DOI: 10.1111/cgf.13915
EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)
Volume 39 (2020), Number 2

Mixing Yarns and Triangles in Cloth Simulation

Juan J. Casafranca[1] Gabriel Cirio[2] Alejandro Rodríguez[2] Eder Miguel[2] Miguel A. Otaduy[1]

[1]Universidad Rey Juan Carlos, Madrid, Spain
[2]Seddi Labs, Madrid, Spain

Figure 1: *Mixing yarns and triangles to efficiently predict fitting details of a shirt. From left to right: simulation result with a full-yarn model, a triangle-only model, and our method, and mixed discretization with triangles and yarns. The triangle-based model misses all the detailed wrinkling of the yarn-based model. We select a region of interest on this triangle-based model, we enrich it with yarns, and the mixed result reproduces the wrinkles of the full model. Wrinkles outside the region of interest are not captured, as expected.*

Abstract

This paper presents a method to combine triangle and yarn models in cloth simulation, and hence leverage their best features. The majority of a garment uses a triangle-based model, which reduces the overall computational and memory cost. Key areas of the garment use a yarn-based model, which elicits rich effects such as structural nonlinearity and plasticity. To combine both models in a seamless and robust manner, we solve two major technical challenges. We propose an enriched kinematic representation that augments triangle-based deformations with yarn-level details. Naïve enrichment suffers from kinematic redundancy, but we devise an optimal kinematic filter that allows a smooth transition between triangle and yarn models. We also introduce a preconditioner that resolves the poor conditioning produced by the extremely different inertia of triangle and yarn nodes. This preconditioner deals effectively with rank deficiency introduced by the kinematic filter. We demonstrate that mixed yarns and triangles succeed to efficiently capture rich effects in garment fit and drape.

CCS Concepts

• ***Computing methodologies*** → ***Physical simulation;***

1. Introduction

Yarn-level models [KJM08, CLMMO14] have emerged as a tool for the simulation of cloth with extreme realism and detail. They reproduce, by construction, the structural nonlinearity and anisotropy of fabrics, both woven and knitted. However, the benefits of yarn-level models come at a high price, as their simulation is extremely computationally expensive, purely due to the high dimensionality of the models. To capture interesting yarn-level effects, the models must be discretized at the resolution of yarn crossings, which, for common garments, are typically in the order of millions.

Instead, the traditional approach to cloth simulation is to consider cloth as a continuous surface, and discretize it using triangles. The power of triangle-based cloth simulation is that the discretization can be adapted to the computational needs, while retaining the large-scale behavior of cloth. This is particularly true with remeshing techniques [NSO12], which place degrees of freedom at regions of interest. However, triangle-based simulation misses the accuracy and expressiveness of yarn-level models. Nonlinearity, anisotropy, and internal friction require complex constitutive material models [WOR11, MBT*12, MTB*13], which are difficult to parameterize and hardly reach comparable results. Furthermore, yarn-level plastic effects, such as snags, are just nearly impossible with triangle-based discretizations alone.

In this paper, we present a computational solution to mix yarns and triangles for cloth simulation. Our approach allows us to exploit

the benefits of both worlds: the computational efficiency of triangles at regions with smooth deformation, and the accuracy and expressiveness of yarns at regions with high detail. Our mixed representation supports efficient triangle-based computations combined with high-resolution yarn-level effects.

To combine yarns and triangles in a robust and seamless manner, we propose an enriched kinematic representation, described in Section 3. In this representation, the base triangle-level motion is augmented with yarn-level displacements. We enforce uniqueness of the representation thanks to a kinematic filter, which is designed to provide an optimal coupling between yarns and triangles.

Following the enriched kinematics, we design enriched dynamics that smoothly blend yarn-level and triangle-level mechanics. The dynamics formulation, described in Section 4, is derived in a sound manner, and handles efficiently the kinematic constraints introduced by the filter.

However, the solution to the enriched dynamics problem exhibits two notable challenges: rank deficiency produced by the kinematic filter, and very poor conditioning produced by the combination of radically different discretizations. In Section 5, we describe an efficient preconditioner that addresses both challenges.

We show multiple examples where we enrich triangle-based cloth simulations with yarns at the resolution of real-world fabrics. While the full yarn simulations would be hardly tractable, with our mixed solution we reproduce yarn-level wrinkles and plasticity. We demonstrate the effectiveness of our method to simulate the fit and drape of garments while retaining yarn-level detail at regions of interest, as shown in Fig. 1.

2. Related Work

Over ten years ago, Kaldor et al. [KJM08] introduced a yarn-level model for the simulation of full garments in computer graphics. Their work demonstrated that complex nonlinear and anisotropic behaviors of garments arise naturally when mechanics are modeled according to the construction structure of the fabrics. They modeled yarns as deformable splines in contact, and the cost of the simulation was dominated by inter-yarn contact handling. Later, Kaldor et al. [KJM10] improved the efficiency of the model, applying careful approximations at contact areas with high temporal coherence. Despite the improvements, the model was still limited by the high density of contacts, with the need to impose small time steps to guarantee robust and accurate contact handling. Due to these limitations, the work was demonstrated on thick-yarn knits.

Cirio et al. [CLMMO14] introduced an alternative yarn-level model, under the assumption that yarns are in persistent contact. They discretized fabrics at yarn crossings, with a single Lagrangian node representing the position of both crossing yarns, plus two Eulerian coordinates to capture yarn sliding. Later, Cirio et al. [CLMO17] extended their model to handle both knitted and woven fabrics under a common representation. The persistent contact model is more robust and allows larger time steps; as a result, Cirio et al. demonstrated the model on cloth with as many as one million crossing nodes. However, this is still far from the density of real-world garments, which could reach or surpass ten million yarn crossings. At the same time, it is hard to justify the need for yarn-scale modeling everywhere on a garment; therefore, simulation cost may be drastically reduced by using a yarn-scale model only at regions of interest. To date, no work showed this.

Yarn-based models have seen application in computer graphics for the design of fabrics and fabrication processes, in particular for knits. Leaf et al. [LWS*18] extended the model of Kaldor et al. [KJM08] with support for periodic boundary conditions, and thus simulate the relaxation of knit patches to aid the design of knit patterns. Yuksel et al. [YKJM12] created a *stitch mesh* data structure that simplifies the generation of knit garments as a tiling procedure, and Wu et al. [WSY19] later extended it to fulfill fabrication constraints. Narayanan et al. [NWYM19] extended the stitch mesh data structure even further, to augment it with fabrication operations, and thus use it in the context of fabrication-oriented editing.

The simulation of cloth at yarn scale could benefit from fast solvers for high-resolution discretizations. These include GPU-based [TTN*13] and multigrid solutions [TJM15]. But yarn-based models provide two benefits: one is very high resolution and hence the ability to reproduce high level of detail, and the other is the inherent ability to capture structural nonlinearity. Adaptive simulation methods offer an alternative to obtain high level of detail in a cost-effective manner. They are complementary to our work, and can in principle be adopted in the triangle-based portion of our method. Extended discussion is provided in a recent survey on adaptive methods [MWN*17]. A different approach to achieve cost-effective simulations is to track geometric details in a Lagrangian manner and resolve complex collisions on an Eulerian grid, as done following the Material Point Method [JGT17].

In recent last years, the ARCSim adaptive remeshing method has reached great success [NSO12]. It defines a sizing field based on local curvature, compression and velocity, and determines the resolution of mesh edges dynamically based on the local value of the field. The sizing criterion offers a compelling heuristic, while remeshing operations take only a fraction of the simulation cost. ARCSim has been extended to support folding and crumpling [NPO13], and tearing and cracking [PNdJO14].

Mesh refinement is one way to increase the resolution of a model, the other is to enrich the basis functions that guide the discretization. Grinspun et al. [GKS02] introduced an adaptive simulation framework using hierarchical basis functions, and applied it to cloth simulation among other use cases. The concept of enrichment can also be viewed in more general terms, as defining a set of coordinates that capture separately coarse motion and local displacements, and solving a dynamics problem on the combined representation. In terms of methodology, our work is most similar to such enrichment simulation methods in computer graphics.

Enrichment has been applied in settings as diverse as adding local detail to reduced simulations [HZ13], combining Eulerian simulations with several Lagrangian modes [FLLP13], or overlapping arbitrarily different simulation domains [MGL*15].

One important aspect in enriched representations is uniqueness of the basis, i.e., a full-space motion should admit only one valid representation in the basis. In finite element simulation, enrichment is achieved by enforcing properties on the basis functions, e.g.,

orthogonality. However, in our setting this is far from trivial, as we combine two radically different representations. In this regard, our work shares challenges with Eulerian-on-Lagrangian simulation [FLLP13] and multifarious hierachies [MGL*15]. In Eulerian-on-Lagrangian simulation, the basis is not unique by construction, but uniqueness is enforced on every simulation step, by first solving an optimization problem on Lagrangian modes, and then solving the residual problem on Eulerian degrees of freedom. In multifarious hierarchies, uniqueness was enforced by imposing linear constraints on the velocities of the detailed simulation domain, and the authors discussed ways to choose the constraints to decouple the solution of the coarse and detailed domains. In our representation, however, it is not possible to decouple the solution of triangle and yarn degrees of freedom, because we blend energy terms that do not affect yarn and triangle degrees of freedom equally. Moreover, we enforce uniqueness thanks to carefully designed constraints, to optimize the blending of yarn and triangle regions.

3. Enriched Cloth Kinematics

To allow a smooth transition between yarns and triangles in cloth simulation, we devise an enriched kinematic representation, where detailed yarn displacements are added to a base triangle motion. We start this section describing the enriched representation. However, in its straightforward form, the kinematic representation is not unique; it requires additional constraints to ensure uniqueness. We formulate constraints such that the base triangle motion is an optimal match to the full yarn motion, and we implement these constraints efficiently through a kinematic filter on yarn displacements.

3.1. Mixed Kinematic Representation

Given a patch of cloth made of yarns, either woven or knitted, we represent its dynamic behavior at the yarn level. Later in Section 4, we will address the ability to transition patches represented at yarn and triangle level, but our kinematic enrichment focuses on yarn-based patches. To capture the kinematics of yarns, we use an enriched kinematic representation: a triangle mesh represents the base motion, while a yarn-based discretization represents detailed motion. Our representation supports arbitrary methods for both the triangle and yarn-level models, but for yarns we choose the persistent contact model of Cirio et al. [CLMMO14]. The model of Cirio et al. combines Lagrangian and Eulerian coordinates, but our enriched kinematic representation is limited to the Lagrangian part of the motion. We use Eulerian coordinates in full-yarn patches of cloth; therefore, they do not require special treatment. As a preprocess, we mesh the cloth using triangles, and for each yarn node we detect its container triangle and compute its barycentric coordinates.

We denote as $\mathbf{v}_j \in \mathbb{R}^3$ the position of a triangle vertex, $\mathbf{x}_i \in \mathbb{R}^3$ the displacement of a yarn node relative to its container triangle, and $\beta_{i,j}$ the barycentric coordinate of the i^{th} yarn node w.r.t. the j^{th} triangle vertex. To obtain the world position $\mathbf{p}_i \in \mathbb{R}^3$ of a yarn node, we simply compute its base position through barycentric interpolation of the vertex positions, and we add the detail displacement:

$$\mathbf{p}_i = \sum_{j \in \text{tri}(i)} \beta_{i,j}\, \mathbf{v}_j + \mathbf{x}_i. \tag{1}$$

We make two important observations about the computation of the world position of a yarn node. First, the representation is not unique; infinite combinations of triangle and yarn coordinate values produce the same world position. The kinematic filter presented in the next section addresses precisely this problem. Second, we choose to represent detail displacements in the global frame. This is in contrast to other enrichment representations [MGL*15], which choose a local frame defined by the base motion. A local frame has the advantage that detail displacements are invariant to base motion. Our choice of global frame, on the other hand, has the advantage that the kinematic filter becomes constant, and this is highly relevant for the computational efficiency of our method.

3.2. Kinematic Filter

To remove the redundancy in the kinematic representation, we must add some constraints. One possibility is to define constraint equations that relate yarn displacements to triangle vertex positions, and later derive the equations of motion following a constrained dynamics formulation. Instead, we impose constraints through a filter matrix [BW98, AB03], and thus we obtain non-redundant coordinates and we derive the equations of motion for unconstrained dynamics. We defer the derivation of the equations of motion to Section 4, and here we focus on the definition of the kinematic filter.

We group triangle vertex positions in a vector $\mathbf{v}$, detailed yarn displacements in a vector $\mathbf{x}$, and full yarn positions in a vector $\mathbf{p}$. We also define a matrix $\mathbf{B}$ that represents the basis for the base motion. It stores barycentric weights, and transforms triangle positions into the base component of world yarn positions according to (1).

Based on these definitions, we define the full yarn positions of a patch of cloth as a function of enriched kinematics:

$$\mathbf{p} = \mathbf{B}\,\mathbf{v} + \mathbf{F}\,\mathbf{x}, \tag{2}$$

where $\mathbf{F}$ represents a kinematic filter that removes redundancy and makes the representation unique. It must be a low-rank matrix that removes from $\mathbf{x}$ as many degrees of freedom as those present in $\mathbf{v}$. There are infinite choices for $\mathbf{F}$, which result in turn in infinite combinations of $\mathbf{v}$ and $\mathbf{x}$ to represent the same $\mathbf{p}$.

We seek one particular filter $\mathbf{F}$, such that enriched cloth patches can be seamlessly coupled to patches represented using only triangles. To do so, we pose the condition that the base triangle motion $\mathbf{v}$ should match the full yarn-level motion $\mathbf{p}$ as closely as possible. This is achieved by making $\mathbf{F}\,\mathbf{x}$ orthogonal to $\mathbf{B}\,\mathbf{v}$. Therefore, the filter $\mathbf{F}$ must be the null-space projection of $\mathbf{B}$:

$$\mathbf{F} = \mathbf{I} - \mathbf{B}\left(\mathbf{B}^T\,\mathbf{B}\right)^{-1}\mathbf{B}^T. \tag{3}$$

It is evident that $\mathbf{B}^T\,\mathbf{F} = 0$.

The filter subtracts from identity a low-rank projection, of size $3\times$ the number of triangle vertices in the enriched patch, in contrast to the large number of yarn nodes. Given some (unfiltered) yarn-level vector, the filter is never explicitly computed, and it acts as follows: (i) it projects the vector to the triangle domain using barycentric weights, (ii) computes the best-matching triangle-level vector by solving a linear system with matrix $\mathbf{B}^T\,\mathbf{B}$, (iii) propagates the result to the yarns using barycentric weights, and (iv) subtracts

this result from the original vector. As a result, it yields the portion of the vector that cannot be represented using the triangles. The matrix $\mathbf{B}^T\mathbf{B}$ is constant, sparse, and small, of size $3\times$ the number of triangle vertices in the enriched patch, and we precompute its Cholesky factorization. Moreover, the filter can be applied independently for each Cartesian coordinate. By leveraging this property, in practice we downsize the matrix by a factor of 9.

Our kinematic filter is used for the derivation of the equations of motion according to unconstrained dynamics, as described next. At runtime, it is used every time that a yarn-level vector is computed, to filter out components that can be captured at triangle level.

4. Enriched Dynamics

The enriched kinematic representation described in the previous section provides a mechanism to smoothly couple yarn and triangle models in cloth simulation. In this section, we define the yarn and triangle models and their transition, and we leverage the enriched representation to derive the equations of motion in a sound manner.

4.1. Coupling Yarns and Triangles

We consider a cloth object divided into three regions. We define these three regions thanks to a blending field $\alpha \in [0,1]$, such that $\alpha = 0$ indicates a triangle-only region, $\alpha = 1$ a full-yarn region, and otherwise a hybrid region. Our enriched kinematic representation is applied on the hybrid region, and enables a smooth transition between yarns and triangles.

Most importantly, we also use the blending field to smoothly transition between yarn-based and triangle-based mechanics. Without loss of generality, at every point on the cloth object, and for every type of mechanical energy (i.e., kinetic, elastic, and gravitational), we define two energy density functions at different resolution: a yarn-based energy Ψ_p is defined on full yarn positions $\mathbf{p}$, and a triangle-based energy Ψ_v is defined on triangle positions $\mathbf{v}$, which, by definition of the kinematic filter, are an optimal match to full yarn positions. Conceptually, we blend the energies as:

$$\Psi = \alpha\,\Psi_p + (1-\alpha)\,\Psi_v. \tag{4}$$

In practice, each energy density function is integrated on its corresponding discretization (yarns or triangles), weighted by its corresponding blending weight.

It follows naturally that, at locations where $\alpha = 0$, the kinematics and mechanics of the cloth can be represented using only triangles. Similarly, at locations where $\alpha = 1$, the kinematics and mechanics of the cloth can be represented using only yarns. Elsewhere, we use our enriched kinematics representation, and triangle and yarn energies are blended. Note that, in the blending or hybrid region, and due to the enriched kinematics, yarn energies depend on both triangle positions and yarn displacements, hence the triangles in the hybrid region contribute to both energies.

In our implementation, we have chosen the Saint Venant-Kirchhoff elasticity [Ogd97] and the discrete shells model [GHDS03] for the in-plane and out-of-plane mechanics of triangles, respectively. Following the choice of the persistent contact discretization of yarns, we have chosen corresponding elastic models for woven and knitted fabrics [CLMO17]. In practice, we weight the various discrete energy terms by the values of the blending field α at the corresponding element centroids.

4.2. Equations of Motion

Thanks to the kinematic filter, we can form a set of generalized coordinates consisting of triangle coordinates $\mathbf{v}$ and (filtered) yarn displacements $\mathbf{x}$, i.e., $\mathbf{q} = \begin{pmatrix} \mathbf{v} \\ \mathbf{x} \end{pmatrix}$. Then, we can apply the Euler-Lagrange equations to obtain the equations of motion [GPS01]. The resulting equations can be regarded as a filtered version of the full redundant equations.

To apply the Euler-Lagrange equations, we must first compute the total kinetic and potential energy of a cloth object. We do so by integrating energy densities over triangles and yarns. In the hybrid region, where both triangles and yarns coexist, we compute all energies at both triangle and yarn level, and we blend the resulting energies following the coupling scheme described in Section 4.1 above. Recall that triangle-based energies depend only on triangle kinematics, but yarn-based energies depend on both yarn and triangle kinematics, due to the enriched kinematics in the hybrid region.

To define the total kinetic energy, we consider mass matrices $\mathbf{M}_v$ and $\mathbf{M}_p$, integrated, respectively, on the triangles and the yarns. In the hybrid region, we compute mass terms as follows. We first define mass on yarns. For each yarn segment, we split its mass into a "yarn mass" and a "triangle mass" based on the local value of the blending weight. We use the "yarn mass" to compute the yarn-based kinetic energy, and hence the mass terms that contribute to $\mathbf{M}_p$. We accumulate the "triangle mass" on triangles, and we lump it on vertices to compute the triangle-based kinetic energy, and hence the mass terms that contribute to $\mathbf{M}_v$. Based on the mass matrices $\mathbf{M}_v$ and $\mathbf{M}_p$, we express the total kinetic energy as:

$$T = T_v + T_p = \frac{1}{2}\,\dot{\mathbf{v}}^T\,\mathbf{M}_v\,\dot{\mathbf{v}} + \frac{1}{2}\,\dot{\mathbf{p}}^T\,\mathbf{M}_p\,\dot{\mathbf{p}}. \tag{5}$$

We rewrite the kinetic energy as a function of the generalized coordinates, using the kinematic relationship (2), to obtain:

$$T = \frac{1}{2}\,\dot{\mathbf{q}}^T\,\mathbf{S}\mathbf{M}\mathbf{S}\,\dot{\mathbf{q}}, \tag{6}$$

$$\text{with generalized filter } \mathbf{S} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{pmatrix}, \tag{7}$$

$$\text{and mass matrix } \mathbf{M} = \begin{pmatrix} \mathbf{M}_v + \mathbf{B}^T\mathbf{M}_p\mathbf{B} & \mathbf{B}^T\mathbf{M}_p \\ \mathbf{M}_p\mathbf{B} & \mathbf{M}_p \end{pmatrix}. \tag{8}$$

Since the computation of mass and kinetic energy in the hybrid region starts from well-defined yarn masses, these quantities are computed correctly.

To define the potential energy, we integrate energy densities on triangles and yarns, and obtain two added energy terms V_v and V_p. From the total potential energy, we obtain the conservative forces on the generalized coordinates. These can be expressed as a function of triangle-only and full-yarn forces $\mathbf{f}_v = -\nabla_v V_v$ and $\mathbf{f}_p = -\nabla_p V_p$:

$$-\nabla_{\mathbf{q}} V = \mathbf{S}\mathbf{f}, \text{ with } \mathbf{f} = \begin{pmatrix} \mathbf{f}_v + \mathbf{B}^T\mathbf{f}_p \\ \mathbf{f}_p \end{pmatrix}. \tag{9}$$

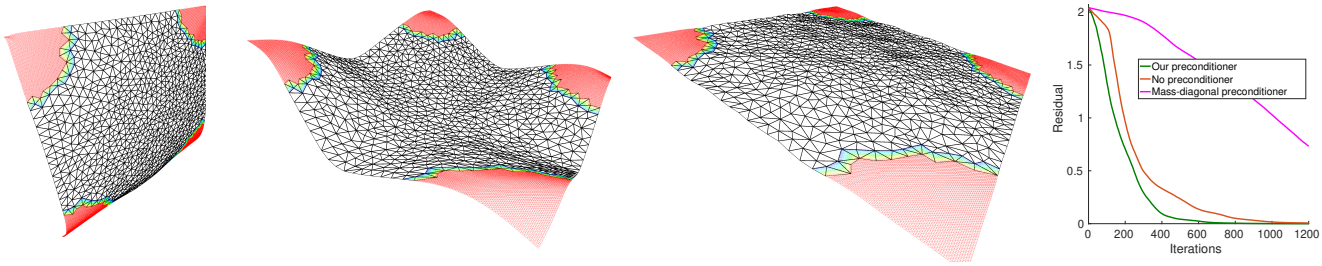

Figure 2: *From left to right: (i) Simulation of a swinging cloth with our preconditioner; (ii) with no preconditioner, yarns lag behind triangles, as they accumulate error; (iii) a naïve mass-diagonal preconditioner does not help, as it ignores the effect of the kinematic filter. In all three cases, we used 100 iterations per MPCG solve. (iv) Comparison of convergence for one MPCG solve.*

We trivially differentiate (2) to obtain the Jacobian of full-yarn positions with respect to generalized coordinates, and we use the transpose of this Jacobian to distribute the forces on full yarn positions, $\mathbf{f}_p$, to the generalized coordinates. They are distributed to the triangles through the barycentric coordinates, and they are distributed to the yarn displacements after being filtered.

We can now express the Euler-Lagrange equations compactly:

$$\mathbf{S}\mathbf{M}\ddot{\mathbf{q}} = \mathbf{S}\mathbf{f}. \tag{10}$$

In practice, we add dissipative forces to $\mathbf{f}$ using the Rayleigh damping model. The yarn-level mass matrix $\mathbf{M}_p$ varies with time due to yarn sliding in the model of Cirio et al. [CLMMO14]. However, we discard its time derivative, which is equivalent to considering it constant within each time step. Finally, note that the expression uses filtered generalized accelerations, i.e., $\mathbf{S}\ddot{\mathbf{q}} = \ddot{\mathbf{q}}$.

4.3. Dynamics Solver

We discretize the equations of motion using backward Euler numerical integration with time step h, and we solve the resulting nonlinear equations using Newton's method. Each Newton iteration involves solving a linear system of the form:

$$\mathbf{S}\left(\mathbf{M} - h\frac{\partial \mathbf{f}}{\partial \dot{\mathbf{q}}} - h^2\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)\Delta\dot{\mathbf{q}} = \mathbf{S}\mathbf{b}. \tag{11}$$

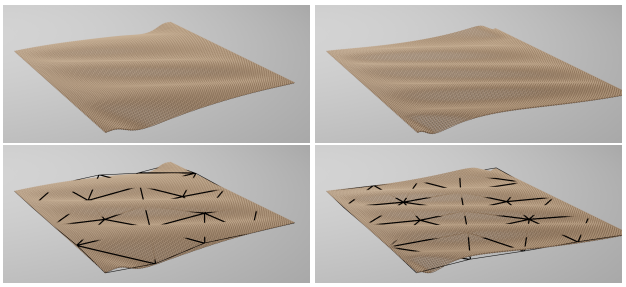

Figure 3: *Our enriched model represents full yarn dynamics exactly. This validation example shows, on top a full yarn simulation, and on the bottom an enriched simulation (with coarse triangles in wireframe). Small differences may appear over time due to differences in the linear system and the convergence of the solver.*

The resulting system matrix is not full-rank, due to the filter matrix $\mathbf{S}$. However, this type of constrained linear system can be solved efficiently through a minor modification to the conjugate gradient solver, by applying the filter to the search direction on each iteration of the solver [BW98, AB03]. This solver is referred to as *modified preconditioned conjugate gradient* (MPCG).

In Fig. 3, we compare simulations of a patch of cloth modeled with our enriched representation vs. yarns. We shear the patch by pulling from two opposite sides, producing high tension as well as wrinkles on the compressed direction. We validate that the enriched representation captures the same motion as the full-yarn model. Note that the exact solution to the nonlinear dynamics is the same in both cases, but the solver of the linear system introduces subtle differences over time. In this example, the enriched representation imposes an unnecessary computational overhead. The enriched representation plays its role when mixing yarns and triangles in one cloth object. However, we found that the straightforward solution to (11) behaved poorly. In the next section we propose a solution based on a carefully designed preconditioner.

5. Preconditioner for Enriched Systems

The linear system for the solution of enriched dynamics shown in the previous section exhibits two notorious challenges. (i) It combines elements of very different resolution (triangles and yarns), which produces poor conditioning and residuals of very different scale. (ii) It is rank-deficient, due to the filter that removes the redundancy in the kinematics. We design a preconditioner that addresses the issues induced by differences in resolution, while responding to the challenge of rank deficiency. We start this section providing further insight into the challenges, and we follow with a detailed description of our preconditioner.

5.1. Problem Statement

To minimize the computational cost of a cloth simulation, triangles are considerably larger than the inter-yarn distance. In our examples, we typically mesh one triangle on top of hundreds of yarn nodes. As a result, the mass density of triangle vertices is several orders of magnitude larger than the mass density of yarn nodes. The difference is even more acute if we consider yarn nodes at the interface of the hybrid region with the triangle-only region. Their

kinetic energy is scaled by a very low blending weight α, hence their mass is also scaled down notably. To partially alleviate this problem, in our implementation we clamp small non-zero weights $\alpha \in (0, 0.01]$ to a value of 0.01.

The residual of the linear system (11) scales errors in velocity by the system matrix. The visual impact of errors in the velocities of triangle or yarn coordinates is effectively similar; however, errors in the velocities of triangle coordinates are scaled by much larger mass values. Consequently, the residual evaluation of the MPCG solver is strongly biased by the error on triangle velocities, and the search direction is not effective against visual error.

We have found that the disparity of mass densities has a devastating effect on the convergence of the MPCG solver, as evidenced in Fig. 2. Interestingly, this challenge is hardly discussed in the computer graphics simulation community, possibly because computer graphics simulations do not exhibit elements with radically different resolutions. Our solution strategy consists of multiplying the system matrix in (11) by the inverse of the (filtered) generalized mass matrix. This constitutes effectively the application of a preconditioner in the MPCG solve. Moreover, an added benefit of this preconditioner is that the residual evaluates errors in velocities, not mass-scaled velocities. Unfortunately, the filtered generalized mass matrix is rank deficient, hence the design of a preconditioner is not a simple task. As a final remark, note that we have focused our attention on the effect of the mass matrix in (11), discarding other terms such as the stiffness matrix. We found this to be successful in practice, but extending the analysis to the full system matrix is an interesting direction for future work.

5.2. Preconditioner

The type of constrained linear problem in (11) was thoroughly studied by Boxerman [Box03], with the difference that he limited his examples to filter matrices resulting from contact constraints. As he argued, a preconditioner for a constrained matrix $\mathbf{SA}$ should not be designed by approximating the unconstrained system matrix $\mathbf{A}$, as this approach ignores the effect of the filter, and then the inverse of the preconditioner fails to improve the conditioning of the constrained system. Boxerman showed that the preconditioner should approximate the augmented system $\mathbf{SA}+\mathbf{I}-\mathbf{S}$ instead. Effectively, the added term makes the system full rank, by extending it to the complementary subspace of the filter.

As discussed earlier, we wish to precondition (11) by removing the mass-scaling of the problem, and thereby evaluate the problem's residual using velocity changes. Then, we design our preconditioner as

$$\mathbf{P} = \mathbf{SM} + \mathbf{I} - \mathbf{S}. \tag{12}$$

The inverse of the preconditioner is applied on every iteration of the MPCG solver; therefore, it must be efficiently computed. We demonstrate that $\mathbf{P}$ can be expressed as a low-rank update (with rank determined by the triangles in the hybrid region) of an easily invertible matrix (with size determined by the yarn nodes in the hybrid region). Then, its inverse can be efficiently computed using the Sherman-Morrison-Woodbury formula [GVL96].

Let us rewrite the generalized filter in (7) as a low-rank update of identity:

$$\mathbf{S} = \mathbf{I} - \mathbf{U}_S \mathbf{A}_S \mathbf{U}_S^T, \tag{13}$$
$$\text{with } \mathbf{U}_S = \begin{pmatrix} \mathbf{0} \\ \mathbf{B} \end{pmatrix} \text{ and } \mathbf{A}_S = \left(\mathbf{B}^T \mathbf{B}\right)^{-1}.$$

Similarly, let us rewrite the generalized mass matrix in (8) as a low-rank update of an easily invertible block-diagonal approximation $\mathbf{M}_0$:

$$\mathbf{M} = \mathbf{M}_0 + \mathbf{U}_M \mathbf{A}_M \mathbf{U}_M^T, \text{ with } \mathbf{U}_M = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{M}_p \mathbf{B} & \mathbf{0} \end{pmatrix}, \tag{14}$$
$$\mathbf{A}_M = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{pmatrix} \text{ and } \mathbf{M}_0 = \begin{pmatrix} \mathbf{M}_v + \mathbf{B}^T \mathbf{M}_p \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_p \end{pmatrix}.$$

By substituting (13) and (14) into (12), we derive an expression of the preconditioner as a low-rank update of $\mathbf{M}_0$:

$$\mathbf{P} = \mathbf{M}_0 + \mathbf{U}_P \mathbf{A}_P \mathbf{V}_P, \text{ with } \mathbf{U}_P = \begin{pmatrix} \mathbf{U}_S & \mathbf{U}_M \end{pmatrix}, \tag{15}$$
$$\mathbf{A}_P = \begin{pmatrix} \mathbf{A}_S & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_M \end{pmatrix} \text{ and } \mathbf{V}_P = \begin{pmatrix} \mathbf{U}_S^T (\mathbf{I} - \mathbf{M}) \\ \mathbf{U}_M^T \end{pmatrix}.$$

We use the Sherman-Morrison-Woodbury formula [GVL96] to apply the preconditioner efficiently. This requires solving two linear problems. One is of full size and involves $\mathbf{M}_0$, whose structure can be seen in (14). The yarn-level block is easy to invert, and for the triangle-level block we compute a Cholesky factorization per time step. The other problem is of low rank, non-symmetric, and we compute an LU factorization. Its size is determined by the size of $\mathbf{A}_P$ in (15). However, same as for the computation of the kinematic filter in Section 3.2, we leverage that the three Cartesian coordinates can be handled independently, and downsize the matrix by a factor of 9. The effective size is then $3\times$ the number of triangle vertices in the hybrid region.

In Fig. 2 we show the effect of the preconditioner on a swinging cloth example. Without preconditioner, yarns lag behind triangles as they accumulate error. With our preconditioner, on the other hand, the residual is correctly weighted on both types of nodes, and the effect of the filter is correctly addressed. A naïve alternative, using as preconditioner the diagonal of the mass matrix, has an adverse effect.

6. Experiments and Results

6.1. Implementation Details

We model cloth objects as collections of 2D patches attached together in 3D, mimicking the sewing process of garments. The rest shape is planar, except for some seams where we may set a non-zero rest angle. The 2D patches are triangulated, and we also lay the yarn structure on the 2D patches. We do this everywhere on the cloth garment, also in areas that are simulated at triangle level, for yarn-level rendering purposes. At every seam, we add an additional rod aligned with the seam and we connect it to yarns that reach the seam. As a postprocess to simulation, we subdivide the triangle mesh on the triangle-only region using Loop subdivision, and we transform dummy yarns from their 2D layout to 3D. We generate splines from the resulting network of yarn crossings using

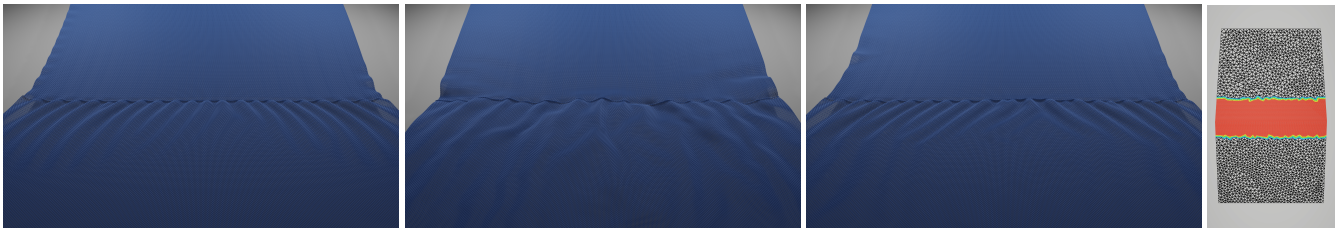

Figure 4: *Simulation of seam puckering, which occurs when two patches of different lengths are sewn together. From left to right: wrinkles of a full-yarn model, incorrect puckering with a triangle-only model, accurate result with our mixed simulation, and distribution of yarns and triangles in the scene.*

the procedural method of Cirio et al. [CLMMO14], we mesh tubes along the splines, and we render this geometry with path tracing.

In all our examples, the full-yarn region is predefined. We run first a triangle-only simulation to identify areas of high stress and/or high curvature, and label those as full yarn. The selection is manual, not automated. Then, we define the hybrid region as the triangles in the 1-ring around the full-yarn region. The blending field α is linearly interpolated between 1 and 0 on these triangles.

Our runtime simulation method is implemented partly on GPU and CPU. We carry out the following operations on the GPU: all force computations (including penalty-based contact forces), sparse matrix multiplications (including multiplication by the force Jacobian and the sparse portions of the preconditioner within MPCG), and multiplications by $\mathbf{B}^T$ and $\mathbf{B}$ in the application of the filter (3). We carry out the following operations on the CPU: collision detection (using spheres for self-collisions and signed distance fields for characters), and direct solves on small linear problems. As discussed in Section 3.2 and Section 5.2, the product with the filter and the preconditioner require, in total, the solution to three small linear problems on every iteration of MPCG. We fetch the right-hand-side from the GPU, solve the systems using precomputed Cholesky or LU factorizations as appropriate on the CPU, and copy the result to the GPU. We found that, thanks to the small number of triangle vertices on the hybrid region, these operations are not a bottleneck.

Due to the extremely dense resolution of yarn-level models, the force Jacobian may simply not fit in video memory. Specific data are reported in Table 1 and discussed in Section 6.3 below. To circumvent the memory limitation, our GPU implementation of the product with the yarn-level force Jacobian is matrix-free. On every MPCG iteration, we visit all yarn-level energy elements, and we evaluate the Jacobian matrix terms of their local stencils.

We have executed all our experiments on a AMD Ryzen 7 3700X 8-core 3.60 GHz PC with 32 GB of RAM and a Nvidia GeForce GTX 1080 Ti GPU with 11 GB of RAM.

6.2. Experiments

We have tested our simulation method on several experiments. We have picked scenarios where the tension of garments produces highly detailed wrinkling and plasticity effects, and hence the yarn-level model becomes imperative. However, such effects are typically localized, which motivates the combination of yarns and triangles for maximum efficiency.

Some of the experiments employ small patches of fabric to demonstrate effects that may appear also on full garments. Fig. 5 shows a typical example of a sleeve wrinkling as the elbow bends. Wrinkles and folds arise, crossing to and from the yarn and triangle regions in the model, and the transition is handled smoothly by our method. Fig. 4 shows the puckering effect when two patches of different length are sewn together. One of the patches is pre-stretched, and it compresses the other patch, producing fine wrinkles. We place yarns only at the seam, yet our method succeeds in capturing puckering accurately. Fig. 7 shows an example of plastic yield on a woven patch. Fine-scale plastic effects are supported naturally, at structural level, by the yarn-level model. The example demonstrates that the effect is captured smoothly even if yarns are placed only close to the high-tension region. Fig. 6 shows an example of plastic deformation due to internal friction at the yarn level. In this case, we use a very small patch; nevertheless, the effect is well represented with localized yarns.

Other experiments employ full garments to demonstrate effects that appear in fit and drape simulations. In these experiments, we initialize the garment by running a triangle-only simulation. Then, we place yarns at regions of interest, and we execute a simulation to relax the garment again. Fig. 1 shows a simulation of a cotton knit shirt of very high resolution (close to 4M yarn crossings), where

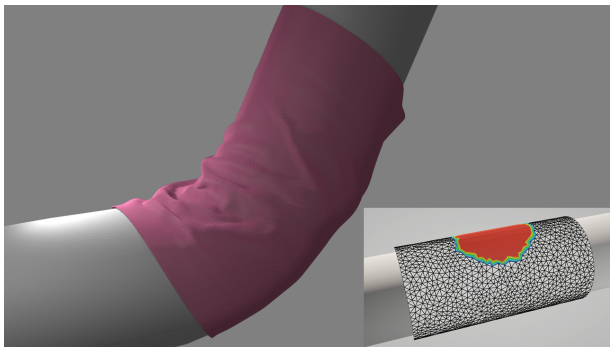

Figure 5: *A sleeve is simulated with a combination of yarns and triangles. As shown in the inset, we place full yarns at the elbow joint. As the elbow bends, wrinkles and folds appear around the joint. These wrinkles and folds extend across both the yarn and triangle regions, but the transition is handled smoothly by our method.*

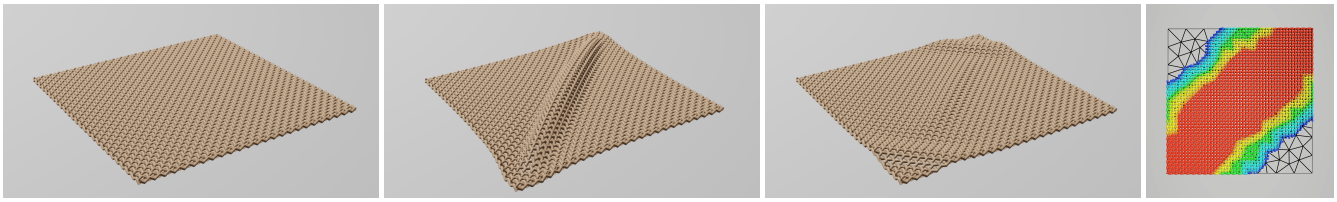

Figure 6: *Simulation of plastic deformation due to internal friction, resolved naturally at the yarn level. The patch is pulled from two corners and then released, leading to plastic deformations. The effect is represented well even using yarns on a very thin strip of fabric.*

the region of interest is placed at an armpit. The shirt is tight, under high stress, and produces fine wrinkles in the area. The region of interest does not cover all the surface where fine wrinkles appear, but the simulation is sufficient to communicate the wrinkling effect.

Fig. 9 shows a simulation of a linen woven dress, where the region of interest is placed on the chest. Again, our mixed simulation method succeeds at capturing fine local detail with a smooth transition between the yarn and triangle models. In this example, we have also evaluated the simulation result using triangle remeshing [NSO12], on ARCSim. In this particular case, ARCSim fails to produce the fine wrinkles of our enriched simulation. This limitation of triangle remeshing may be due to two reasons. First, the continuum elastic energy model may not reach the same accuracy as the yarn-based model. Second, note that remeshing in ARCSim is determined based on a heuristic sizing field, not a conservative error estimation, hence it cannot guarantee a fully accurate result. Note, however, that triangle remeshing is complementary to our approach. Triangle remeshing would probably allow even smaller yarn-based regions, thanks to the gradual transition from fine to coarse triangles, and hence a lower computational cost. However, by relying only on remeshing we would miss the extra mechanical accuracy and yarn-level plasticity effects of our method.

Our examples focus on static cloth drape and fit. Beyond animation, these static drape simulations are of utter importance to applications such as fashion. Moreover, the simulations are not static even if the desired final result is static. The fabric is initialized under high tension, as the yarns are laid on top of a triangle-based initialization, and this tension induces notable local dynamics. Nevertheless, our method works well for animated scenes. As a validation, we show the small example in Fig. 8. In dynamic scenes, however, the locations of wrinkles and detailed deformations are more difficult to predict, and yarns should be introduced dynamically.

6.3. Performance Comparisons

In Table 1, we report the problem size, memory footprint, and computational cost for the examples shown in the paper. In most cases, we compare data of full simulations and our mixed simulations. On the full garments, we achieve speed-ups of 25x (on the dress) and 5.6x (on the shirt). It is also important to pay attention to the memory footprint. With a full representation, the memory footprint of the full-matrix implementation is often excessive for full garments, and we must resort to a less efficient matrix-free implementation. With the mixed representation, on the other hand, a full-matrix implementation would be possible, which might introduce additional performance gain.

All simulations were executed with adaptive time stepping, varying depending on the complexity of the contact configuration, the stress of the fabric, and hence on the number of iterations of the Newton solve. We used a reference time step of 0.1 ms for the full-garment simulations, and 1 ms for the smaller examples.

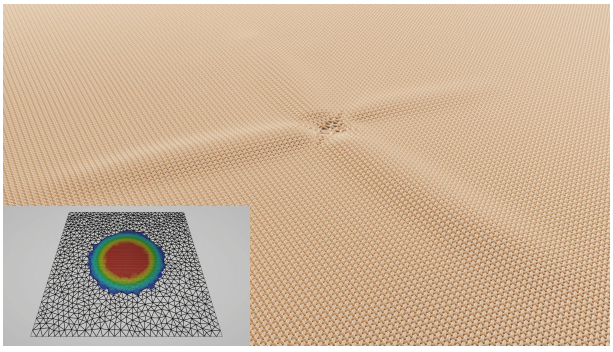

Figure 7: *We pull a woven patch from adjacent yarns in opposite directions, and produce yarn-level plastic yield. The effect is captured smoothly with yarns placed only close to the high tension region.*

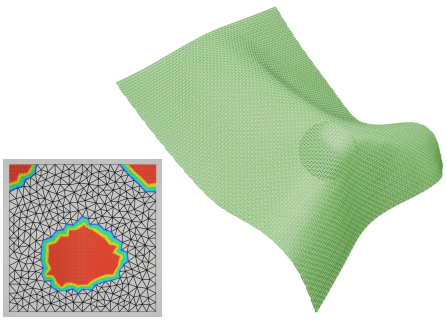

Figure 8: *Example showing dynamic interactions of a small patch of cloth (5 cm × 5 cm) with a ball. We place yarns on the attachment and collision regions, and the simulation handles smoothly the transition to and from yarns and triangles.*

Example	Full yarn nodes	Hybrid yarn nodes	Triangle-only vertices	Hybrid vertices	Memory (MB) (matrix-free)	Memory (MB) (with matrix)	Time (secs) per step
Shirt (mixed) (Fig. 1)	88 388	62 261	2 066	66	371	772	6.54
Shirt (full) (Fig. 1)	3 757 255	-	-	-	3 419	13 500	36.5
Dress (mixed) (Fig. 9)	83 382	31 460	6 067	124	373	679	2.88
Dress (full) (Fig. 9)	2 662 944	-	-	-	2 997	10 096	72.2
Elbow (Fig. 5)	14 437	4 002	2 352	115	221	270	4.65
Puckering (mixed) (Fig. 4)	57 808	12 814	1 781	231	293	481	0.41
Puckering (full) (Fig. 4)	245 156	-	-	-	457	1 110	1.72
Plastic yield (mixed) (Fig. 7)	6 124	18 675	1 031	220	278	476	0.80
Plastic yield (full) (Fig. 7)	44 622	-	-	-	446	564	2.63

Table 1: *Statistics for the experiments shown in the paper. We report the problem size, memory footprint, and computational cost. The reference time step was* 0.1 *ms for the full garments, and* 1 *ms for the smaller examples. As a reference, we also report the estimated memory footprint with a matrix-based implementation of the force Jacobian.*

7. Limitations and Future Work

In this work, we have presented a simulation method that combines yarn and triangle models on the same cloth object. We obtain high detail at regions of interest, with a smooth transition to a coarser but more efficient representation on the overall object. The solution is possible thanks to a novel enrichment approach that tackles kinematics, dynamics, and solver efficiency.

Our work demonstrates the potential of mixing yarns and triangles, but it also suffers from limitations, and several further improvements are possible. At present, our model supports static but not dynamic adaptivity of the representation. Dynamic adaptivity can be approached in two ways. One is to enrich the triangle representation with yarns on the fly. The other one is to adopt adaptive remeshing [NSO12] on the triangle region. In our method, we leverage precomputation to use efficient solvers in the filter and the preconditioner. Frequent and large updates to the discretization of the hybrid region could hamper this precomputation; therefore, the design of a solution for adaptivity is not free of challenges.

Since yarn and triangle models are mixed, a smooth transition also depends on the similarity of their energy models. We estimated parameters for the triangle model to match energies of the yarn model under uniform stretch and bending. However, the behavior of the two models is not identical. For knits, in particular, our choice of triangle-level energy cannot capture the anisotropy of the fabrics. More complex energy models, based on strain-dependent interpolation [WOR11, MBT*12], might improve the similarity.

As mentioned in Section 3, our enriched representation is limited to the elastic deformation of the fabric; it does not support plastic deformation. Plasticity is possible and demonstrated in our examples, but it is handled only in full-yarn regions. It might be possible to include a triangle-level plasticity model, and design an enrichment approach for the Eulerian coordinates of the yarn model.

To conclude, our mixed representation succeeds to balance well the number of degrees of freedom of the overall simulation, and hence its impact on the simulation cost, by placing yarns only at regions of interest. However, enriching a triangle-based simulation with yarns also has a negative effect on the time step, due to the use of smaller elements. One approach to alleviate this problem might be the use of a multi-step numerical integration scheme.

Acknowledgments. Alberto Martín, Rosa Sánchez, Héctor Barreiro, Igor Santesteban and Javier Tapia for help with renders and videos; KDAB and Stack Overflow for coding tips. This work was funded in part by the ERC (ERC-2017-CoG-772738 TouchDesign) and the Spanish Ministry of Science (RTI2018-098694-B-I00 VizLearning, PTQ-17-09154, PTQ-17-09156, PTQ2018-010308).

References

[AB03] ASCHER U. M., BOXERMAN E.: On the modified conjugate gradient method in cloth simulation. *Vis. Comput. 19*, 7-8 (2003). 3, 5

[Box03] BOXERMAN E.: *Speeding Up Cloth Simulation*. Master's thesis, The University of British Columbia, Canada, 2003. 6

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (1998), SIGGRAPH '98, ACM, p. 43–54. 3, 5

[CLMMO14] CIRIO G., LOPEZ-MORENO J., MIRAUT D., OTADUY M. A.: Yarn-level simulation of woven cloth. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 207:1–207:11. 1, 2, 3, 5, 7

[CLMO17] CIRIO G., LOPEZ-MORENO J., OTADUY M. A.: Yarn-level cloth simulation with sliding persistent contacts. *IEEE Transactions on Visualization and Computer Graphics 23*, 2 (2017), 1152–1162. 2, 4

[FLLP13] FAN Y., LITVEN J., LEVIN D. I. W., PAI D. K.: Eulerian-on-lagrangian simulation. *ACM Trans. Graph. 32*, 3 (2013). 2, 3

[GHDS03] GRINSPUN E., HIRANI A. N., DESBRUN M., SCHRÖDER P.: Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), SCA '03, Eurographics Association, p. 62–67. 4

[GKS02] GRINSPUN E., KRYSL P., SCHRÖDER P.: CHARMS: a simple framework for adaptive simulation. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), SIGGRAPH '02, ACM, p. 281–290. 2

[GPS01] GOLDSTEIN H., POOLE C. P., SAFKO J. L.: *Classical Mechanics (3rd Edition)*, 3 ed. Addison-Wesley, June 2001. 4

[GVL96] GOLUB G. H., VAN LOAN C. F.: *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, 1996. 6

[HZ13] HARMON D., ZORIN D.: Subspace integration with local deformations. *ACM Trans. Graph. 32*, 4 (2013), 107:1–107:10. 2

Figure 9: *Simulation of fit and drape of a linen dress. We compare our mixed model to triangle remeshing using ARCSim [NSO12]. In the top row, we show the placement of yarns in the chest area with our model (left), a full view of our simulation result (center), and the result of ARCSim (right). In the bottom row, we show a closeup of the chest with our coarse triangles only (left), fine wrinkles produced by high tension with our mixed model (center), and the result with triangle remeshing (right).*

[JGT17] JIANG C., GAST T., TERAN J.: Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM TOG 36*, 4 (2017). 2

[KJM08] KALDOR J. M., JAMES D. L., MARSCHNER S.: Simulating knitted cloth at the yarn level. *ACM Trans. Graph. 27*, 3 (2008), 65:1–65:9. 1, 2

[KJM10] KALDOR J. M., JAMES D. L., MARSCHNER S.: Efficient yarn-based cloth with adaptive contact linearization. *ACM Transactions on Graphics 29*, 4 (July 2010), 105:1–105:10. 2

[LWS*18] LEAF J., WU R., SCHWEICKART E., JAMES D. L., MARSCHNER S.: Interactive design of periodic yarn-level cloth patterns. *ACM Trans. Graph. 37*, 6 (2018), 202:1–202:15. 2

[MBT*12] MIGUEL E., BRADLEY D., THOMASZEWSKI B., BICKEL B., MATUSIK W., OTADUY M. A., MARSCHNER S.: Data-driven estimation of cloth simulation models. *CGF 31*, 2 (2012). 1, 9

[MGL*15] MALGAT R., GILLES B., LEVIN D. I. W., NESME M., FAURE F.: Multifarious hierarchies of mechanical models for artist assigned levels-of-detail. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (New York, NY, USA, 2015), ACM, pp. 27–36. 2, 3

[MTB*13] MIGUEL E., TAMSTORF R., BRADLEY D., SCHVARTZMAN S. C., THOMASZEWSKI B., BICKEL B., MATUSIK W., MARSCHNER S., OTADUY M. A.: Modeling and estimation of internal friction in cloth. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 212:1–212:10. 1

[MWN*17] MANTEAUX P.-L., WOJTAN C., NARAIN R., REDON S., FAURE F., CANI M.-P.: Adaptive physically based models in computer graphics. *Comput. Graph. Forum 36*, 6 (2017), 312–337. 2

[NPO13] NARAIN R., PFAFF T., O'BRIEN J. F.: Folding and crumpling adaptive sheets. *ACM Trans. Graph. 32*, 4 (2013), 51:1–51:8. 2

[NSO12] NARAIN R., SAMII A., O'BRIEN J. F.: Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph. 31*, 6 (2012), 152:1–152:10. 1, 2, 8, 9, 10

[NWYM19] NARAYANAN V., WU K., YUKSEL C., MCCANN J.: Visual knitting machine programming. *ACM Trans. Graph. 38*, 4 (2019). 2

[Ogd97] OGDEN R. W.: *Non-linear elastic deformations*. Dover Publications, New York, NY, 1997. 4

[PNdJO14] PFAFF T., NARAIN R., DE JOYA J. M., O'BRIEN J. F.: Adaptive tearing and cracking of thin sheets. *ACM Trans. Graph. 33*, 4 (2014), 110:1–110:9. 2

[TJM15] TAMSTORF R., JONES T., MCCORMICK S. F.: Smoothed aggregation multigrid for cloth simulation. *ACM Trans. Graph. 34*, 6 (2015), 245:1–245:13. 2

[TTN*13] TANG M., TONG R., NARAIN R., MENG C., MANOCHA D.: A gpu-based streaming algorithm for high-resolution cloth simulation. *Comput. Graph. Forum 32* (2013), 21–30. 2

[WOR11] WANG H., O'BRIEN J. F., RAMAMOORTHI R.: Data-driven elastic models for cloth: modeling and measurement. *ACM Transactions on Graphics (SIGGRAPH 2011) 30*, 4 (Aug. 2011), 71:1–71:12. 1, 9

[WSY19] WU K., SWAN H., YUKSEL C.: Knittable stitch meshes. *ACM Trans. Graph. 38*, 1 (2019), 10:1–10:13. 2

[YKJM12] YUKSEL C., KALDOR J. M., JAMES D. L., MARSCHNER S.: Stitch meshes for modeling knitted clothing with yarn-level detail. *ACM Trans. Graph. 31*, 4 (2012), 37:1–37:12. 2