doi.org/10.1145/3414685.3417763.

Li, Jie, Daviet, Gilles, Narain, Rahul, Bertails-Descoubes, Florence, Overby, Matthew, Brown, George E., and Boissieux, Laurence (July 2018). "An implicit frictional contact solver for adaptive cloth simulation". In: *ACM Trans. Graph.* 37.4. ISSN: 0730-0301. DOI: 10.1145/3197517.3201308. URL: https://doi.org/10.1145/3197517.3201308.

Li, Yifei, Du, Tao, Wu, Kui, Xu, Jie, and Matusik, Wojciech (Oct. 2022a). "DiffCloth: Differentiable Cloth Simulation with Dry Frictional Contact". In: *ACM Trans. Graph.* 42.1. ISSN: 0730-0301. DOI: 10.1145/3527660. URL: https://doi.org/10.1145/3527660.

Li, Yudi, Tang, Min, Yang, Yun, Huang, Zi, Tong, Ruofeng, Yang, Shuangcai, Li, Yao, and Manocha, Dinesh (2022b). "N-Cloth: Predicting 3D Cloth Deformation with Mesh-Based Networks". In: *Computer Graphics Forum* 41.2, pp. 547–558. DOI: https://doi.org/10.1111/cgf.14493. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14493.

Li, Yunzhu, Wu, Jiajun, Tedrake, Russ, Tenenbaum, Joshua B., and Torralba, Antonio (Apr. 2019). "Learning Particle Dynamics for Manipulating Rigid Bodies, Deformable Objects, and Fluids". In: *arXiv:1810.01566 [physics, stat]*. arXiv: 1810.01566. URL: http://arxiv.org/abs/1810.01566 (visited on 04/02/2021).

Liang, Junbang, Lin, Ming C., and Koltun, Vladlen (2019). "Differentiable cloth simulation for inverse problems". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc. DOI: 10.5555/3454287.3454357. URL: https://dl.acm.org/doi/10.5555/3454287.3454357.

Libai, Avinoam and Simmonds, James G. (1983). "Nonlinear Elastic Shell Theory". In: ed. by John W. Hutchinson and Theodore Y. Wu. Vol. 23. Advances in Applied Mechanics. Elsevier, pp. 271–371. DOI: 10.1016/S0065-2156(08)70245-X. URL: https://www.sciencedirect.com/science/article/pii/S006521560870245X.

Lin, Hua, Clifford, Mike J., Long, Andrew C., Lee, Ken, and Guo, Ning (2012). "A finite element approach to the modelling of fabric mechanics and its application to virtual fabric design and testing". In: *The Journal of The Textile Institute* 103.10, pp. 1063–1076. DOI: 10.1080/00405000.2012.660755. URL: https://doi.org/10.1080/00405000.2012.660755.

Liu, Ce (2009). "Beyond pixels: exploring new representations and applications for motion analysis". AAI0822221. PhD thesis. USA. URL: https://people.csail.mit.edu/celiu/Thesis/CePhDThesis.pdf.

Liu, Tiantian, Bargteil, Adam W., O'Brien, James F., and Kavan, Ladislav (Nov. 2013). "Fast simulation of mass-spring systems". In: *ACM Trans. Graph.* 32.6. ISSN: 0730-0301. DOI: 10.1145/2508363.2508406. URL: https://doi.org/10.1145/2508363.2508406.

Logan, Daryl L (2022). *First Course in the Finite Element Method, Enhanced Edition, SI Version*. Cengage Learning.

Lojen, Darja Zunic and Jevsnik, Simona (2007). "Some aspects of fabric drape". In: *Fibres and Textiles in Eastern Europe* 15.4, p. 39. URL: http://fibtex.lodz.pl/article407.html.

Long, James, Burns, Katherine, and Yang, Jingzhou (James) (2011). "Cloth Modeling and Simulation: A Literature Survey". In: *Digital Human Modeling*. Ed. by Vincent G. Duffy. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 312–320. ISBN: 978-3-642-21799-9. DOI: 10.1007/978-3-642-21799-9_35. URL: https://doi.org/10.1007/978-3-642-21799-9_35.

Loock, Achim, Schömer, Elmar, and Stadtwald, Im (2001). "A virtual environment for interactive assembly simulation: From rigid bodies to deformable cables". In: *5th World Multiconference on Systemics, Cybernetics and Informatics (SCI'01)*. Vol. 3. Citeseer, pp. 325–332. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ba06f4cfc9f896c4669bec95ed062753a4fa045e.

Lu, Lu, Meng, Xuhui, Mao, Zhiping, and Karniadakis, George Em (2021). "DeepXDE: A Deep Learning Library for Solving Differential Equations". In: *SIAM Review* 63.1, pp. 208–228. DOI: 10.1137/19M1274067. URL: https://doi.org/10.1137/19M1274067.

Luible, Christiane and Magnenat-Thalmann, Nadia (2008). "The simulation of cloth using accurate physical parameters". In: *Proceedings of the Tenth IASTED International Conference on Computer Graphics and Imaging*. CGIM '08. Innsbruck, Austria: ACTA Press, pp. 123–128. ISBN: 9780889867208. DOI: 10.5555/1722302.1722328. URL: https://dl.acm.org/doi/10.5555/1722302.1722328.

Ly, Mickaël, Jouve, Jean, Boissieux, Laurence, and Bertails-Descoubes, Florence (Aug. 2020). "Projective dynamics with dry frictional contact". In: *ACM Trans. Graph.* 39.4. ISSN: 0730-0301. DOI: 10.1145/3386569.3392396. URL: https://doi.org/10.1145/3386569.3392396.

Macklin, Miles and Müller, Matthias (July 2013). "Position based fluids". In: *ACM Trans. Graph.* 32.4. ISSN: 0730-0301. DOI: 10.1145/2461912.2461984. URL: https://doi.org/10.1145/2461912.2461984.

Macklin, Miles, Müller, Matthias, and Chentanez, Nuttapong (2016). "XPBD: position-based simulation of compliant constrained dynamics". In: *Proceedings of the 9th International Conference on Motion in Games*. MIG '16. Burlingame, California: Association for Computing Machinery, pp. 49–54. ISBN: 9781450345927. DOI: 10.1145/2994258.2994272. URL: https://doi.org/10.1145/2994258.2994272.

Macklin, Miles, Müller, Matthias, Chentanez, Nuttapong, and Kim, Tae-Yong (July 2014). "Unified particle physics for real-time applications". In: *ACM Trans. Graph.* 33.4. ISSN: 0730-0301. DOI: 10.1145/2601097.2601152. URL: https://doi.org/10.1145/2601097.2601152.

Magnus, Jan R and Neudecker, Heinz (2019). *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons. ISBN: 9781119541202. DOI: 10.1002/9781119541219. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/9781119541219.

Marchesi, Maria Chiara, Vangosa, Francesco Briatico, and Pavan, Andrea (2012). "A new test and apparatus for characterizing the time-dependent recovery of polymeric fibers and yarns from bending". In: *Textile Research Journal* 82.5, pp. 463–473. DOI: 10.1177/0040517511427964. URL: https://doi.org/10.1177/0040517511427964.

Marschner, Steve and Shirley, Peter (2015). *Fundamentals of Computer Graphics*. CRC Press.

Martin, Sebastian, Thomaszewski, Bernhard, Grinspun, Eitan, and Gross, Markus (2011). "Example-based elastic materials". In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH '11. Vancouver, British Columbia, Canada: Association for Computing Machinery. ISBN: 9781450309431. DOI: 10.1145/1964921.1964967. URL: https://doi.org/10.1145/1964921.1964967.

McCormick, Stephen F (1987). *Multigrid methods*. SIAM.

Meyer, Mark, Debunne, Gilles, Desbrun, Mathieu, and Barr, Alan H. (2001). "Interactive animation of cloth-like objects in virtual reality". In: *The Journal of Visualization and Computer Animation* 12.1, pp. 1–12. DOI: https://doi.org/10.1002/vis.244. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/vis.244.

Miguel, E., Bradley, D., Thomaszewski, B., Bickel, B., Matusik, W., Otaduy, M. A., and Marschner, S. (May 2012). "Data-Driven Estimation of Cloth Simulation Models". In: *Comput. Graph. Forum* 31.2pt2, pp. 519–528. ISSN: 0167-7055. DOI: 10.5555/2318896.2318912. URL: https://dl.acm.org/doi/10.5555/2318896.2318912.

Miguel, Eder, Tamstorf, Rasmus, Bradley, Derek, Schvartzman, Sara C., Thomaszewski, Bernhard, Bickel, Bernd, Matusik, Wojciech, Marschner, Steve, and Otaduy, Miguel A. (Nov. 2013). "Modeling and estimation of internal friction in cloth". In: *ACM Trans. Graph.* 32.6. ISSN: 0730-0301. DOI: 10.1145/2508363.2508389. URL: https://doi.org/10.1145/2508363.2508389.

Minazio, Pier Giorgio (1995). "FAST–fabric assurance by simple testing". In: *International Journal of Clothing Science and Technology*. ISSN: 0955-6222. DOI: 10.1108/09556229510087146. URL: https://doi.org/10.1108/09556229510087146.

Mohammed, U., Lekakou, C., Dong, L., and Bader, M.G. (2000). "Shear deformation and micromechanics of woven fabrics". In: *Composites Part A: Applied Science and Manufacturing* 31.4, pp. 299–308. ISSN: 1359-835X. DOI: https://doi.org/10.1016/S1359-835X(99)00081-0. URL: https://www.sciencedirect.com/science/article/pii/S1359835X99000810.

Montazeri, Zahra, Xiao, Chang, Fei, Yun, Zheng, Changxi, and Zhao, Shuang (Jan. 2021). "Mechanics-Aware Modeling of Cloth Appearance". In: *IEEE Transactions on Visualization*

and Computer Graphics 27.1, pp. 137–150. ISSN: 1077-2626. DOI: 10.1109/TVCG.2019.2937301. URL: https://doi.org/10.1109/TVCG.2019.2937301.

Moore, Matthew and Wilhelms, Jane (June 1988). "Collision Detection and Response for Computer Animation". In: *SIGGRAPH Comput. Graph.* 22.4, pp. 289–298. ISSN: 0097-8930. DOI: 10.1145/378456.378528. URL: https://doi.org/10.1145/378456.378528.

Müller, Matthias (2008). "Hierarchical Position Based Dynamics". In: *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2008)*. Ed. by Francois Faure and Matthias Teschner. The Eurographics Association. ISBN: 978-3-905673-70-8. DOI: 10.2312/PE/vriphys/vriphys08/001-010. URL: http://dx.doi.org/10.2312/PE/vriphys/vriphys08/001-010.

Müller, Matthias and Chentanez, Nuttapong (2010). "Wrinkle meshes". In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '10. Madrid, Spain: Eurographics Association, pp. 85–92. DOI: 10.5555/1921427.1921441. URL: https://dl.acm.org/doi/abs/10.5555/1921427.1921441.

Müller, Matthias, Chentanez, Nuttapong, Kim, Tae-Yong, and Macklin, Miles (2015). "Strain based dynamics". In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '14. Copenhagen, Denmark: Eurographics Association, pp. 149–157.

Müller, Matthias, Heidelberger, Bruno, Hennix, Marcus, and Ratcliff, John (2007). "Position based dynamics". In: *Journal of Visual Communication and Image Representation* 18.2, pp. 109–118. ISSN: 1047-3203. DOI: https://doi.org/10.1016/j.jvcir.2007.01.005. URL: https://www.sciencedirect.com/science/article/pii/S1047320307000065.

Narain, Rahul, Pfaff, Tobias, and O'Brien, James F. (July 2013). "Folding and crumpling adaptive sheets". In: *ACM Trans. Graph.* 32.4. ISSN: 0730-0301. DOI: 10.1145/2461912.2462010. URL: https://doi.org/10.1145/2461912.2462010.

Narain, Rahul, Samii, Armin, and O'Brien, James F. (Nov. 2012). "Adaptive anisotropic remeshing for cloth simulation". In: *ACM Trans. Graph.* 31.6. ISSN: 0730-0301. DOI: 10.1145/2366145.2366171. URL: https://doi.org/10.1145/2366145.2366171.

Nayebi, Amin, Munteanu, Alexander, and Poloczek, Matthias (Sept. 2019). "A Framework for Bayesian Optimization in Embedded Subspaces". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 4752–4761. URL: https://proceedings.mlr.press/v97/nayebi19a.html.

Nealen, Andrew, Müller, Matthias, Keiser, Richard, Boxerman, Eddy, and Carlson, Mark (2006). "Physically Based Deformable Models in Computer Graphics". In: *Computer Graphics Forum* 25.4, pp. 809–836. DOI: 10.1111/j.1467-8659.2006.01000.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2006.01000.x.

Ng, Hing N and Grimsdale, Richard L (1996). "Computer graphics techniques for modeling cloth". In: *IEEE Computer Graphics and Applications* 16.5, pp. 28–41. ISSN: 0272-1716. DOI: 10.1109/38.536273. URL: https://doi.org/10.1109/38.536273.

Ng Hing N.and Grimsdale, Richard L. (1995). "GEOFF — A geometrical editor for fold formation". In: *Image Analysis Applications and Computer Graphics*. Ed. by Roland T. Chin, Horace H. S. Ip, Avi C. Naiman, and Ting-Chuen Pong. Berlin, Heidelberg: Springer-Verlag, pp. 124–131. ISBN: 3540606971.

Ngo, Cyril and Boivin, Samuel (2004). "Nonlinear cloth simulation". PhD thesis. INRIA.

Nocedal, Jorge and Wright, Stephen J (2006). *Numerical optimization*. Springer. DOI: 10.1007/978-0-387-40065-5. URL: https://link.springer.com/book/10.1007/978-0-387-40065-5.

Nvidia (2024). *PhysX physics engine*. URL: https://developer.nvidia.com/physx-sdk.

O'Brien, James F., Bargteil, Adam W., and Hodgins, Jessica K. (July 2002). "Graphical modeling and animation of ductile fracture". In: *ACM Trans. Graph.* 21.3, pp. 291–294. ISSN: 0730-0301. DOI: 10.1145/566654.566579. URL: https://doi.org/10.1145/566654.566579.

Ogden, Raymond William and Hill, Rodney (1972). "Large deformation isotropic elasticity – on the correlation of theory and experiment for incompressible rubberlike solids". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 326.1567, pp. 565–

584. DOI: 10.1098/rspa.1972.0026. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1972.0026.

Pandurangan, Pradeep, Eischen, Jeffrey, Kenkare, Narahari, and Lamar, Traci A. M. (2008). "Enhancing accuracy of drape simulation. Part II: Optimized drape simulation using industry-specific software". In: *The Journal of The Textile Institute* 99.3, pp. 219–226. DOI: 10.1080/00405000701489198. URL: https://doi.org/10.1080/00405000701489198.

Parsons, Ethan M., Weerasooriya, Tusit, Sarva, Sai, and Socrate, Simona (2010). "Impact of woven fabric: Experiments and mesostructure-based continuum-level simulations". In: *Journal of the Mechanics and Physics of Solids* 58.11, pp. 1995–2021. ISSN: 0022-5096. DOI: https://doi.org/10.1016/j.jmps.2010.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0022509610000967.

Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, Desmaison, Alban, Köpf, Andreas, Yang, Edward, DeVito, Zach, Raison, Martin, Tejani, Alykhan, Chilamkurthy, Sasank, Steiner, Benoit, Fang, Lu, Bai, Junjie, and Chintala, Soumith (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. DOI: 10.48550/arXiv.1912.01703. arXiv: 1912.01703 [cs.LG]. URL: https://doi.org/10.48550/arXiv.1912.01703.

Patel, Chaitanya, Liao, Zhouyingcheng, and Pons-Moll, Gerard (June 2020). "TailorNet: Predicting Clothing in 3D as a Function of Human Pose, Shape and Garment Style". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 7363–7373. DOI: 10.1109/CVPR42600.2020.00739. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00739.

Peirce, F. T. (1930). "26—THE "HANDLE" OF CLOTH AS A MEASURABLE QUANTITY". In: *Journal of the Textile Institute Transactions* 21.9, T377–T416. DOI: 10.1080/19447023008661529. URL: https://doi.org/10.1080/19447023008661529.

Peng, X.Q., Cao, J., Chen, J., Xue, P., Lussier, D.S., and Liu, L. (2004). "Experimental and numerical analysis on normalization of picture frame tests for composite materials". In: *Composites Science and Technology* 64.1, pp. 11–21. ISSN: 0266-3538. DOI: https://doi.org/

10.1016/S0266-3538(03)00202-1. URL: https://www.sciencedirect.com/science/article/pii/S0266353803002021.

Peng, Xiongqi and Cao, Jian (2005). "A continuum mechanics-based non-orthogonal constitutive model for woven composite fabrics". In: *Composites Part A: Applied Science and Manufacturing* 36.6, pp. 859–874. ISSN: 1359-835X. DOI: https://doi.org/10.1016/j.compositesa.2004.08.008. URL: https://www.sciencedirect.com/science/article/pii/S1359835X04002593.

Pennestrì, Ettore, Rossi, Valerio, Salvini, Pietro, and Valentini, Pier Paolo (2016). "Review and comparison of dry friction force models". In: *Nonlinear dynamics* 83, pp. 1785–1801. ISSN: 1573-269X. DOI: 10.1007/s11071-015-2485-3. URL: https://doi.org/10.1007/s11071-015-2485-3.

Petrak, Slavenka, Mahnić Naglić, Maja, Rogale, Dubravko, and Geršak, Jelka (2021). "Analysis of Polygonal Computer Model Parameters and Influence on Fabric Drape Simulation". In: *Materials* 14.21. ISSN: 1996-1944. DOI: 10.3390/ma14216259. URL: https://www.mdpi.com/1996-1944/14/21/6259.

Pfaff, Tobias, Fortunato, Meire, Sanchez-Gonzalez, Alvaro, and Battaglia, Peter W. (2021). *Learning Mesh-Based Simulation with Graph Networks*. DOI: 10.48550/arXiv.2010.03409. arXiv: 2010.03409 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2010.03409.

Pizana, José M., Rodríguez, Alejandro, Cirio, Gabriel, and Otaduy, Miguel A. (2020). "A Bending Model for Nodal Discretizations of Yarn-Level Cloth". In: *Computer Graphics Forum* 39.8, pp. 181–189. DOI: 10.1111/cgf.14112. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14112.

Portilla, Javier and Simoncelli, Eero P (2000). "A parametric texture model based on joint statistics of complex wavelet coefficients". In: *International journal of computer vision* 40, pp. 49–70. ISSN: 1573-1405. DOI: 10.1023/A:1026553619983. URL: https://doi.org/10.1023/A:1026553619983.

Powell, Michael JD (1971). "Recent advances in unconstrained optimization". In: *Mathematical programming* 1, pp. 26–57. ISSN: 1436-4646. DOI: 10.1007/BF01584071. URL: https://doi.org/10.1007/BF01584071.

Prevorsek, D.C., Butler, R.H., and Lamb, G.E.R. (1975). "Influence of Fiber Properties on Wrinkling Behavior of Fabrics Part I: Procedures and Apparatus". In: *Textile Research Journal* 45.1, pp. 60–67. DOI: 10.1177/004051757504500111. URL: https://doi.org/10.1177/004051757504500111.

Provot, Xavier (1995). "Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour". In: *Proceedings of Graphics Interface '95*. GI '95. Quebec, Canada: Canadian Human-Computer Communications Society, pp. 147–154. ISBN: 0-9695338-4-5. DOI: 10.20380/GI1995.17. URL: http://graphicsinterface.org/wp-content/uploads/gi1995-17.pdf.

Provot, Xavier (1997). "Collision and self-collision handling in cloth model dedicated to design garments". In: *Computer Animation and Simulation '97*. Ed. by Daniel Thalmann and Michiel van de Panne. Vienna: Springer Vienna, pp. 177–189. ISBN: 978-3-7091-6874-5. DOI: 10.1007/978-3-7091-6874-5_13. URL: https://doi.org/10.1007/978-3-7091-6874-5_13.

Qi, Charles R, Su, Hao, Mo, Kaichun, and Guibas, Leonidas J (July 2017). "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 77–85. DOI: 10.1109/CVPR.2017.16. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.16.

Quan, Wei, Teng, Dexin, Li, Hua, and Han, Cheng (May 2020). "An Extended Position Based Dynamics Method for Cloth Simulation". In: *Journal of Physics: Conference Series* 1550.3, p. 032083. DOI: 10.1088/1742-6596/1550/3/032083. URL: https://dx.doi.org/10.1088/1742-6596/1550/3/032083.

Raissi, Maziar, Perdikaris, Paris, and Karniadakis, George E (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378, pp. 686–

707. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045. URL: https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Rasheed, Abdullah Haroon, Romero, Victor, Bertails-Descoubes, Florence, Wuhrer, Stefanie, Franco, Jean-Sébastien, and Lazarus, Arnaud (2020). "Learning to Measure the Static Friction Coefficient in Cloth Contact". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9909–9918. DOI: 10.1109/CVPR42600.2020.00993. URL: https://doi.org/10.1109/CVPR42600.2020.00993.

Richardson, R.S.H. and Nolle, H. (1976). "Surface friction under time-dependent loads". In: *Wear* 37.1, pp. 87–101. ISSN: 0043-1648. DOI: 10.1016/0043-1648(76)90183-6. URL: https://www.sciencedirect.com/science/article/pii/0043164876901836.

Rohmer, Damien, Popa, Tiberiu, Cani, Marie-Paule, Hahmann, Stefanie, and Sheffer, Alla (Dec. 2010). "Animation wrinkling: augmenting coarse cloth simulations with realistic-looking wrinkles". In: *ACM Transactions on Graphics (ToG)* 29.6. ISSN: 0730-0301. DOI: 10.1145/1882261.1866183. URL: https://doi.org/10.1145/1882261.1866183.

Runia, Tom FH, Gavrilyuk, Kirill, Snoek, Cees GM, and Smeulders, Arnold WM (June 2020). "Cloth in the Wind: A Case Study of Physical Measurement Through Simulation". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 10495–10504. DOI: 10.1109/CVPR42600.2020.01051. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.01051.

Sanad, Reham, Cassidy, Tom, and Cheung, TLV (2012). "Fabric and garment drape measurement-Part 1". In: *Journal of Fiber Bioengineering and Informatics* 5.4, pp. 341–358. DOI: 10.3993/jfbi12201201. URL: http://dx.doi.org/10.3993/jfbi12201201.

Sanad, Reham, Cassidy, Tom, Cheung, Vien, and Evans, Elaine (2013). "Fabric and garment drape measurement-Part 2". In: *Binary Information Press & Textile Bioengineering and Informatics Society* 6.1, pp. 1–22. DOI: 10.3993/jfbi03201301. URL: https://dx.doi.org/doi:10.3993/jfbi03201301.

Sánchez-Banderas, Rosa M. and Otaduy, Miguel A. (2017). "Dissipation Potentials for Yarn-Level Cloth". In: *Spanish Computer Graphics Conference (CEIG)*. Ed. by Fco. Javier Melero

and Nuria Pelechano. The Eurographics Association. ISBN: 978-3-03868-046-8. DOI: 10.2312/ceig.20171202. URL: http://dx.doi.org/10.2312/ceig.20171202.

Sánchez-Banderas, Rosa M., Rodríguez, Alejandro, Barreiro, Héctor, and Otaduy, Miguel A. (Aug. 2020). "Robust eulerian-on-lagrangian rods". In: *ACM Trans. Graph.* 39.4. ISSN: 0730-0301. DOI: 10.1145/3386569.3392489. URL: https://doi.org/10.1145/3386569.3392489.

Sanchez-Gonzalez, Alvaro, Godwin, Jonathan, Pfaff, Tobias, Ying, Rex, Leskovec, Jure, and Battaglia, Peter W. (2020). "Learning to simulate complex physics with graph networks". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org. DOI: 10.5555/3524938.3525722. URL: http://proceedings.mlr.press/v119/sanchez-gonzalez20a/sanchez-gonzalez20a.pdf.

Santesteban, Igor, Otaduy, Miguel A., and Casas, Dan (2019). "Learning-Based Animation of Clothing for Virtual Try-On". In: *Computer Graphics Forum* 38.2, pp. 355–366. DOI: 10.1111/cgf.13643. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13643.

Schenck, Connor and Fox, Dieter (29–31 Oct 2018). "SPNets: Differentiable Fluid Dynamics for Deep Neural Networks". In: *Proceedings of The 2nd Conference on Robot Learning*. Ed. by Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto. Vol. 87. Proceedings of Machine Learning Research. PMLR, pp. 317–335. URL: https://proceedings.mlr.press/v87/schenck18a.html.

Schulman, John, Wolski, Filip, Dhariwal, Prafulla, Radford, Alec, and Klimov, Oleg (2017). *Proximal Policy Optimization Algorithms*. DOI: 10.48550/arXiv.1707.06347. arXiv: 1707.06347 [cs.LG]. URL: https://doi.org/10.48550/arXiv.1707.06347.

Selle, Andrew, Su, Jonathan, Irving, Geoffrey, and Fedkiw, Ronald (2008). "Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction". In: *IEEE Transactions on Visualization and Computer Graphics* 15.2, pp. 339–350. DOI: 10.1109/TVCG.2008.79. URL: https://pubmed.ncbi.nlm.nih.gov/19147895/.

Shao, Xuqiang, Wu, Wei, and Wang, Baoyi (2018). "Position-based simulation of cloth wetting phenomena". In: *Computer Animation and Virtual Worlds* 29.1. e1788 cav.1788, e1788. DOI: https://doi.org/10.1002/cav.1788. eprint: https://onlinelibrary.wiley.com/doi/

pdf/10.1002/cav.1788. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.1788.

Shen, Siyuan, Yang, Yin, Shao, Tianjia, Wang, He, Jiang, Chenfanfu, Lan, Lei, and Zhou, Kun (July 2021). "High-order differentiable autoencoder for nonlinear model reduction". In: *ACM Trans. Graph.* 40.4. ISSN: 0730-0301. DOI: 10.1145/3450626.3459754. URL: https://doi.org/10.1145/3450626.3459754.

Shewchuk, Jonathan R (1994). *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep. USA. URL: https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf.

Sifakis, Eftychios, Marino, Sebastian, and Teran, Joseph (2008). "Globally coupled collision handling using volume preserving impulses". In: *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '08. Dublin, Ireland: Eurographics Association, pp. 147–153. ISBN: 9783905674101.

Skelton, J. (1968). "22—A THEORETICAL AND EXPERIMENTAL INVESTIGATION OF THE CREASE RECOVERY OF PLAIN-WEAVE FABRICS WOVEN FROM STAPLE YARNS". In: *The Journal of The Textile Institute* 59.6, pp. 261–284. DOI: 10.1080/00405006808659986. URL: https://www.tandfonline.com/doi/abs/10.1080/00405006808659986.

Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. (2012). "Practical Bayesian optimization of machine learning algorithms". In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'12. Lake Tahoe, Nevada: Curran Associates Inc., pp. 2951–2959. DOI: 10.5555/2999325.2999464. URL: https://dl.acm.org/doi/10.5555/2999325.2999464.

Sperl, Georg, Narain, Rahul, and Wojtan, Chris (Aug. 2020). "Homogenized yarn-level cloth". In: *ACM Trans. Graph.* 39.4. ISSN: 0730-0301. DOI: 10.1145/3386569.3392412. URL: https://doi.org/10.1145/3386569.3392412.

Sperl, Georg, Narain, Rahul, and Wojtan, Chris (July 2021). "Mechanics-aware deformation of yarn pattern geometry". In: *ACM Trans. Graph.* 40.4. ISSN: 0730-0301. DOI: 10.1145/3450626.3459816. URL: https://doi.org/10.1145/3450626.3459816.

Sperl, Georg, Sánchez-Banderas, Rosa M., Li, Manwen, Wojtan, Chris, and Otaduy, Miguel A. (July 2022). "Estimation of yarn-level simulation models for production fabrics". In: *ACM Trans. Graph.* 41.4. ISSN: 0730-0301. DOI: 10.1145/3528223.3530167. URL: https://doi.org/10.1145/3528223.3530167.

Spillmann, J. and Teschner, M. (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*. SCA '07. San Diego, California: Eurographics Association, pp. 63–72. ISBN: 9781595936240.

Strang, Gilbert (2012). *Linear algebra and its applications*.

Stribeck, Richard (1902). "Die Wesentlichen Eigenschaften der Gieit-und Rollenlager–the key qualities of sliding and roller bearings". In: *Zeitschrift des Vereines Seutscher Ingenieure* 46.38, pp. 1342–1348.

Stuyck, Tuur (2022). *Cloth simulation for computer graphics*. Springer Nature.

Stylios, George K., Powell, Norman J., and Cheng, Lu (2002). "An investigation into the engineering of the drapability of fabric". In: *Transactions of the Institute of Measurement and Control* 24.1, pp. 33–50. DOI: 10.1191/0142331202tm049oa. URL: https://doi.org/10.1191/0142331202tm049oa.

Sueda, Shinjiro, Jones, Garrett L., Levin, David I. W., and Pai, Dinesh K. (July 2011). "Large-scale dynamic simulation of highly constrained strands". In: *ACM Trans. Graph.* 30.4. ISSN: 0730-0301. DOI: 10.1145/2010324.1964934. URL: https://doi.org/10.1145/2010324.1964934.

Sun, Fengxin and Hu, Xiaorui (July 2020). "Effect of meso-scale structures and hyper-viscoelastic mechanics on the nonlinear tensile stability and hysteresis of woven materials". In: *Materials Research Express* 7.7, p. 075306. DOI: 10.1088/2053-1591/aba6be. URL: https://dx.doi.org/10.1088/2053-1591/aba6be.

Swevers, J., Al-Bender, F., Ganseman, C.G., and Projogo, T. (2000). "An integrated friction model structure with improved presliding behavior for accurate friction compensation". In:

IEEE Transactions on Automatic Control 45.4, pp. 675–686. DOI: 10.1109/9.847103. URL: https://doi.org/10.1109/9.847103.

Tamstorf, Rasmus and Grinspun, Eitan (Nov. 2013). "Discrete Bending Forces and Their Jacobians". In: *Graph. Models* 75.6, pp. 362–370. ISSN: 1524-0703. DOI: 10.1016/j.gmod.2013.07.001. URL: http://dx.doi.org/10.1016/j.gmod.2013.07.001.

Tang, Min, Liu, Zhongyuan, Tong, Ruofeng, and Manocha, Dinesh (July 2018a). "PSCC: Parallel Self-Collision Culling with Spatial Hashing on GPUs". In: 1.1. DOI: 10.1145/3203188. URL: https://doi.org/10.1145/3203188.

Tang, Min, Manocha, Dinesh, Lin, Jiang, and Tong, Ruofeng (2011). "Collision-streams: fast GPU-based collision detection for deformable models". In: *Symposium on Interactive 3D Graphics and Games*. I3D '11. San Francisco, California: Association for Computing Machinery, pp. 63–70. ISBN: 9781450305655. DOI: 10.1145/1944745.1944756. URL: https://doi.org/10.1145/1944745.1944756.

Tang, Min, Manocha, Dinesh, and Tong, Ruofeng (2010). "Fast continuous collision detection using deforming non-penetration filters". In: *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. I3D '10. Washington, D.C.: Association for Computing Machinery, pp. 7–13. ISBN: 9781605589398. DOI: 10.1145/1730804.1730806. URL: https://doi.org/10.1145/1730804.1730806.

Tang, Min, Tong, Ruofeng, Narain, Rahul, Meng, Chang, and Manocha, Dinesh (2013). "A GPU-based Streaming Algorithm for High-Resolution Cloth Simulation". In: *Computer Graphics Forum* 32.7, pp. 21–30. DOI: 10.1111/cgf.12208. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12208.

Tang, Min, Wang, Tongtong, Liu, Zhongyuan, Tong, Ruofeng, and Manocha, Dinesh (Dec. 2018b). "I-cloth: incremental collision handling for GPU-based interactive cloth simulation". In: *ACM Trans. Graph.* 37.6. ISSN: 0730-0301. DOI: 10.1145/3272127.3275005. URL: https://doi.org/10.1145/3272127.3275005.

Terzopoulos, Demetri and Fleischer, Kurt (June 1988). "Modeling inelastic deformation: viscolelasticity, plasticity, fracture". In: *SIGGRAPH Comput. Graph.* 22.4, pp. 269–278. ISSN: 0097-8930. DOI: 10.1145/378456.378522. URL: https://doi.org/10.1145/378456.378522.

Terzopoulos, Demetri, Platt, John, Barr, Alan, and Fleischer, Kurt (Aug. 1987). "Elastically deformable models". In: *SIGGRAPH Comput. Graph.* 21.4, pp. 205–214. ISSN: 0097-8930. DOI: 10.1145/37402.37427. URL: https://doi.org/10.1145/37402.37427.

Thomaszewski, Bernhard, Pabst, Simon, and Straßer, Wolfgang (2009). "Continuum-based Strain Limiting". In: *Computer Graphics Forum* 28.2, pp. 569–576. DOI: 10.1111/j.1467-8659.2009.01397.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2009.01397.x.

Thomaszewski, Bernhard, Wacker, Markus, Straßer, Wolfgang, Lyard, Etienne, Luible, C., Volino, Pascal, Kasap, M., Muggeo, V., and Magnenat-Thalmann, Nadia (2007). "Advanced Topics in Virtual Garment Simulation". In: *Eurographics 2007 - Tutorials*. Ed. by Karol Myszkowski and Vlastimil Havran. The Eurographics Association. DOI: 10.2312/egt.20071069. URL: http://dx.doi.org/10.2312/egt.20071069.

Tortorelli, D. A. and Michaleris, P. (1994). "Design sensitivity analysis: Overview and review". In: *Inverse Problems in Engineering* 1.1, pp. 71–105. DOI: 10.1080/174159794088027573. URL: https://doi.org/10.1080/174159794088027573.

Treloar, L. R. G. (1965). "42—THE EFFECT OF TEST-PIECE DIMENSIONS ON THE BEHAVIOUR OF FABRICS IN SHEAR". In: *Journal of the Textile Institute Transactions* 56.10, T533–T550. DOI: 10.1080/19447026508662313. URL: https://doi.org/10.1080/19447026508662313.

Umetani, Nobuyuki, Schmidt, Ryan, and Stam, Jos (2015). "Position-based elastic rods". In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '14. Copenhagen, Denmark: Eurographics Association, pp. 21–30. DOI: 10.5555/2849517.2849522. URL: https://dl.acm.org/doi/abs/10.5555/2849517.2849522.

Vassilev, T., Spanlang, B., and Chrysanthou, Y. (2001). "Fast Cloth Animation on Walking Avatars". In: *Computer Graphics Forum* 20.3, pp. 260–267. DOI: 10.1111/1467-8659.00518. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/1467-8659.00518.

Verma, Nitika, Boyer, Edmond, and Verbeek, Jakob (June 2018). "FeaStNet: Feature-Steered Graph Convolutions for 3D Shape Analysis". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 2598–2606. DOI: 10.1109/CVPR.2018.00275. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00275.

Vidaurre, Raquel, Santesteban, Igor, Garces, Elena, and Casas, Dan (2020). "Fully Convolutional Graph Neural Networks for Parametric Virtual Try-On". In: *Computer Graphics Forum* 39.8, pp. 145–156. DOI: 10.1111/cgf.14109. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14109.

Volino, Pascal, Cordier, Frederic, and Magnenat-Thalmann, Nadia (May 2005). "From early virtual garment simulation to interactive fashion design". In: *Comput. Aided Des.* 37.6, pp. 593–608. ISSN: 0010-4485. DOI: 10.1016/j.cad.2004.09.003. URL: https://doi.org/10.1016/j.cad.2004.09.003.

Volino, Pascal, Courchesne, Martin, and Magnenat Thalmann, Nadia (1995). "Versatile and efficient techniques for simulating cloth and other deformable objects". In: *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '95. New York, NY, USA: Association for Computing Machinery, pp. 137–144. ISBN: 0897917014. DOI: 10.1145/218380.218432. URL: https://doi.org/10.1145/218380.218432.

Volino, Pascal and Magnenat-Thalmann, Nadia (2000). *Virtual clothing: Theory and practice*. Springer Science & Business Media.

Volino, Pascal and Magnenat-Thalmann, Nadia (2005). "Implicit midpoint integration and adaptive damping for efficient cloth simulation". In: *Computer Animation and Virtual Worlds* 16.3-4, pp. 163–175. DOI: 10.1002/cav.78. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.78.

Volino, Pascal, Magnenat-Thalmann, Nadia, and Faure, Francois (Sept. 2009). "A simple approach to nonlinear tensile stiffness for accurate cloth simulation". In: *ACM Trans. Graph.* 28.4. ISSN: 0730-0301. DOI: 10.1145/1559755.1559762. URL: https://doi.org/10.1145/1559755.1559762.

Wang, Huamin (July 2014). "Defending continuous collision detection against errors". In: *ACM Trans. Graph.* 33.4. ISSN: 0730-0301. DOI: 10.1145/2601097.2601114. URL: https://doi.org/10.1145/2601097.2601114.

Wang, Huamin (July 2021). "GPU-based simulation of cloth wrinkles at submillimeter levels". In: *ACM Trans. Graph.* 40.4. ISSN: 0730-0301. DOI: 10.1145/3450626.3459787. URL: https://doi.org/10.1145/3450626.3459787.

Wang, Huamin, Hecht, Florian, Ramamoorthi, Ravi, and O'Brien, James F. (July 2010a). "Example-based wrinkle synthesis for clothing animation". In: *ACM Trans. Graph.* 29.4. ISSN: 0730-0301. DOI: 10.1145/1778765.1778844. URL: https://doi.org/10.1145/1778765.1778844.

Wang, Huamin, O'Brien, James, and Ramamoorthi, Ravi (Dec. 2010b). "Multi-resolution isotropic strain limiting". In: *ACM Trans. Graph.* 29.6. ISSN: 0730-0301. DOI: 10.1145/1882261.1866182. URL: https://doi.org/10.1145/1882261.1866182.

Wang, Huamin, O'Brien, James F., and Ramamoorthi, Ravi (2011). "Data-driven elastic models for cloth: modeling and measurement". In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH '11. Vancouver, British Columbia, Canada: Association for Computing Machinery. ISBN: 9781450309431. DOI: 10.1145/1964921.1964966. URL: https://doi.org/10.1145/1964921.1964966.

Wang, Jun, Paton, Rowan, and Page, John R. (1999). "The draping of woven fabric preforms and prepregs for production of polymer composite components". In: *Composites Part A: Applied Science and Manufacturing* 30.6, pp. 757–765. ISSN: 1359-835X. DOI: https://doi.org/10.1016/S1359-835X(98)00187-0. URL: https://www.sciencedirect.com/science/article/pii/S1359835X98001870.

Wang, Kun, Aanjaneya, Mridul, and Bekris, Kostas (Oct. 2020). "A First Principles Approach for Data-Efficient System Identification of Spring-Rod Systems via Differentiable Physics Engines". In: *Proceedings of the 2nd Conference on Learning for Dynamics and Control*. Ed. by Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger. Vol. 120. Proceedings of Machine Learning Research. PMLR, pp. 651–665. URL: https://proceedings.mlr.press/v120/wang20b.html.

Wang, Tuanfeng Y., Ceylan, Duygu, Popović, Jovan, and Mitra, Niloy J. (Dec. 2018). "Learning a shared shape space for multimodal garment design". In: *ACM Trans. Graph.* 37.6. ISSN: 0730-0301. DOI: 10.1145/3272127.3275074. URL: https://doi.org/10.1145/3272127.3275074.

Wang, X., Liu, X., and Deakin, C.Hurren (2008). "4 - Physical and mechanical testing of textiles". In: *Fabric Testing*. Ed. by Jinlian Hu. Woodhead Publishing Series in Textiles. Woodhead Publishing, pp. 90–124. ISBN: 978-1-84569-297-1. DOI: https://doi.org/10.1533/9781845695064.90. URL: https://www.sciencedirect.com/science/article/pii/B9781845692971500048.

Wang, Ziyu, Hutter, Frank, Zoghi, Masrour, Matheson, David, and De Feitas, Nando (2016). *Bayesian Optimization in a Billion Dimensions via Random Embeddings*. DOI: 10.48550/arXiv.1301.1942. arXiv: 1301.1942 [stat.ML]. URL: https://doi.org/10.48550/arXiv.1301.1942.

Weil, Jerry (Aug. 1986). "The synthesis of cloth objects". In: *SIGGRAPH Comput. Graph.* 20.4, pp. 49–54. ISSN: 0097-8930. DOI: 10.1145/15886.15891. URL: https://doi.org/10.1145/15886.15891.

Wong, Tsz Ho (2014). "Improvements to physically based cloth simulation". eng. PhD thesis. RMIT University. URL: https://researchrepository.rmit.edu.au/esploro/outputs/9921861982301341.

Wong, Tsz Ho, Leach, Geoff, and Zambetta, Fabio (2013). "Modelling Bending Behaviour in Cloth Simulation Using Hysteresis". In: *Computer Graphics Forum* 32.8, pp. 183–194. DOI: 10.1111/cgf.12196. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12196.

Wu, Jiajun, Lim, Joseph J, Zhang, Hongyi, Tenenbaum, Joshua B, and Freeman, William T (2016). "Physics 101: Learning Physical Object Properties from Unlabeled Videos." In: *BMVC*. Vol. 2. 6, p. 7. URL: http://phys101.csail.mit.edu/papers/phys101_bmvc.pdf.

Wu, Jiajun, Yildirim, Ilker, Lim, Joseph J, Freeman, Bill, and Tenenbaum, Josh (2015). "Galileo: Perceiving Physical Object Properties by Integrating a Physics Engine with Deep Learning". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/d09bf41544a3365a46c9077ebb5e35c3-Paper.pdf.

Xiao, Yijun and Wang, William Yang (2019). "Quantifying uncertainties in natural language processing tasks". In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI'19/IAAI'19/EAAI'19. Honolulu, Hawaii, USA: AAAI Press. ISBN: 978-1-57735-809-1. DOI: 10.1609/aaai.v33i01.33017322. URL: https://doi.org/10.1609/aaai.v33i01.33017322.

Xu, Weiwei, Umentani, Nobuyuki, Chao, Qianwen, Mao, Jie, Jin, Xiaogang, and Tong, Xin (July 2014). "Sensitivity-optimized rigging for example-based real-time clothing synthesis". In: *ACM Trans. Graph.* 33.4. ISSN: 0730-0301. DOI: 10.1145/2601097.2601136. URL: https://doi.org/10.1145/2601097.2601136.

Yang, Liu, Meng, Xuhui, and Karniadakis, George Em (2021). "B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data". In: *Journal of Computational Physics* 425, p. 109913. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2020.109913. URL: https://www.sciencedirect.com/science/article/pii/S0021999120306872.

Yang, Shan, Liang, Junbang, and Lin, Ming C. (2017). "Learning-Based Cloth Material Recovery from Video". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4393–4403. DOI: 10.1109/ICCV.2017.470. URL: https://doi.org/10.1109/ICCV.2017.470.

Zhang, Xinyu and Kim, Young J. (2014). "Scalable Collision Detection Using p-Partition Fronts on Many-Core Processors". In: *IEEE Transactions on Visualization and Computer Graphics* 20.3, pp. 447–456. DOI: 10.1109/TVCG.2013.239. URL: https://doi.org/10.1109/TVCG.2013.239.

Zhang, Zheyan, Jimack, Peter K., and Wang, He (July 2021). "MeshingNet3D: Efficient generation of adapted tetrahedral meshes for computational mechanics". In: *Advances in Engineering Software* 157.C. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2021.103021. URL: https://doi.org/10.1016/j.advengsoft.2021.103021.

Zhang, Zheyan, Wang, Yongxing, Jimack, Peter K., and Wang, He (2020). "MeshingNet: A New Mesh Generation Method Based on Deep Learning". In: *Computational Science – ICCS 2020*. Ed. by Valeria V. Krzhizhanovskaya, Gábor Závodszky, Michael H. Lees, Jack J. Dongarra, Peter M. A. Sloot, Sérgio Brissos, and João Teixeira. Cham: Springer International Publishing, pp. 186–198. ISBN: 978-3-030-50420-5.

Zhong, Yaofeng Desmond, Dey, Biswadip, and Chakraborty, Amit (Feb. 2021). "A Differentiable Contact Model to Extend Lagrangian and Hamiltonian Neural Networks for Modeling Hybrid Dynamics". In: DOI: 10.48550/arXiv.2102.06794. arXiv: 2102.06794 [cs.RO]. URL: https://doi.org/10.48550/arXiv.2102.06794.

Zhong, Yaofeng Desmond, Han, Jiequn, and Brikis, Georgia Olympia (2022). *Differentiable Physics Simulations with Contacts: Do They Have Correct Gradients w.r.t. Position, Velocity and Control?* DOI: 10.48550/arXiv.2207.05060. arXiv: 2207.05060 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2207.05060.

Zhou, Yi, Ali, Muhammad, Gong, Xiaozhou, and Yang, Dan (2019). "An overview of yarn pull-out behavior of woven fabrics". In: *Textile Research Journal* 89.2, pp. 223–234. DOI: 10.1177/0040517517741156. URL: https://doi.org/10.1177/0040517517741156.

Appendix A

Appendix One: Yarn-level Differentiable Cloth Simulator

All simulations are available in the accompanied videos:`https://youtu.be/pCB8AD9R4Dk`

A.1 Training Details

Our ground-truth data is simulated with a piece of cloth hanging at its two corners, blown by a wind with a constant magnitude (Fig. A.1). The simulation is conducted with a time step h = 0.001. In all experiments, we use Stochastic Gradient Descent and run 70 epochs for training, except in XXX-(1,3) where we trained our model for 90 epochs. The training is conducted on a machine with Intel(R) Xeon(R) Silver 4216 CPU, 187G memory, NVIDIA TITAN RTX graphics card on Linux. The main factors of training speed are the cloth size and the training data size. In our experiments, the training takes approximately 68, 133, and 328 seconds per epoch on a 17×17 cloth with training data containing 5, 10, and 25 frames respectively. The training per epoch takes approximately 13, 106, 328, and 1310 seconds with 25 training frames, on a 5×5, 10×10, 17×17, and 25×25 cloth respectively.

Additional experiments. Further, we also conduct comparisons on the data simulated under the same settings by a sheet-level simulator (Narain et al. 2012), which tends to be stiffer. This is to compare the performance when the ground-truth does not contain the same level of subtle dynamics. Since there is no Eulerian coordinates in the sheet-level simulation, we only use Lagragian coordinates in the loss function. The visual comparison is in Fig. A.2 and

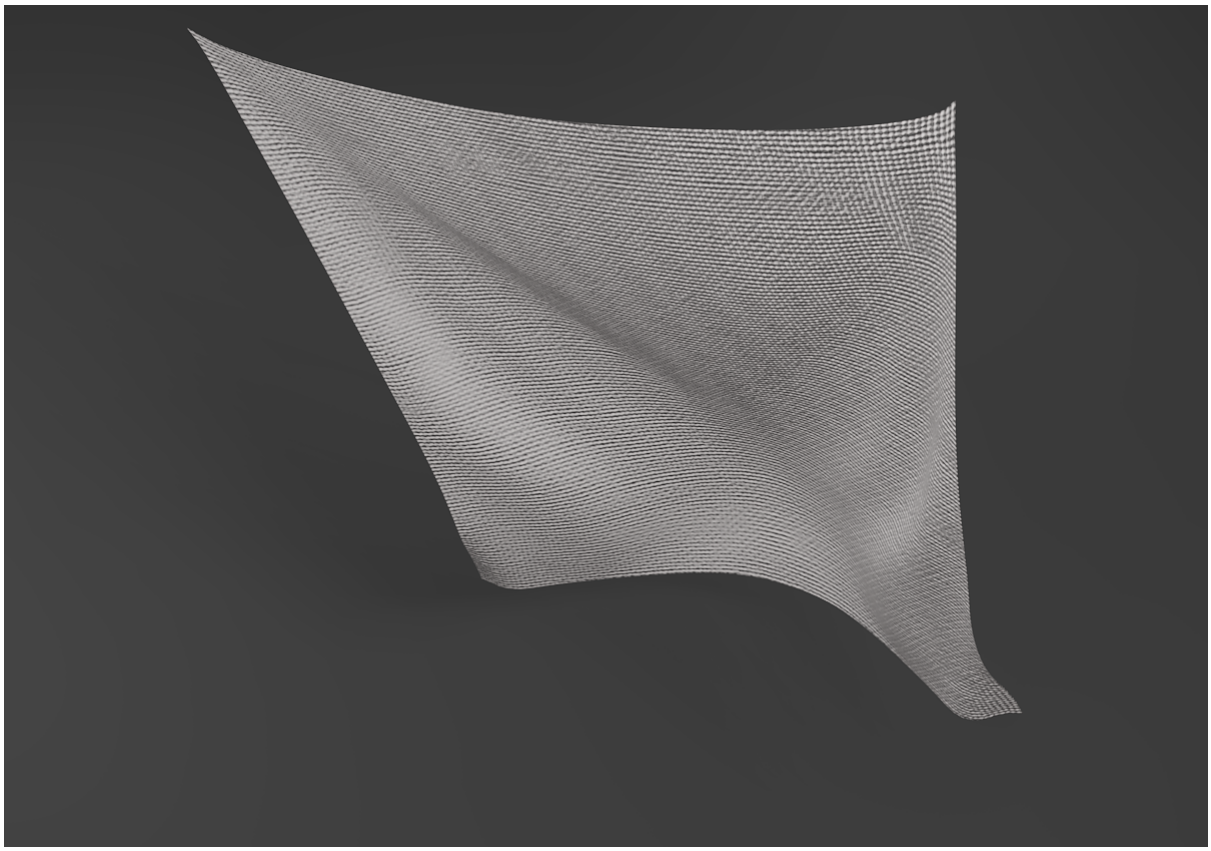


Figure A.1: A piece of square cloth blown by constant magnitude wind.

the prediction errors are shown in Table A.1. Our model can learn comparable results on 5 frames, and better results on 10 and 25 frames. The slightly worse 5-frame result is mainly because the first 5 frames contain small dynamics and therefore is insufficient for our model to learn the overall stiffness of the cloth. However, when 10 and 25 frames are given, the learning is significantly improved and even outperforms (Liang et al. 2019). Also, since there is no woven pattern information in the ground truth, we examine our model across the three woven patterns, all giving more accurate predictions. Overall, the comparisons show our model has higher prediction accuracy regardless the granularity of the underlying physics model.

Parameters. We induce prior knowledge to limit the parameter learning within valid ranges, so that the multi-solution problem, also met by existing methods, can be mitigated. All cloths we used are made of two types of yarns. We use the same range, $d \in [0.001, 0.003]$, $b \in [0.00005, 0.00018]$, $S \in [0, 1200]$ and $\mu \in [0, 1.0]$ for both yarns, where d, b, S and μ are the density, bending modulus, shear modulus and friction coefficient respectively. We use $s_1 \in [0, 800000]$ and $s_2 \in [0, 300000]$ for the stretching for both yarns. For other coefficients, we use $k_f = 1000$ and $d_f = 1000$ in the friction force, $c = 3$ and $\sigma = 0.6$ in the shear force, $k_c = 1$ in

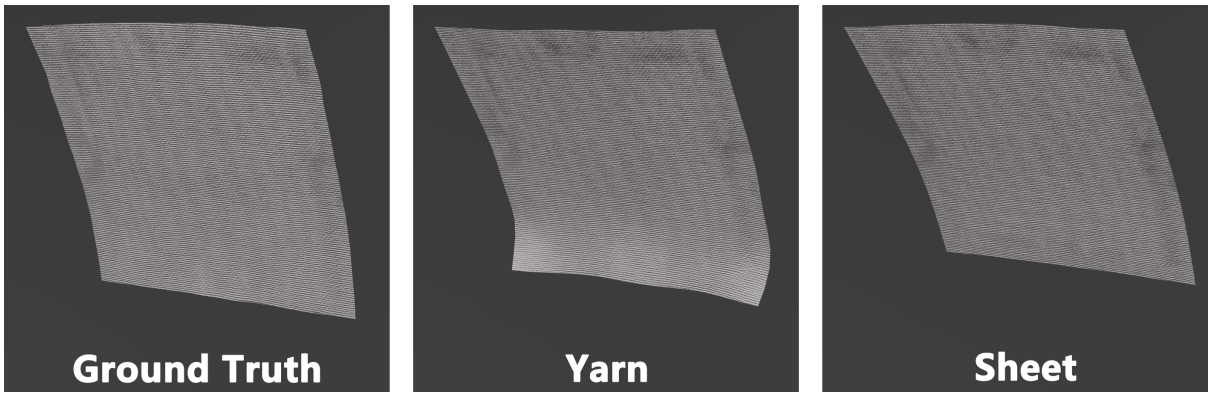


Figure A.2: The visual learning results of the differentiable sheet-level simulator (Liang et al. 2019) and our model learns on the data generate by (Narain et al. 2012)

Table A.1: Testing errors ($\times 10^{-6}$) of our model and (Liang et al. 2019) trained on 5, 10 and 25 frames generated by (Narain et al. 2012).

fabrics/frames	5	10	25
Plain-(1,2)	6.702	1.167	0.496
Satin-(1,2)	7.972	1.225	0.624
Twill-(1,2)	8.218	1.772	0.776
(Liang et al. 2019)	4.098	4.752	1.716

yarn-to-yarn collision in all experiments.

When training our model on the data generated by a sheet-level cloth simulator (Narain et al. 2012), we use a pure woven cloth made of one type of yarn. This is because it is not possible to specify multiple yarn behaviors in a sheet simulator, so we use a pure yarn cloth for generating the ground truth. The cloth parameters are from the 'white-dots-on-black' cloth in the work proposed by Wang et al. (2011) which is 100 percent polyester. To learn from it, we employ all three woven patterns in our model as there is no prior knowledge about the woven pattern of the 'white-dots-on-black' cloth. We also fix the friction coefficient $\mu = 0.5$ and impose the ranges on parameters shown in Table A.5. Finally, we would like to point it out in real-world applications, information such as woven patterns and yarn materials are easily available so that the ranges of parameter values such as density, bending and stretching can be obtained. Although the knowledge of shearing and friction cannot be easily acquired, the ranges we use are general enough.

Note that in all experiments, the prior knowledge we induce is only a weak prior, i.e. using the same general ranges for multiple experiments across different woven patterns, so that the learning success still lies in our model's ability to infer the right parameter values.

Table A.2: Learning cloth parameters with different initial values (part one).

Size	Shear S	Friction μ
5×5	1011.79 ± 6.12	0.39 ± 0.08
10×10	983.41 ± 6.84	0.44 ± 0.03
17×17	962.29 ± 8.99	0.47 ± 0.06

Table A.3: Learning cloth parameters with different initial values (part two).

Size	Yarn	Density	Stretch	Bend
5×5	1	$1.98 \times 10^{-3} \pm 3.00 \times 10^{-5}$	498595 ± 8862	$1.37 \times 10^{-4} \pm 1.41 \times 10^{-6}$
	2	$2.45 \times 10^{-3} \pm 4.81 \times 10^{-5}$	186710 ± 3776	$1.11 \times 10^{-4} \pm 4.78 \times 10^{-6}$
10×10	1	$2.03 \times 10^{-3} \pm 5.04 \times 10^{-5}$	542375 ± 7099	$1.44 \times 10^{-4} \pm 2.08 \times 10^{-6}$
	2	$2.47 \times 10^{-3} \pm 4.73 \times 10^{-5}$	180032 ± 1848	$1.05 \times 10^{-4} \pm 8.18 \times 10^{-6}$
17×17	1	$2.00 \times 10^{-3} \pm 6.66 \times 10^{-5}$	519993 ± 3175	$1.43 \times 10^{-4} \pm 5.55 \times 10^{-6}$
	2	$2.45 \times 10^{-3} \pm 5.04 \times 10^{-5}$	176232 ± 1514	$1.19 \times 10^{-4} \pm 6.50 \times 10^{-6}$

Parameter Initialization. The material estimation results are affected by initialization. To test if our model can learn stably, we report the mean and the standard deviation of multiple experiments with different parameter initial values. The initial values of the physical parameters are randomly selected from a range of $\pm 10\%$ of the average of the two yarns. For instance, in learning the stretch in Plain-(1,2), we only know the ranges of the stretching parameters Y1 and Y2 of Yarn1 and Yarn2 but not the exact values. Therefore, when initializing Y1 and Y2, we randomly sample values from a range of $\pm 10\%$ of the mean stretch stiffness of the Yarn1 and Yarn2, $[\text{mean(Y1, Y2)} \times 0.9, \text{mean(Y1, Y2)} \times 1.1]$ for initialization. The results of the 5 repetitions are shown Table A.2 and Table A.3. Given that the standard deviations are small, it shows that our model can stably learn reasonable parameter values.

Different Force Magnitude. To evaluate the influence of the wind force, we conduct experiments using 5N, 10N, and 15N wind force to blow a piece of 17×17 Plain-(1,2) cloth. The learning result is shown in the Table A.4 which demonstrate wind force strength has ignorable influence on the learned parameters.

Influence of Woven Patterns. The investigation on different woven patterns is crucial as they affect the cloth dynamics significantly. To show this, we conducted simulations of three pieces of cloths with the same parameters, but with different woven patterns. We shear three pieces of cloth then release them. The Fig. A.3 shows three pieces of cloth in the initial state and 10 steps later. There are obvious differences after merely 10 steps. This demonstrates

Table A.4: Learning cloth physical parameters with different wind force.

Wind	Shear S	Friction μ	Yarn	Density	Stretch	Bend
5	947	0.402	1	1.969×10^{-3}	505421	1.323×10^{-4}
			2	2.440×10^{-3}	171304	1.034×10^{-4}
10	942	0.520	1	2.026×10^{-3}	494109	1.311×10^{-4}
			2	2.441×10^{-3}	168267	1.049×10^{-4}
15	934	0.586	1	2.029×10^{-3}	487918	1.341×10^{-4}
			2	2.437×10^{-3}	167601	1.066×10^{-4}

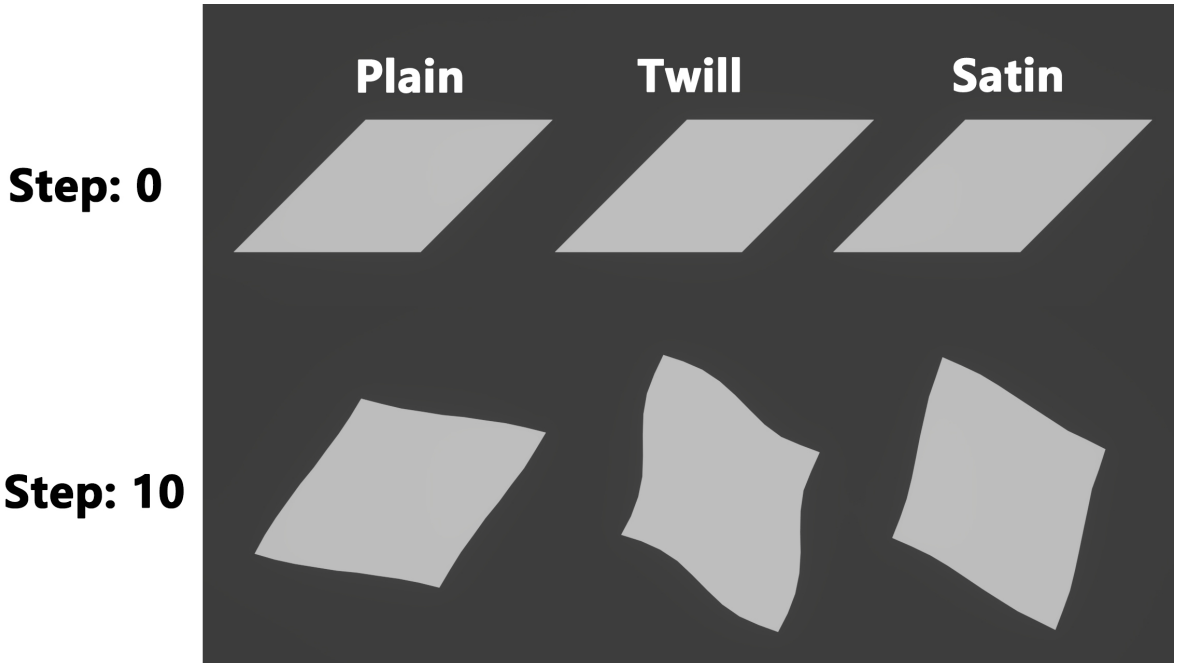


Figure A.3: Three pieces of cloth woven in different patterns show different dynamics.

woven patterns have considerable influences on the overall mechanical properties.

A.2 Visual results

Here we show some snapshots of our model on cloths of different sizes in Fig. A.4. As expected, small cloths tend to show low dynamics and appear to be more 'rigid'. Bigger cloths tend to have more subtle dynamics such as wrinkles, even under the same external impact, i.e. gravity

Table A.5: Cloth parameters' initial values and ranges when ground-truth generated by sheet-level cloth simulator(Narain et al. 2012)

Name	Density(kg/m)	Stretch(N/m)	Bend(N/m)	Shear(N/m)
Initial Value	0.004	1e6	0.0001	20000
Upper limit	0.008	2e6	0.0002	30000
Lower limit	0.001	0	0	0

Table A.6: Testing errors ($\times 10^{-6}$) of our model (left) and (Liang et al. 2019) (right) trained on 5, 10 and 25 frames. Ground-truth generated by a yarn-level simulator (Cirio et al. 2014).

fabrics/frames	5	10	25	5	10	25
Plain-(1,2)	1.152×10^{-4}	1.068×10^{-4}	3.962×10^{-5}	1.462	0.7375	0.4124
Plain-(1,3)	1.516×10^{-4}	1.268×10^{-4}	3.555×10^{-5}	1.608	0.7906	0.4567
Plain-(2,3)	5.233×10^{-4}	1.291×10^{-4}	2.117×10^{-5}	1.952	0.5999	0.2294
Satin-(1,2)	1.134×10^{-4}	1.070×10^{-4}	4.285×10^{-5}	1.466	0.7405	0.4146
Satin-(1,3)	1.551×10^{-4}	1.355×10^{-4}	4.362×10^{-5}	1.624	0.8004	0.4445
Satin-(2,3)	6.254×10^{-4}	1.355×10^{-4}	4.413×10^{-5}	2.128	0.5949	0.2265
Twill-(1,2)	1.130×10^{-4}	1.068×10^{-4}	4.208×10^{-5}	1.472	0.7451	0.4160
Twill-(1,3)	1.550×10^{-4}	1.349×10^{-4}	4.200×10^{-5}	1.633	0.8059	0.4577
Twill-(2,3)	6.470×10^{-4}	1.352×10^{-4}	4.938×10^{-5}	2.181	0.5994	0.2278

Figure A.4: The visual results of our model learning on different cloth sizes. From left to right: 5×5, 10×10, 17×17 and 25×25.

and wind with a constant magnitude. More visual results can be found in the supplementary video.

A.3 Yarn-level versus Sheet-level

A full comparison between our model and the model proposed by Liang et al. (2019) is shown in Table A.6, where a yarn-level simulator (Cirio et al. 2014) is used to generate the ground-truth.

Table A.7: Learned parameters by Bayesian Optimization on different kinds of fabrics.

Frames	Density	Stretch	Bend	Density	Stretch	Bend
5	2.483×10^{-3}	647270	0.636×10^{-4}	2.125×10^{-3}	270641	1.576×10^{-4}
10	2.176×10^{-3}	577235	0.798×10^{-4}	2.264×10^{-3}	217144	1.542×10^{-4}
25	2.328×10^{-3}	537434	1.687×10^{-4}	2.097×10^{-3}	249896	0.976×10^{-4}
5	2.202×10^{-3}	605289	1.403×10^{-4}	2.349×10^{-3}	272153	0.868×10^{-4}
10	1.669×10^{-3}	257877	1.582×10^{-4}	2.635×10^{-3}	268451	0.529×10^{-4}
25	1.454×10^{-3}	315715	1.213×10^{-4}	2.950×10^{-3}	23702	1.656×10^{-4}
5	2.514×10^{-3}	250093	1.611×10^{-4}	2.363×10^{-3}	20371	0.985×10^{-4}
10	2.964×10^{-3}	164021	0.524×10^{-4}	2.255×10^{-3}	49648	1.225×10^{-4}
25	2.414×10^{-3}	73734	0.890×10^{-4}	2.436×10^{-3}	267452	1.113×10^{-4}

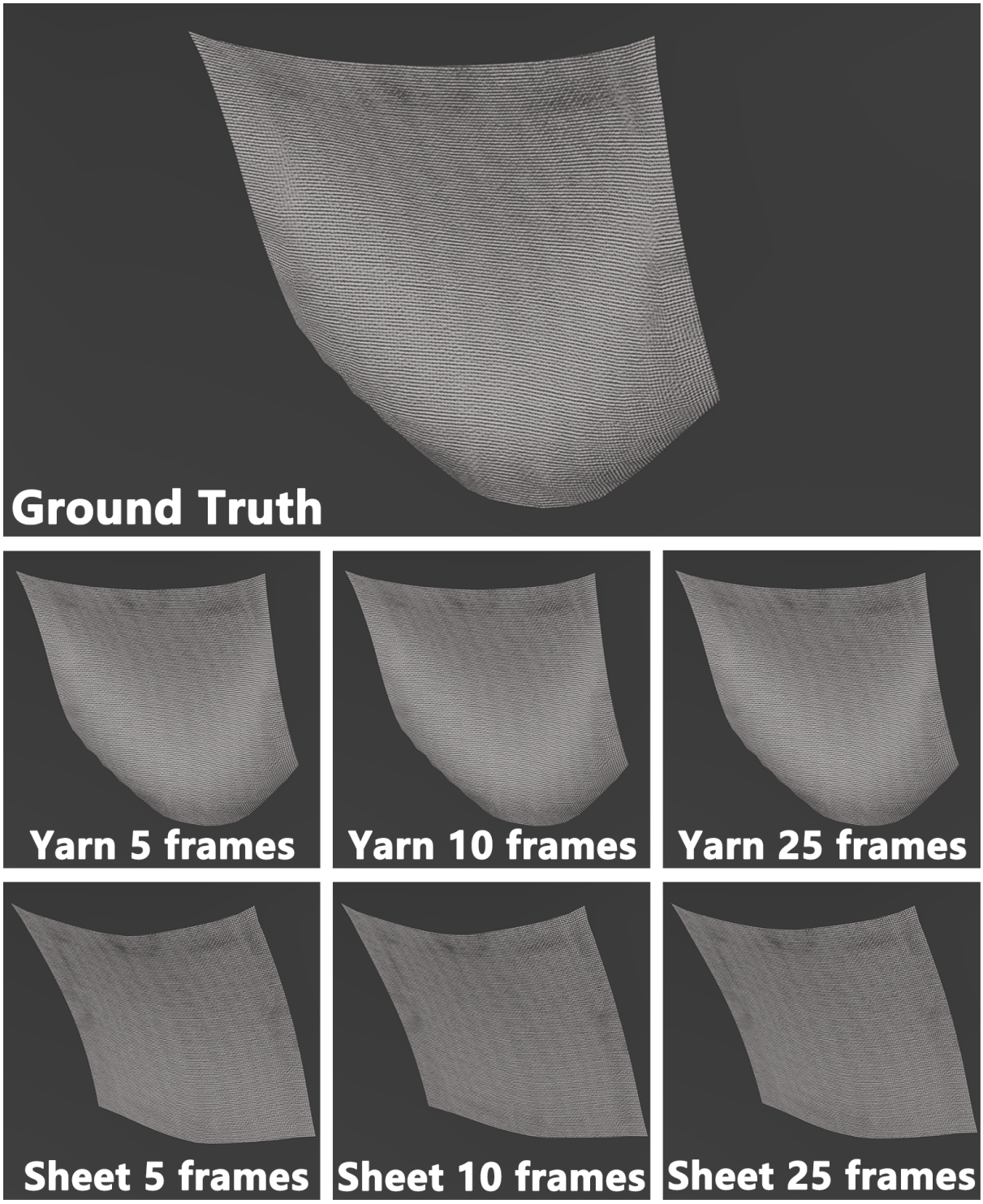


Figure A.5: The visual results of Plain-(1, 2) ground-truth, our model, and sheet-level model trained with different number of frames. The snapshots are the $133th$ frame of the simulations after learning.

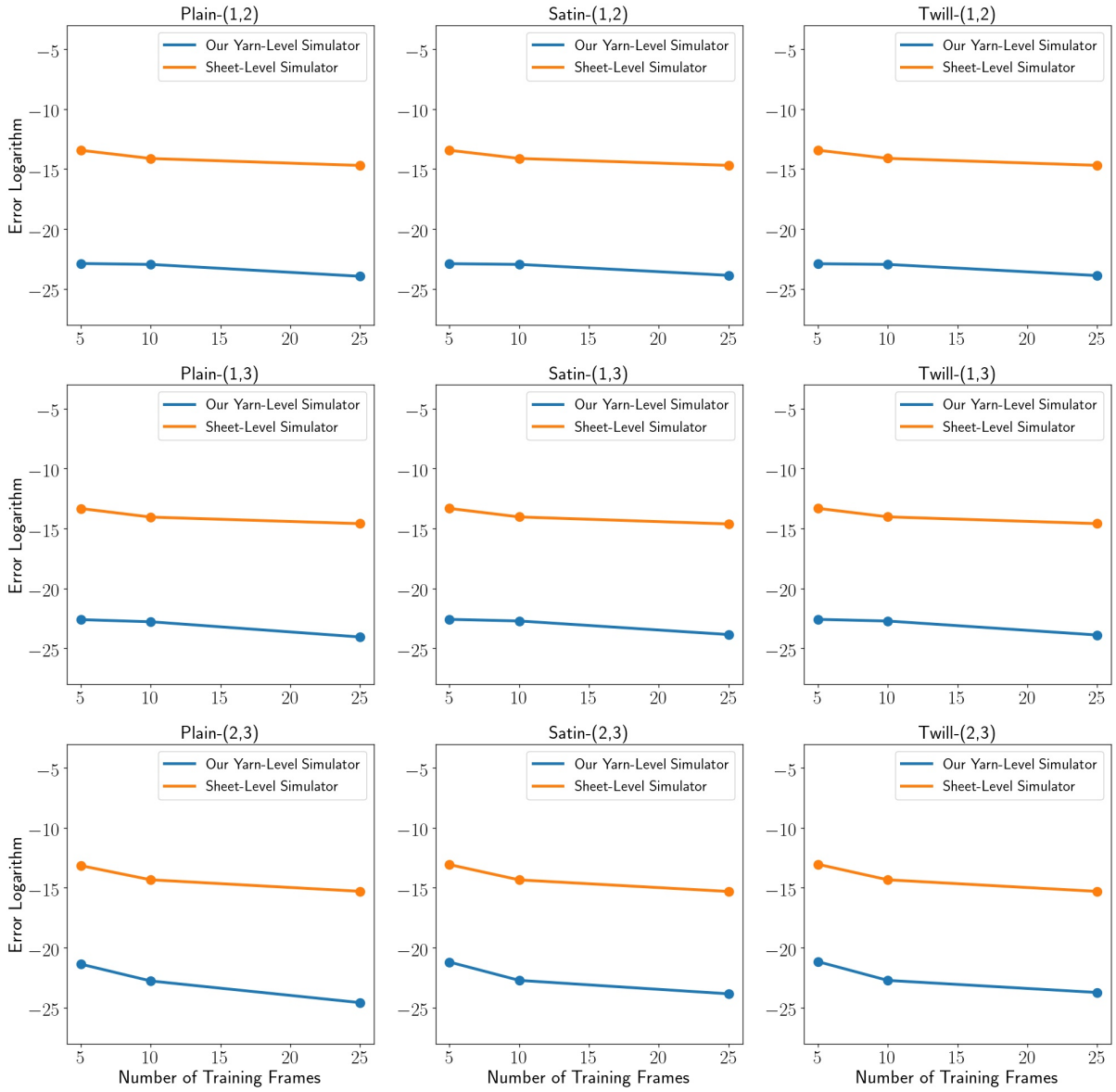


Figure A.6: Prediction error logarithm vs training data.

We exhaustively conduct comparisons using all combinations of yarns and woven patterns. We can see that our model is consistently better than the model of Liang et al. (2019) by large margins. Visually, we show snapshots in Fig. A.5. The sheet model results are in general more rigid and do not contain as much subtle dynamics as ours do, across different training frame numbers. Since 5, 10 and 25 frames contain different amounts of information on (subtle) motion dynamics, Fig. A.5 shows that there is a lack of granularity in the sheet model when capturing subtle dynamics compared with ours.

Further, we also show the plots on the data efficiency in Fig. A.6, under all 9 yarn-woven pattern combinations, across different amounts of training data. In all settings, our data efficiency is

significantly higher. By extrapolation, it would take a large number of extra training frames for the sheet-level model to achieve similar accuracy. More comparisons are also available in the supplementary video.

A.4 Our model versus Bayesian Optimization

Table A.8 shows the testing errors of the Bayesian Optimization. Although the MSE errors are small, the learned parameters are far from the ground truth (shown in the Table A.7), which is somewhat surprising. After examining the results, we find that Bayesian Optimization suffers from the multi-solution problem so that it merely gives a set of working parameters instead of the true parameters. In other words, although the prediction error is low, physically speaking, the learned parameters are far from the true materials. This happens even when we use the same parameter ranges as in our model. This is an intrinsic property of Bayesian optimization which is based on sampling, and therefore difficult to avoid during learning.

Table A.8: Testing error ($\times 10^{-6}$) of Bayesian Optimization with yarn-level simulator (Cirio et al. 2014) learned on 5, 10, and 25 frames.

Fabrics/Frames	5	10	25
Plain-(1,2)	0.512	0.176	0.109
Plain-(1,3)	1.280	1.269	0.738
Plain-(2,3)	28.19	19.22	18.16

A.5 Control Experiment Setting

The control experiment scenario is illustrated in the Fig. A.7.

A.6 Significant Error in Visual

We discussed the significance of the small error in physics-based simulation. Fig. A.8 and Fig. A.9 visually prove our explanations in the main paper: the error accumulates over time and increases with increasing cloth size.

Figure A.7: A square cloth is thrown from the table into the black box by four forces applied on the four corners of the cloth.

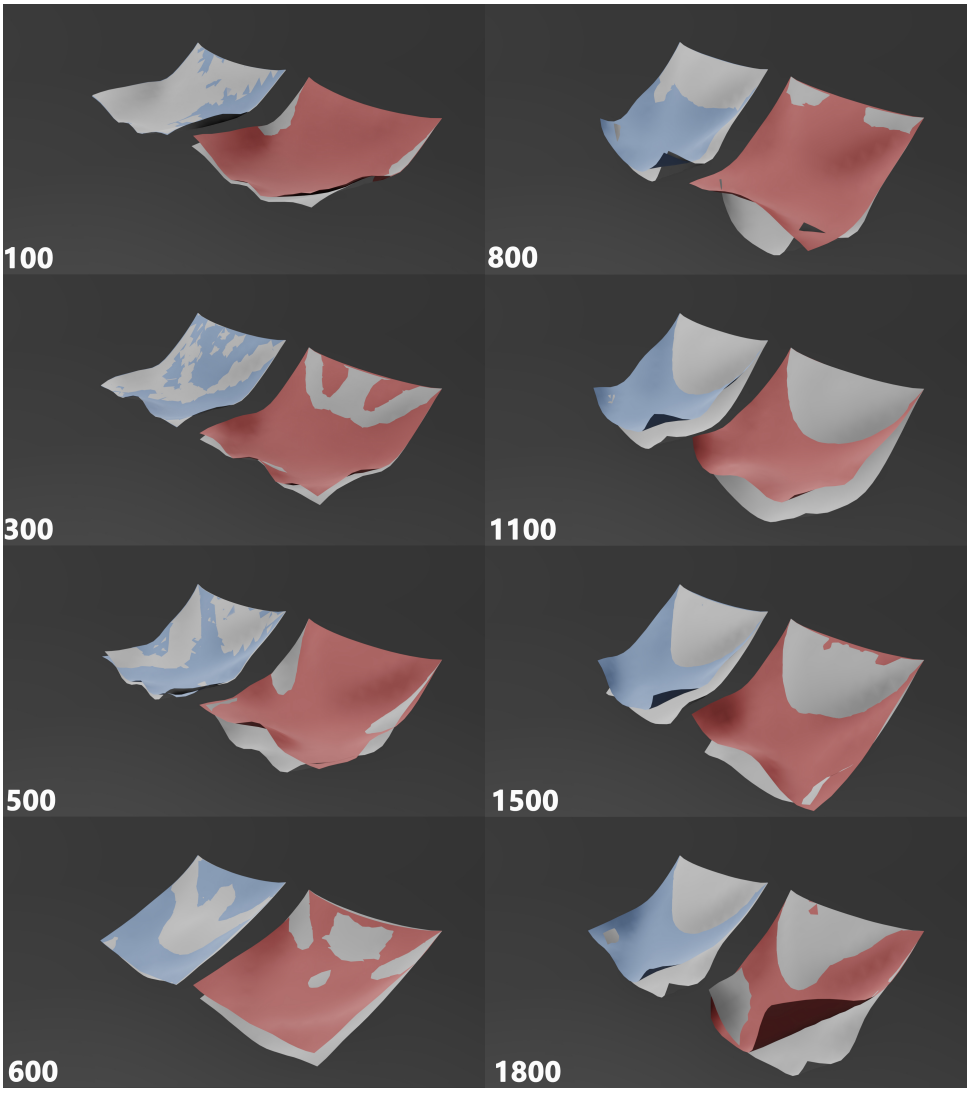


Figure A.8: Visual differences in long simulations. The grey cloth is ground truth. The blue cloth and the red cloth are simulated with the parameters learned by our model and BO. The blue cloth shows smaller visual differences than the red one.

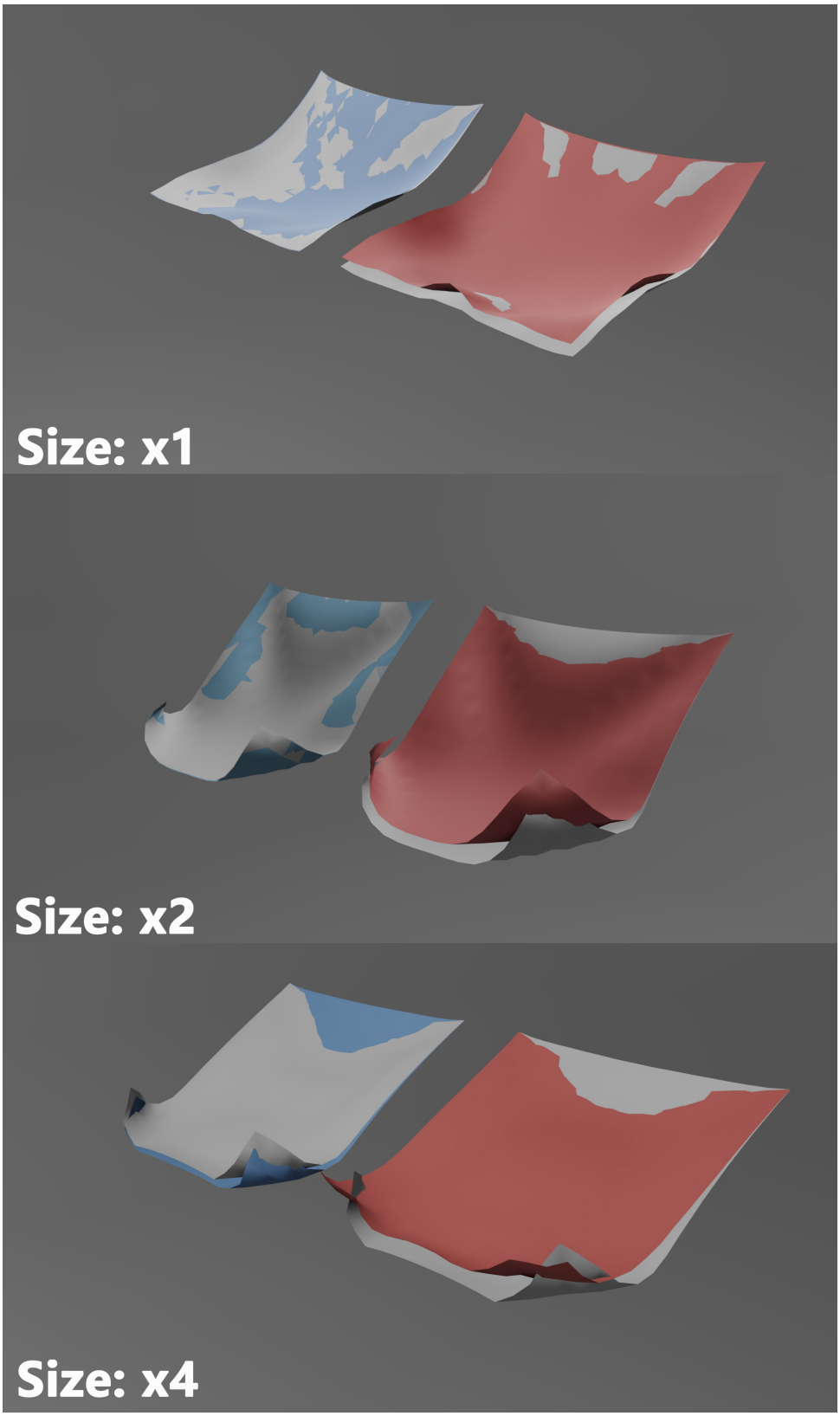


Figure A.9: Visual differences on larger cloths and long simulation (500 steps). The grey cloth is ground truth. The blue cloth and the red cloth are simulated with the parameters learned by our model and BO. The blue cloth shows smaller differences than the red one.

Appendix B

Appendix Two: Bayesian Differentiable Physics for Fabric Parameter Estimation

B.1 Cusick Drape Dataset

Our current Cusick drape dataset includes 25 types of common fabrics, each of which with multiple samples. Table B.1 lists these fabrics' material, woven pattern, area density (multiple samples), and average thickness (multiple samples), which are obviously different. In our Cusick drape test, they also show distinctive drape shapes. We test the two sides of each sample twice. In every test, our Cusick drape meter captures a drape image and reconstructs its 3D mesh. Therefore, there are 660 drape images and meshes in our current dataset. Fabric 1-5 are used in our experiments which are the Cotton White, Cotton Blue, Viscose White, Cotton Pink, and Wool Red respectively.

Table B.1: Fabric information in our Cusick drape dataset. Fabric 18 and Fabric 19 are double-layer fabrics whose one side is plain woven and the other side is knit.

Fabric Index	Material	Woven	# Samples	Avg ρ (kg/m^2)	Avg Thickness(mm)
Fabric 1	Cotton	Plain	12	0.059	0.188
Fabric 2	Cotton	Basket	12	0.192	0.402
Fabric 3	Viscose(95%) Elastane (5%)	Knit	12	0.213	0.560
Fabric 4	Cotton	Plain	12	0.114	0.200
Fabric 5	Wool	Twill	12	0.274	0.571
Fabric 6	Polyester	Satin	12	0.183	0.240
Fabric 7	Polyester(65%) Cotton(35%)	Plain	12	0.100	0.195
Fabric 8	Linen	Plain	12	0.230	0.485
Fabric 9	Cotton	Plain	12	0.249	0.423
Fabric 10	Viscose (70%) Polyester (30%)	Plain	12	0.204	0.498
Fabric 11	Wool	Twill	2	0.148	0.210
Fabric 12	Cotton	Plain	2	0.313	0.188
Fabric 13	Cotton	Plain	2	0.107	0.15
Fabric 14	Cotton	Plain	2	0.139	0.292
Fabric 15	Cotton	Plain	2	0.211	0.436
Fabric 16	Synthetic	Knit	2	0.191	0.514
Fabric 17	Cotton	Twill	2	0.278	0.7
Fabric 18	Synthetic	Plain/Knit	2	0.154	0.306
Fabric 19	Synthetic	Plain/Knit	2	0.282	0.627
Fabric 20	Synthetic	Twill	2	0.259	0.596
Fabric 21	Synthetic	Knit	5	0.186	0.7
Fabric 22	Synthetic	Plain	5	0.231	0.386
Fabric 23	Cotton	Knit	5	0.163	0.704
Fabric 24	Cotton	Plain	5	0.104	0.200
Fabric 25	Cotton	Plain	5	0.060	0.152