Unilaterally Incompressible Skinning

Mickaël Ly, Florence Bertails-Descoubes, Damien Rohmer

▶ To cite this version:

Mickaël Ly, Florence Bertails-Descoubes, Damien Rohmer. Unilaterally Incompressible Skinning. [Research Report] Inria - Research Centre Grenoble – Rhône-Alpes. 2017. hal-01512412v1

HAL Id: hal-01512412
https://inria.hal.science/hal-01512412v1

Submitted on 22 Apr 2017 (v1), last revised 2 May 2017 (v2)
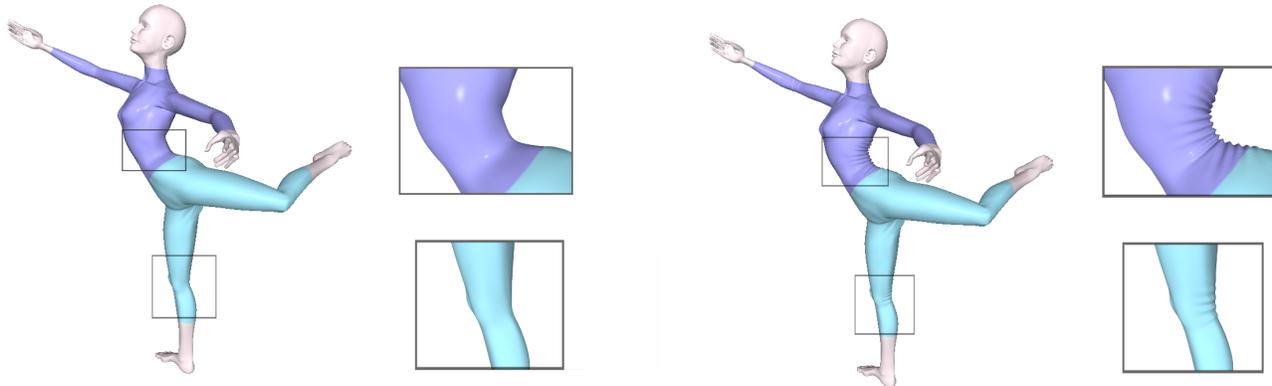
Unilaterally Incompressible Skinning

Mickaël Ly   Florence Bertails-Descoubes   Damien Rohmer

INRIA and Laboratoire Jean Kuntzmann (Grenoble University, CNRS), France

Figure 1: *Compared to traditional skinning (left), our new skinning algorithm (right) strictly prevents the cloth surface from compressing beyond its rest length, thus spontaneously yielding folding patterns when the dancer performs an arabesque. On this example and with the same level of subdivision for the mesh, performance are only slowed down by a factor 2 compared to classical skinning.*

Abstract

Skinning was initially devised for computing the skin of a character deformed through a skeleton; but it is now also commonly used for deforming tight garments at a very cheap cost. However, unlike skin which may easily compress and stretch, tight cloth strongly resists compression: inside bending regions such as the interior of an elbow, cloth does not shrink but instead buckles, causing interesting folds and wrinkles which are completely missed by skinning methods. Our goal is to extend traditional skinning in order to capture such folding patterns automatically, without sacrificing efficiency. The key of our model is to replace the usual skinning formula – derived from, e.g., Linear Blend Skinning or Dual Quaternions – with a complementarity constraint, making an automatic switch between, on the one hand, classical skinning in zones prone to stretching, and on the other hand, a quasi-isometric scheme in zones prone to compression. Moreover, our method provides some useful handles to the user for directing the type of folds created, such as the fold density or the overall shape of a given fold. Our results show that our method can generate similar complexity of folds compared to full cloth simulation, while retaining interactivity of skinning approaches and offering intuitive user control.

Keywords: skinning, cloth animation, buckling, unilateral constraint, complementarity

1 Introduction

From princesses dressed in a bustier dress with long gloves to superheroes wearing their trademark costume, feature films are populated with characters wearing skin-tight garments. For the sake of simplicity and reduced computational cost, such garments are, in general, not simulated physically. Instead, their deformation is usually generated geometrically from that of the character skeleton, following a so-called *skinning* technique. However, while skinning captures elongation well, it fails to mimic the actual behavior of cloth in compression zones. As a consequence, when Queen Elsa from *Frozen* gives a bow, her bustier seems to be made of rubber instead of real fabrics, as no fold appears where expected to be. Such a visual disturbance, common in feature movies and games, has motivated us to propose an improved version of skinning specifically dedicated to skin-tight cloth.

Our goal is to include an assymetric behavior for skinning, in order to account for the fact that tight cloth may easily dilate, but strongly resists compression and instead buckles. We materialize this assymetric behavior thanks to an unilaterally incompressible constraint, formulated through nonlinear complementarity. Such a model automatically activates and adjusts positive offset variables along the cloth mesh so as to allow the cloth to detach from the usual skinning surface in zones where compression would otherwise occur. The density of folds and their general shape can be specified by the user, giving her the freedom to design the overall look of the folds without having to manipulate indirect material parameters, nor having to care about isometry constraints.

Our new skinning formula only adds a few scalar variables per articulation compared to the standard formulae (derived from, e.g., linear blend skinning or dual quaternions). Those new offset variables are accurately solved for using just one iteration of a damped quasi-Newton method. Compared to previous approaches, our model prevents cloth compression in an implicit and quasi-exact way, spontaneously creating folds where needed, and whose general pattern remains easily controllable by a user.

Overall, our method greatly enhances the realism of deformed tight garments, without sacrificing computational efficiency of traditional skinning techniques.

2 Related Work

Prior art relevant to our approach includes methods dedicated to skinning, cloth modeling and cloth simulation, as well as correction

methods adding some geometrical details to garments as a post-process.

Skinning The introduction of skeleton-driven skin deformation (so-called *skinning*), is generally attributed to Magnenat-Thalmann *et al.* [1988]. It consists in deforming a mesh around an articulated rigid skeleton through weighted combinations of rigid transformations. Due to its simplicity of formulation and its cheap evaluation cost (at least in its most trivial form), skinning has been extensively used in production for deforming the flesh of animated characters. The simplest technique, called linear blend skinning (LBS), suffers from well-known surface shrinking artifacts (such as the *collapsing elbow* effect), which has motivated subsequent improvements such as new skinning formulations [Kavan and Zara 2005; Kavan et al. 2008; Jacobson and Sorkine 2011] and post-correction schemes [Vaillant et al. 2013; Zhu et al. 2011; Kry et al. 2002; Weber et al. 2007; Wang et al. 2007].

It is noteworthy that the above skinning literature has been constantly striving to improve the deformed shape of a homogeneous layer around bones, made of human flesh and skin – which can, indeed, reasonably be considered as compressible at a macroscopic scale. However, these approaches are not seeking to synthesize wrinkles that should appear in the case of some non-compressible layer on top of skin, such as *cloth* (see Figure 2a). As a consequence, tight garments modeled using mere skinning – which is of common practice in production – dramatically lack realism as they rather seem to be made of rubber instead of actual fabrics (see Figure 2b).
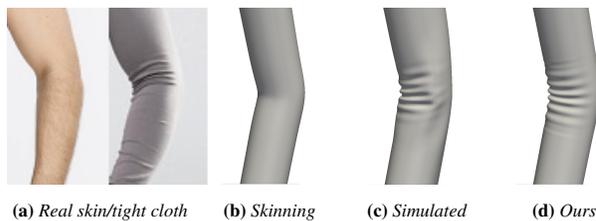
Cloth physical simulation Cloth physics-based simulation has emerged in the 90's, boosted by the seminal work of Baraff and Witkin [1998] who showed that full garments could be animated in reasonable timings thanks to first-order implicit integration. Since then the field has successfully expanded with more and more sophisticated methods for treating bending [Bridson et al. 2003; Grinspun et al. 2003], stretch limiting [Goldenthal et al. 2007; English and Bridson 2008; Thomaszewski et al. 2009] and frictional contact [Bridson et al. 2002; Otaduy et al. 2009], or for incorporating real material properties in the model [Wang et al. 2011].

However, all above methods were designed to animate loose garments, whose apperance is many dictated by dynamical events such as free-flight motion, impacts, or stick-slip phenomena due to frictional contact. Moreover, capturing wrinkle patterns both accurately and efficiently remains a major issue: as detailed below, the common solution to accelate the framerate is rather to use simulation at coarse resolution, then to enrich geometry with procedurally computed folds.

Still, one legitimate question is: since simulation approaches are able to deal with very complex phenomena such as knot tying or multiple layered cloth at high speeds, why would they be unable to handle skin-tight garments, which (at first sight) seem to be simpler scenarios to deal with? The message we want to convey in this paper is that skin-tight garments are a very specific and challenging type of scenario, for which physics-based simulator are particularly bad-suited for dealing with. They are notably characterized by an assymetric stretching behavior, by the major influence of static friction, and by the predominance of buckling due to bending motion (see Section 3); all of this giving rise to very peculiar fold patterns difficult to reproduce with current simulators. Our own simulation experiments (reported in Section 6) show that a deformation scenario involving skin-tight cloth is not only complex to set up, but also very costly to compute. Moreover, the resulting folding pattern turns out to be hardly controllable.

In contrast, our method, which is purely geometric, avoids the unnecessarily computational burden due to full cloth simulation.

Cloth geometrical deformation Closer to our method, wrinkles may also be synthesized using geometrical deformation, leading to faster and easier control of the resulting wrinkles appearance. On one side, wrinkle can be fully automatically synthesized. For instance, data-driven approaches apply deformations to model wrinkles with respect to some training examples parameterized using skeleton position [Wang et al. 2010; Kim et al. 2013; Xu et al. 2014], subspace reduction [Kavan et al. 2011; Guan et al. 2012; Hahn et al. 2014], or edge compression [Zurdo et al. 2013]. Such approaches can handle large variety of wrinkles appearance at interactive frame rate, but require an existing data base of finely designed wrinkles for each character cloth. Subdivision applied as a post-process on top of low resolution meshes was proposed by Muller and Chentanez [2010] and allows approximate length preservation for general meshes. Purely procedural approach based on expected wrinkle pattern were proposed by Decaudin *et al.* [2006], but the method is limited to cylindrical shape and do not handle tight cloth wrinkles. Remillard et al. [2013] proposed a method mixing physical simulation and a-priori knowledge of wrinkle wavelength, but suffer from the computational cost of the simulation. On the other side, fully manual placement of individual wrinkle on character animation has been proposed [Cutler et al. 2005], although such approach may be time consuming to set up and do not guarantee that wrinkles are placed accurately with respect to the deformation. Semi-automatic approach enable to automatically compute the wrinkle position and magnitude while letting some user control on the wrinkle shape appearance, such as their wavelength. Based on the measurement of the length compression with respect to a rest pose, wrinkles locations were computed and their appearances were synthesized using texture pattern [Hadap et al. 1999], or geometrical deformation [Larboulette and Cani 2004]. These methods were further extended for cloth [Rohmer et al. 2010; Gillette et al. 2015] and skin wrinkles [Turchet et al. 2015] in order to ensure an exact length preservation for each individual synthesized wrinkle. Still, these approaches do not guarantee that the length is preserved across several wrinkles as we propose in this work. Furthermore, the latter approaches are restricted to semi-circular profiles, while our method can handle more general shape, as given by third order polynomials in our examples, possibly fitting to artistic control.

.

(a) *Real skin/tight cloth* **(b)** *Skinning* **(c)** *Simulated* **(d)** *Ours*

Figure 2: *Comparing the result of standard skinning (b), simulation (c), and our interactive approach (d), against a reference picture showing real cloth wrinkle patterns on a bending arm (a).*

3 Unilaterally incompressible skinning

This paper focuses on the realistic yet efficient and controllable deformation of *skin-tight garments*.

3.1 Skin-tight garments

Skin-tight garments, common in real and virtual worlds under the form of, e.g., bustiers, leotards, sportswear tee-shirts and pants, skinny jeans, leggings, or socks, are generally made of spandex, a synthetic fiber featuring an exceptional elasticity. Unlike other

types of garments which may bend easily but are far less stretchable, skin-tight garments have the particularity to be *stretched* when put on a human body.

One first observation is that a worn skin-tight garment features a peculiar *assymetric* behavior when the body deforms. Think for instance of a dancer wearing a lycra ballet leotard: when the back arches, the fabric sticks to the skin at the belly and the chest – where the body stretches – but detaches from the skin and forms folds at the back – where the body bends. Mechanically, the fabric can always stretch elastically when the body stretches, whereas it cannot compress beyond a certain threshold (corresponding to its rest configuration). In practice, this threshold is commonly reached very soon after the body has started to bend.

A second observation is that wrinkle patterns emerging in skin-tight garments are fairly simple and *predictible*, compared to those appearing in loose garments, or simply in non pre-stretched garments. As depicted in Figure 2a, folds all follow parallel lines perpendicular to the bending direction. As a result, ringing patterns emerge around bending articulations such as the back of the knee or the interior of the elbow.

Finally, as illustrated in the accompanying video, we have observed that torsion does not yield folds in a pre-stretched garment: bending is the only source of wrinkling. All these observations encouraged us to think that a realistic yet interactive model could be designed for capturing the deformations of skin-tight garments, through a simple and cheap modification of standard skinning.

3.2 Model overview

To model the assymetric deformation behavior typical of skin-tight cloth, we propose *unilaterally incompressible skinning*, a double-layered version of traditional skinning where the top (cloth) surface is authorized to detach from the underlying (skin) surface while being prevented from compressing. This dual mechanism, combined with a parametric elastic model, spontaneously creates the emergence of folds in compression regions, while retaining the desirable behavior of skinning in stretching regions.

Complementarity for preventing compression The term *unilateral incompressibility* was first coined by Narain et al. [2009] in the context of crowd simulation, and then reused by several authors for modeling the dynamics of granular materials [Narain et al. 2010; Daviet and Bertails-Descoubes 2016]. It expresses, through a complementarity constraint, the fact that the material that is considered may dilate infinitely, but should not compress beyond a certain threshold. In our case, a complementarity constraint on $\mathbb{R}$, $0 \le x \perp y \ge 0$, which represents the simultaneous non-negativeness and orthogonality of the two variables x and y, is also a good way to model our assymetric compression constraint.

Let S^0 be the rest pose mesh, S' the skinned mesh obtained via any skinning method from S^0, and S'' the (unknown) mesh resulting from our modified skinning algorithm. Formally, one would like to have

$$0 \le X \perp \tau \ge 0,$$

where τ measures the stretching of the surface S'' w.r.t. the rest surface S^0, and X is an offset variable serving to "activate" taking off from the skinned surface. The constraint means, on the one hand, that if the surface stretches ($\tau > 0$), we have necessarily $X = 0$ and we keep standard skinning; on the other hand, if the surface wants to compress, it is prevented to do so and kept strictly isometric ($\tau = 0$) by modifying skinning thanks to a positive offset X.

In the following, we build a precise 1D model for the offset X, accounting for the bending elasticity of the surface, and we explain how S'' and τ can be efficiently computed from X. We also show that an extremely fast solver can be leveraged to solve our complementarity constraint with maximum precision. For a full surface, we note that it is sufficient to guarantee exact non-compression on *folding lines*, whereas the transverse direction is reasonably prevented from compression through a decaying extrusion of the fold.

Contributions More specifically, our contributions include:

- An exact and fast non-compressible skinning algorithm in 1D, relying upon a new complementarity formulation which is solved extremely efficiently with maximum precision. Combined with a user-defined wrinkle density and general fold shape, our method creates desirable quasi-isometric folds where needed (section 4).

- The extension of our 1D scheme to an arbitrary 2D mesh through a number of automatic processes, including adaptive subdivision, detection of *folding lines*, and generation of local parameterizations around folding lines (section 5).

- A careful evaluation of our method w.r.t. classical skinning and physics-based cloth simulation, in terms of realism, computational efficiency, and user controllability (section 6).

Disclaimer Note that our method is not meant to handle loose garments, nor to fix common artifacts of skinning. Its very purpose is to provide the user with a convenient tool for manipulating skin-tight cloth interactively, in the same spirit skinning was designed to deform character skin in an efficient, robust, and easy-to-control way.

4 Unilaterally incompressible skinning in 1D

We first design a quasi-isometric non-compression algorithm for 1D meshes. Our method will be extended to 2D surfaces in Section 5.

4.1 Naïve model without bending elasticity

In this first simplistic model, we assume that each vertex of the mesh is allowed to move on a half-line perpendicular to the skinned surface S', directed by its outward normal. For now, no bending elasticity constraint is considered between the new vertices of the mesh S'': vertices are related to each other only through the unilaterally incompressible constraint, which here boils down to an isometric constraint in compression zones.

Let $\mathbf{p}_i^0$ be the n vertices of the rest mesh S^0, and $\mathbf{p}_i'$ the vertices of the surface S' obtained from $\mathbf{p}_i$ by (any) standard skinning. We construct new vertices $\mathbf{p}_i''$ of our non-compressed surface S'' as

$$\mathbf{p}_i''(X_i) = \mathbf{p}_i' + X_i \mathbf{v}_i, \tag{1}$$

where $\mathbf{v}_i$ is the outward normal to the skinned surface S' at vertex $\mathbf{p}_i'$. Moreover, denoting by ℓ_i^0 the local rest length of the mesh, we can compute the elongation of the mesh with this simple formula,

$$\tau_i(\mathbf{X}) = \|\mathbf{p}_{i+1}''(X_{i+1}) - \mathbf{p}_i''(X_i)\|_2 - \underbrace{\|\mathbf{p}_{i+1} - \mathbf{p}_i^0\|_2}_{\ell_i^0},$$

where $\boldsymbol{\tau} = \{\tau_i\}$ is a $\mathbb{R}^n \to \mathbb{R}^n$ function which turns out to be non-linear w.r.t. our vector of offset variables $\mathbf{X} = \{X_i\} \in \mathbb{R}^n$. In the remainder of the paper, the $\boldsymbol{\tau}$ function will be referred to as our *stretching function*.

Nonlinear Complementarity Problem   Our full problem reads

$$0 \leq \mathbf{X} \perp \boldsymbol{\tau}(\mathbf{X}) \geq 0, \tag{2}$$

which boils down to a Nonlinear Complementarity Problem (NCP) (which has to be understood componentwise). While many solvers have been designed for solving Linear Complementarity Problems (LCPs), the literature on NCPs is much less abundant. Yet, one common approach to solve a (scalar) complementarity problem of the form $0 \leq a \perp b \geq 0$ is to use an equivalent functional formulation of the form $\Phi(a,b) = 0$, where Φ is a *merit* function which has the particularity to be nonsmooth [Fischer and Jiang 2000]. One may then solve the above root-finding problem by minimizing $\|\Phi(a,b))\|$ thanks to a nonsmooth optimization method. One popular merit function Φ is the so-called *Fischer-Burmeister* function, which reads

$$\Phi_{\text{FB}} : \quad \begin{array}{ccc} (\mathbb{R}_+)^2 & \to & \mathbb{R} \\ (a,b) & \to & \sqrt{a^2+b^2} - a - b. \end{array}$$

One may easily check that $0 \leq a \perp b \geq 0 \iff \Phi_{\text{FB}}(a,b) = 0$. Unlike other merit functions based on projections, the Fischer-Burmeister function benefits from better regularity properties, which makes it better-suited for algorithmic.

Our complementarity problem (2) is thus equivalent to cancelling the error function ξ defined as

$$\xi : \quad \begin{array}{ccc} (\mathbb{R}_+)^n & \to & \mathbb{R} \\ \mathbf{X} & \to & \|\boldsymbol{\Phi}_{\text{FB}}(\mathbf{X})\|_2, \end{array}$$

where $\boldsymbol{\Phi}_{\text{FB}}$ is the multidimensional function

$$\boldsymbol{\Phi}_{\text{FB}} : \quad \begin{array}{ccc} (\mathbb{R}_+)^n & \to & \mathbb{R}^n \\ \mathbf{X} & \to & \left[\Phi_{\text{FB}}(X_i, \tau_i(\mathbf{X}))\right]_{i \in [\![1,n]\!]}. \end{array}$$
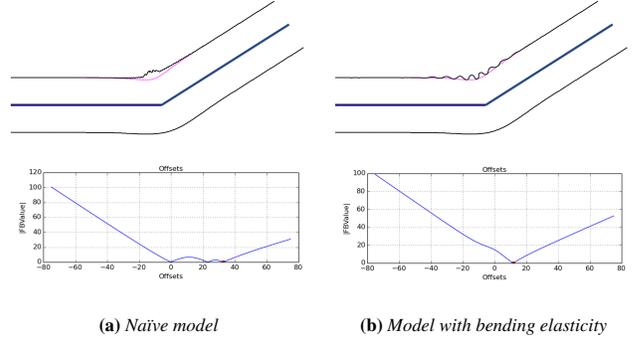
When experimenting with this model, we have in practice attempted to solve (2) by minimizing ξ using a generalized BFGS method where gradients are replaced with subgradients at nonsmooth points. Note that as the Fischer-Burmeister function has only one single nonsmooth point which coïncides with its minimum, we chose a zero subgradient at this location. Regular gradients can be evaluated at any other point, since the function is differentiable everywhere else. This method however led to a number of issues which are reported below.

Model drawbacks   This first per-vertex non-compressive model actually suffers from two major limitations.

On the one hand, some serious *solvability* problems are tight to this formulation. This is not really surprising, since the question of existence and uniqueness of a solution to our problem (2) is directly related to the strict monotonicity of the function $\boldsymbol{\tau}(\mathbf{X})$ [Ferris and Kanzow 1998], which is *not* garanteed in our case. Actually, the reader can even convince herself/himself through a simple example with a flat basis that this problem does not necessarily admit a solution. As a result, our minimization algorithm only tries to minimize the error function ξ, but often gets a nonzero residue. A first consequence is that exact isometry is not reached in such cases. Another, perhaps even more serious consequence, is that many different approximate solutions may exist, since ξ appears to have several local minimas (see Figure 3a). Without warmstarting, the method thus yields some popping artefacts. With warmstarting, our algorithm follows a continuous set of approximate solutions, but many different paths could be similarly followed, meaning that starting our algorithm at different poses would lead to different results.

Second, the *model* itself is clearly insufficient to create realistic folds in compression zones. The fundamental concept of *bending*

elasticity, which penalizes certain types of folds (especially sharp folds), is lacking to our formulation. In the following we propose to add a notion of bending elasticity to our model. This will not only improve the shape of folds we obtain, but also improve drastically the well-posedness of our problem, while providing some useful control handles to the user.

(a) *Naïve model*   (b) *Model with bending elasticity*

Figure 3: *Results without (a) and with (b) bending elasticity (the skinning surface S' is depicted in pink). In the naïve case (a), several approximate solutions (local minima) may be obtained, each of them leading to inconsistent fold patterns. In the case (b), the model offers a unique and consistent solution. Plots are displayed for one central peak/fold of the wrinkling patterns.*

4.2   Incorporating bending elasticity

Instead of dealing with individual vertices that may buckle independently of any bending energy, our idea is to reason on a *fold* basis, that is a consistent set of neighboring vertices that detach together from the skinned surface.

Fold geometry   The shape of folds emerging in compressed elastic sheets has been accurately described by physicists in [Audoly and Pomeau 2010, Section 8.5.3]. In this reference, an analytic dimensionless expression for the general shape of an isometric fold is given, which depends only on the compression rate applied onto the sheet. However, this formula is only valid for a sheet compressed over a flat support, and does not generalize to folds caused by an underlying bending motion. To the best of our knowledge, such bending folds have not yet been characterized geometrically.

Per-fold unilateral non-compression   Our idea is thus to start from a predefined fold geometry on a flat support – either physics-inspired or freely designed by the user – and, during animation, to adjust its *amplitude* so that unilateral non-compression is exactly guaranteed on the region of vertices covered by the fold. Globally, this means that we aim at computing, at each step of the animation, the amplitude of N (disjoint) folds in order to prevent compression. Compared to our previous problem, we actually keep our complementarity formulation (2), except that it is reduced to dimension N and relates a fold amplitude to a fold compression function: for $k \in [\![1,N]\!]$, each X_k now represents the amplitude of the kth fold, and each τ_k computes the difference between the total length of the new (unknown) fold and its total rest length L_k^0.
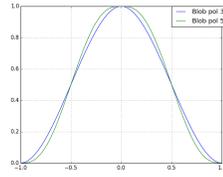
Before going deeper into equations, let us mention some important differences with our former per-vertex model:

- A positive amplitude X_k of the kth fold causes the detachment of *all* fold vertices (and not just one vertex), with an amplitude that is modulated by the general shape of the fold. Some example of fold shapes are given in the following;

- In compression regions, our former isometric constraint is replaced with a *fold length preservation* constraint. We thus relax the local isometric constraint in favor of a more global constraint, which actually makes our problem much better posed mathematically, as shown later on. Some local post-correction of isometry is also proposed in the following;

- Folds are *independent*, i.e., each fold is responsible for guaranteeing the complementarity constraint on the region it covers, independently from other folds. This is in contrast with the former model where all vertices were related to each other through a global stretching function. Now, our new stretching function τ_k is *local* to the kth fold (i.e., it only depends on X_k), as it evaluates difference in length only based on the vertices belonging to the corresponding fold region.

These three major differences do not only increase the realism of emerging folding patterns compared to our first model, but also greatly improve the well-posedness and ease of resolution of our problem. Later in this section, we explain how an extremely efficient solver can be designed for computing the (unique) solution of our full problem at maximum accuracy.

Predefined fold profile Our new non-compressive algorithm takes as input a general look of folds. This input takes the form of a 1D blob function of a scalar variable x. The user is free to design any function she desires, provided it is nonnegative and has compact support. Sharp, or in contrast flattened shapes, can thus freely be used. Concave shape is not mandatory; though, for the sake of simplicity, we consider this particular case in the following, with $x = 0$ coïnciding with the argmax of the fold. In practice, most results of this paper were obtained by using a polynomial blob of third order and radius α,

$$B_{3,\alpha}(x) \quad \begin{array}{ccc} [0,\alpha] & \rightarrow & [0,1] \\ x & \rightarrow & 1 + \frac{-3}{\alpha^2}x^2 + \frac{2}{\alpha^3}x^3, \end{array}$$

which turned out to yield visually pleasing folds.

Note that this blob function is smooth and has compact support, since we have

$$\begin{array}{llll} B_{3,\alpha}(0) & = & 1 & B_{3,\alpha}(\alpha) & = & 0 \\ B'_{3,\alpha}(0) & = & 0 & B'_{3,\alpha}(\alpha) & = & 0. \end{array}$$

Higher-order polynomial (for instance 5) would create a flattened shape (see insert, green curve). Of course, the fold profile does not need to be generated mathematically, it could well be designed by a 2D drawing (satisfying the contraint to be a positive function with compact support).

Modified skinning Let C be the function mapping a vertex index i to its blob index $C(i)$, corresponding to the index of the vertex that is allowed to move with maximum amplitude $X_{C(i)}$ (for symmetric blobs, $\mathbf{p}_{C(i)}$ corresponds to the center vertex of the blob). Note that if all blobs have disjoint interiors and only coincide through their extreme vertices, then $\{C(i), i \in [\![1,n]\!]\}$ forms a partition of $[\![1,N]\!]$. Our previous modified skinning formula (1) becomes

$$\mathbf{p}''_i(X_{C(i)}) = \mathbf{p}'_i + \underbrace{X_{C(i)}}_{\text{blob height}} \underbrace{B_\alpha(d(i,C(i)))}_{\text{Decaying coefficient}} \mathbf{v}_i, \qquad (3)$$

where $d(i,C(i))$ denotes geodesic distance on the rest mesh S^0 between vertices i and $C(i)$.
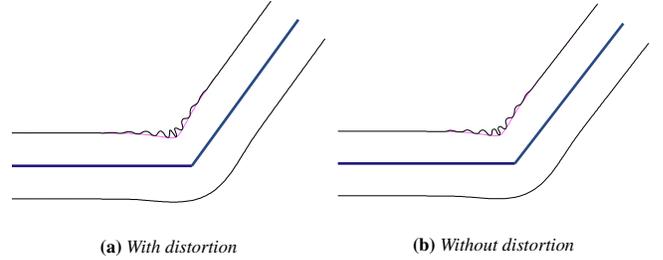
Our stretching function now reads

$$\tau_k(\mathbf{X}) = \left(\sum_{i=l_k}^{r_k-1} \left\| \mathbf{p}''_{i+1}(X) - \mathbf{p}''_i(X) \right\|_2 \right) - L_k^0,$$

where $L_k^0 = \sum_{i=l_k}^{r_k-1} \ell_i^0$ is the length of the kth fold on the rest mesh S^0, and l_k and r_k are the left and right bound indexes of the fold vertices, respectively.

Avoiding distortion for quasi-isometry Eventhough our blobs are guaranteed to have a constant length, vertices located on them may compress or dilate. This may cause two types of visual disturbance: distortion of the general fold shape (getting away from the intended shape designed by the user), and distortion when textured. To limit such distortions, we modify the directions $\mathbf{v}_i$ that vertices are allowed to follow. Initially, each one of this direction is chosen to be the outward normal of the skinned mesh S' at vertex i. During animation, the direction $\mathbf{v}_i^{t+1}$ at time $t + 1$ is corrected using the resulting fold $k = C(i)$ obtained at previous time step t from $X_{C(i)}^t$. More specifically, since the fold at time t has exactly length L_k, we resample it using the same metrics as the one on the initial mesh S^0, yielding new vertices $\mathbf{p}_i^{iso,t}$, and update the new directions $\mathbf{v}_i^{t+1}$ as

$$\mathbf{v}_i^{t+1} = normalized(\mathbf{p}_i^{iso,t} - \mathbf{p}_i''^{,t}).$$

The result of this correction is illustrated in Figure 4.

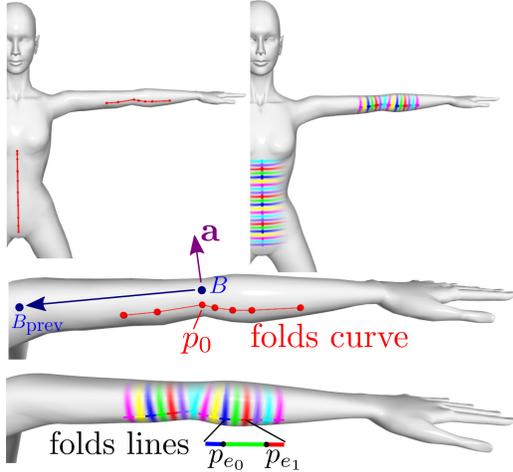**(a)** *With distortion* **(b)** *Without distortion*

Figure 4: *Distortion correction is especially useful at high bending angles.*

Super fast solver Our algorithm boils down to cancelling N independent error functions $\xi_k(X_k) = \|\Phi_{FB}(X_k)\|_2$, each one being function of a scalar variable X_k. We have moreover observed in all our tests that the Fischer-Burmeister function is particularly well-suited in our case, since our error function always admits a unique vanishing minimum, located in a convex area (see an illustration on the 2d bending arm in Figure 3b). This means that our complementarity problem (2) admits a unique solution. Moreover, since the problem is one-dimensional and locally convex, it can be solved very efficiently using a reasonable descent direction (provided in our case by BFGS, though the gradient method also suffices most of the time), combined with a good linear-search. Like previously, we use a zero subgradient at the unique nonsmooth point[1].

We have tested several linear-search methods, and favoured the approach by Moré and Thuente [1994], which relies upon quadratic and cubic interpolation of the function for optimizing some curvature condition. With this optimizer, we have always observed exact convergence (i.e., with exactly 0 residue) in 1 or 2 iterations only, starting from a positive initial guess (not necessarily close to the solution). This means that wherever our modified skinning algorithm is applied to in the animated sequence, it always converges extremely quickly to the solution.

---

[1]Note that we could have minimized instead the square norm of the Fischer-Burmeister function, to avoid singularity; in practice we did not note any difference between the two appraoches.

5 Extension to 3D surfaces

Figure 5: *View of the folds curves (red curves) and their subdivision into folds lines (colored curves). The folds lines are used to defined the weights ω, seen as the varying color on the mesh, distributing the deformation on the surface.*

Extending the 1D formulation into a mesh surface embedded in 3D space requires the following extensions. First, the locus where the folds are applied should be computed, we call them *folds curves*. The folds curves are defined to be curves made of consecutive mesh edges, and are precomputed once. During animation, the surface is deformed and these folds curves may be extended or compressed with respect to the rest state of the object.

Second, the curves are regularly subdivided and each subdivision k defines the locus of an individual fold, called *fold line*. This leads to the 1D problem of finding the height X_k associated to each *fold line k* with known rest and deformed length, and allows us to solve for the unknown X_k using the unilaterally incompressible approach described previously.

Finally, the height X_k is propagated in a neighborhood of the fold line using precomputed weights provided by the user defined blob, while guaranteeing that each fold curve that has been compressed by the skinning deformation is strictly isometric to the one on the rest shape. We describe in the following the first and last step which both correspond to pre-computations in our method.

5.1 Computing the fold line

Let us consider the surface to be given by a mesh with vertices $\mathcal{V}$ and deformed by skinning using an animation skeleton, whose animation is supposed to be predefined. The user can first select a subset of bones of the skeleton potentially associated with one or more folds curves. The manual selection allows for instance to differentiate bones deforming skin regions, where folds should not be applied, to bones deforming tight cloth regions where we aim to add folds. Moreover, the user can set the length of the fold line on the rest shape associated to each selected bone. For the sake of simplicity, we suppose that the selected bone is such that it is roughly aligned with its parent.

For all the selected bones at position B, we compute the local angle $\theta > 0$ and axis of rotation **a** with respect to the parent bone at position B_{prev}. For every angle θ greater than some user threshold, $10°$ in our implementation, we consider that the deformation is sufficient to initiate the generation of an associated fold curve computed as follows.

First, an initial vertex p_0 is selected as the closest vertex of the bone

located in the inward region of the bending deformation characterized by the direction $\mathbf{c} = \mathbf{a} \times (B_{\text{prev}} - B)$. Then p_0 is given by

$$p_0 = \underset{p \in \mathcal{V}, (p-B) \cdot \mathbf{c} > 0}{\operatorname{argmin}} \| \operatorname{proj}_P(p) - B \| ,$$

where $\operatorname{proj}_P(p)$ is defined as the projection of the vertex p onto the plane passing by B with normal $\mathbf{c}$. Then, the vertices belonging to the folds curve are assembled in marching along the edges of the mesh and selecting the neighboring vertex whose edge is the most aligned with $\pm(B_{\text{prev}} - B)$ until reaching the expected curve length.

Finally, this process is repeated for all, or, at least on a subset of the animation frames, and a fold curve is stored if its minimal distance to all other existing fold curve is greater than the width of a bump which is defined by the user.

5.2 Propagating the deformation on the surface

The last step consists in computing the weights $\omega(p)$ expressing the influence of a blob on a given vertex p. We compute this weight using two component called respectively *linear $\omega_l(p)$* and *orthogonal $\omega_o(p)$* with respect to a given fold line. First we consider the fold line, and more specifically, the edge belonging to the fold line with the closest orthogonal distance with respect to the vertex p. We call $\operatorname{proj}_L(p)$ the orthogonal projection of the vertex p onto this edge and (p_{e_0}, p_{e_1}) the two vertices defining this edge. The weights are computed as follows,

$$\omega_l(p) = B_\alpha \left(\frac{\| \operatorname{proj}_L(p) - p_{e_0} \|}{\| p_{e_1} - p_{e_0} \|} \right) .$$

$$\omega_o(p) = B_\alpha \left(\frac{\| p - \operatorname{proj}_L(p) \|}{\sigma} \right) ,$$

where σ corresponds to the width of the fold, and may be defined by the user for each fold curve. Finally, $\omega(p) = \omega_l(p) \, \omega_o(p)$.

6 Evaluation and results

6.1 Framework

We have tested our modified skinning approach on two different scenarios:

- A *bending cylinder* example, with high resolution, and mesh edges aligned with the bending direction and the transverse direction. In this simple example, we did not have to make precomputation steps listed in Section 5, since we directly relied upon the well-aligned mesh for extracting all necessary elements such as the fold line.

- A *dancer* example, whose mesh was arbitrarily generated and skinned by an artist, who provided us with the (low resolution) mesh sequence as well as the skeleton. For this more difficult example, we had to apply all steps of Section 5 before being able to apply our 1D unilaterally incompressible algorithm.

Final results are presented in Figure 1 as well as in our accompanying video. In the following, we thoroughly evaluate our approach in terms of physical realism, computational performance, and user-control, before discussing the limitations of our model.

6.2 Evaluation

Realism   With our method we are able to automatically recover the ringing wrinkle patterns forming at joints, with some realistic decaying of the fold height as one goes away from the joint center. As shown in our accompanying video and in Figure 2, our result qualitatively matches reality, and drastically improves standard skinning.

Table 1: *Computational time for our examples*

Example	$f_w{}^3$	$n_{\mathrm{subd}}{}^3$	Time ^{1}	Time ^{3}	**Time** ^{3}
Cylinder	1 to 3	0	16	-	**16**
Dancer1	3	2	16	20	**30**
Dancer2	2.2	3	16	70	**115**
Dancer3	1.5	4	16	265	**440**

^{1} Time for 1 frame (ms) for standard skinning without loop subdivision
^{2} Time for 1 frame (ms) for standard skinning with loop subdivision
^{3} Time for 1 frame (ms) for our **modified skinning**
^{3} Fold width (cm)
^{4} Minimum number of subdivision loops required

Comparison with physics-based simulation   To better assess the quality of our results, we compared our wrinkling patterns with those generated by a purely physically-based simulation, on the bending cylinder scenario. First, let us say that setting up the simulation was already a challenging problem. Indeed, we had to be able to capture a high-resolution pre-stretched garment with all of its vertices subject to unilateral contact with the underlying arm (materialized by the skinned mesh S') and with static Coulomb friction (sliding being forbidden). To get as close as possible to these requirements, we chose the recent simulator by Daviet and colleagues [2015] which models exact Coulomb friction at vertices, in a robust and efficient way. With this approach, and using a high friction coefficient $\mu = 0.6$, we were indeed able to capture the dual taking-off/sticking phenomenon which typically occurs during the bending of the cylindrical arm (see reference in our accompanying video).

However, we were not able to start from a pre-stretched garment: the problem became ill-conditioned and the solver failed to converge at some point. Finally, after many attempts we had to settle for a simpler scenario with the rest garment pose equal to the skinned arm S'. After struggling a few days with the right choice of material parameters (a full simulation sequence took 90 minutes to be computed), we finally managed to reproduce the expected ringing patterns, at least in the first part of the simulation (see Figure 2c). Then, when the arm started to bend to much, folds disappeared and merged, because of some too high stretching parameter. Decreasing the stretching stiffness however caused the initial folds to propagate quasistatically, with a propagation speed directly related to the stretching stiffness. We concluded that without any unilateral stretch limiting, physical simulation with standard elasticity failed to reproduce skin-tight garments properly. We have noted that such a physics-based model, incorporating a complementarity constraint for preventing compression beyond some threshold, was proposed in the past [Thanh and Gagalowicz 2011] in the case of loose garments, but never coupled to a contact solver with static friction.
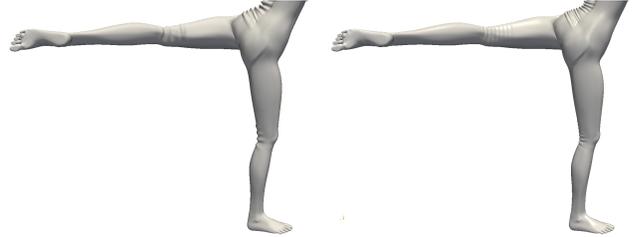
Performance   All our results were computed on a single-threaded application running on a recent laptop featuring Intel(R) Core(TM) i7-5600U @2.60GHz processors.

Our results are given in the table below. For the cylinder example, it is noteworthy that the cost of our modified skinning is almost negligible compared to standard skinning. For the dancer example, to create thin folds, a number of subdivision loops is required, which is the main source of overhead. Still, our results remain interactive even for the thinner folds scenario, which requires 4 subdivision loops of the initial mesh.

In all our examples, our solver converged in 100% of the problems resolved to maximum precision, in one single iteration in most cases, and 2 iterations at most.

Let us also mention that physics-based simulation is totally out of interactive ranges, since for the cylinder example, one frame took 1.30 minute on average to be computed, that is more than 5,000 times more than our method.

User-control   Our method also allows for an intuitive control of the wrinking patterns. Indeed, the user can freely choose the fold general profile, the length of the folding line, as well as the fold width (directly related to the folding density). Figure 6 and our accompanying video depict the effects of tuning the fold width on the overall shape of the character.

Figure 6: *Tuning the fold width parameter.*

6.3   Discussion and limitations

Overall, our method provides a fast and intuitive way to improve the visual realism of skin-tight garments compared to existing interactive techniques. By mixing simple geometrical considerations with physical constraints through the design of a well-posed optimization problem, we have shown that increasing realism does not necessarily imply losing interactivity.

However, our method could be improved in a number of ways. First of all, we have seen that the current bottleneck of our approach is not our automatic algorithm for creating folds where needed (which is almost inexpensive), but rather the subdivision steps that are required to refine the input mesh at a resolution suitable for adding realistic folds. Thin folds in particular do require a fine input mesh. Also, possible distorted parameterization of the input mesh may alter the final shape of the created wrinkle patterns. To free ourselves from the input mesh resolution and parameterization, one idea would be to map the high-resolution cylinder model locally onto the regions labeled for correction. Since our method only raises vertices up compared to the initially skinned mesh, applying the corrected cylinder patch on top of those regions would perfectly fit without causing any visual artifact.

Then, similarly to standard skinning, our algorithm is currently unable to avoid self-penetration. In some cases where a joint bends strongly, our modified skinning algorithm may yield more visually disturbing artifacts than the standard technique, since self-penetrating folds become quite visible. One (non-trivial) idea to solve this problem would be to merge contacting folds by making a continuous geometric transition from two blobs to a single one.

Finally, our method relies upon folding lines which are precomputed and remain constant over the animation. Computing them on the fly, dealing with some dynamic changes in their direction, and with possible intersections with other folding lines, are all challenging problems that we plan to investigate in future work.

7   Conclusion

We propose a new approach for making traditional skinning better-suited for skin-tight garments, by adding some physical constraint on top of the standard skinning formulation. Our method is able to generate realistic and controllable wrinkle patterns interactively,

while avoiding the computational burden and cumbersome parameter settings inherent to physics-based cloth simulation.

One natural extension of this work would be to deal with tight but non pre-stretched garments, made of hardly stretchable fabric (such as denim slim jeans or tight cotton tee-shirts). Such garments feature more complex wrinkling patterns, which are generated not only by bending but also by twisting motion (see accompanying video). More favorably however, garments still deform quasi-statically, and depict some characteristic wrinkling patterns which may be parameterized compactly. One promising work in that direction was [Decaudin et al. 2006], which explictly modelled the diamond folding patterns typical of stiff cylindrical meshes under compression. This gives us the hope to build an interactive system allowing for a smooth and realistic transition between the two regimes.

References

AUDOLY, B., AND POMEAU, Y. 2010. *Elasticity and Geometry: from hair curls to the nonlinear response of shells*. Oxford University Press.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Computer Graphics Proceedings (Proc. ACM SIGGRAPH'98)*, 43–54.

BRIDSON, R., FEDKIW, R., AND ANDERSON, R. 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph. 21*, 3, 594–603.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH - EG Symposium on Computer Animation (SCA'03)*, ACM-EG SCA, 28–36.

CUTLER, L. D., GERSHBEIN, R., WANG, X. C., CURTIS, C., MAIGRET, E., PRASSO, L., AND FARSON, P. 2005. An art-directed wrinkle system for CG character clothing. *SCA*.

DAVIET, G., AND BERTAILS-DESCOUBES, F. 2016. A Semi-Implicit Material Point Method for the Continuum Simulation of Granular Materials. *ACM Transactions on Graphics 35*, 4 (July), 13.

DAVIET, G., BERTAILS-DESCOUBES, F., AND CASATI, R., 2015. Fast Cloth Simulation with Implicit Contact and Exact Coulomb Friction. ACM SIGGRAPH / Eurographics Symposium on Computer Animation, Aug. Poster.

DECAUDIN, P., JULIUS, D., WITHER, J., BOISSIEUX, L., SHEFFER, A., AND CANI, M.-P. 2006. Virtual Garments: A Fully Geometric Approach for Clothing Design. *EUROGRAPHICS 25*, 3.

ENGLISH, E., AND BRIDSON, R. 2008. Animating developable surfaces using nonconforming elements. *ACM Transactions on Graphics (Proc. of SIGGRAPH'08) 27*, 3, 1–5.

FERRIS, M. C., AND KANZOW, C., 1998. Complementarity and related problems: A survey.

FISCHER, A., AND JIANG, H. 2000. Merit functions for complementarity and related problems: A survey. *Comput. Optim. Appl. 17*, 2-3 (Dec.), 159–182.

GILLETTE, R., PETERS, C., VINING, N., EDWARDS, E., AND SHEFFER, A. 2015. Real-time dynamic wrinkling of coarse animated cloth. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '15, 17–26.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient simulation of inextensible cloth. In *ACM Transactions on Graphics (Proc. ACM SIGGRAPH'07*, ACM, New York, NY, USA, SIGGRAPH '07.

GRINSPUN, E., HIRANI, A., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete Shells. In *ACM SIGGRAPH - EG Symposium on Computer Animation (SCA'03)*, ACM-EG SCA, 62–67.

GUAN, P., REISS, L., HIRSHBERG, D. A., WEISS, A., AND BLACK, M. J. 2012. DRAPE: DRessing Any PErson. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH 31*, 4.

HADAP, S., BANGERTER, E., VOLINO, P., AND MAGNENAT-THALMANN, N. 1999. Animating wrinkles on clothes. *IEEE Proceedings on Visualization*, 175–182.

HAHN, F., THOMASZEWSKI, B., COROS, S., SUMNER, R. W., COLE, F., MEYER, M., DEROSE, T., AND GROSS, M. 2014. Subspace Clothing Simulation Using Adaptive Bases. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH 33*, 14.

JACOBSON, A., AND SORKINE, O. 2011. Stretchable and Twistable Bones for Skeletal Shape Deformation. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH Asia 30*, 6.

KAVAN, L., AND ZARA, J. 2005. Spherical blend skinning: a real-time deformation of articulated models. *I3D*.

KAVAN, L., COLLINS, S., ZARA, J., AND O'SULLIVAN, C. 2008. Geometric Skinning with Approximate Dual Quaternion Blending. *ACM Transaction on Graphics 27*, 4.

KAVAN, L., GERSZEWSKI, D., BARGTEIL, A. W., AND SLOAN, P.-P. 2011. Physics-Inspired Upsampling for Cloth Simulations in Games. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH 30*, 4.

KIM, D., KOH, W., NARAIN, R., FATAHALIAN, K., TREILLE, A., AND O'BRIEN, J. 2013. Near-exhaustive precomputation of secondary cloth effects. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH 32*, 4.

KRY, P. G., JAMES, D. L., AND PAI, D. K. 2002. EigenSkin: real time large deformation character skinning in hardware. *SCA*, 153–159.

LARBOULETTE, C., AND CANI, M.-P. 2004. Real-Time Dynamic Wrinkles. *Computer Graphics International*.

MAGNENAT-THALMANN, N., LAPERRIÈRE, R., AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. *Graphics Interface*, 26–33.

MORÉ, J. J., AND THUENTE, D. J. 1994. Line search algorithms with guaranteed sufficient decrease. *ACM Trans. Math. Softw. 20*, 3 (Sept.), 286–307.

MÜLLER, M., AND CHENTANEZ, N. 2010. Wrinkle meshes. *SCA*, 85–92.

NARAIN, R., GOLAS, A., CURTIS, S., AND LIN, M. C. 2009. Aggregate dynamics for dense crowd simulation. *ACM Transactions on Graphics 28*, 5 (Dec.), 1.

NARAIN, R., GOLAS, A., AND LIN, M. C. 2010. Free-flowing granular materials with two-way solid coupling. *ACM Transactions on Graphics 29*, 6 (Jan.), 1–10.

OTADUY, M. A., TAMSTORF, R., STEINEMANN, D., AND GROSS, M. 2009. Implicit contact handling for deformable objects. *Computer Graphics Forum 28*, 2, 559–568.

RÉMILLARD, O., AND KRY, P. G. 2013. Embedded thin shells for wrinkle simulation. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH 32*, 4.

ROHMER, D., POPA, T., CANI, M.-P., HAHMANN, S., AND SHEFFER, A. 2010. Animation Wrinkling: Augmenting Coarse Cloth Simulations with Realistic-Looking Wrinkles. *ACM Transactions on Graphics (TOG). Proceedings of ACM SIGGRAPH ASIA. 29*, 5.

THANH, T., AND GAGALOWICZ, A. 2011. *Computer Vision/Computer Graphics Collaboration Techniques: 5th International Conference, MIRAGE 2011, Rocquencourt, France, October 10-11, 2011. Proceedings.* Springer Berlin Heidelberg, Berlin, Heidelberg, ch. A New Buckling Model for Cloth Simulation, 251–261.

THOMASZEWSKI, B., PABST, S., AND STRASSER, W. 2009. Continuum-based strain limiting. *Computer Graphics Forum (Proc. Eurographics'09) 28*, 2 (apr).

TURCHET, F., FRYAZINOV, O., AND ROMEO, M. 2015. Extending Implicit Skinning with Wrinkles. *CVMP.*

VAILLANT, R., BARTHE, L., GUENNEBAUD, G., CANI, M.-P., ROHMER, D., WYVILL, B., GOURMEL, O., AND PAULIN, M. 2013. Implicit Skinning: Real-Time Skin Deformation with Contact Modeling. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH 32*, 4.

WANG, R. Y., PULLI, K., AND POPOVIC, J. 2007. Real-time enveloping with rotational regression. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH 26*, 3.

WANG, H., HECHT, F., RAMAMOORTHI, R., AND O'BRIEN, J. 2010. Example-Based Wrinkle Synthesis for Clothing Animation. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH 29*, 4.

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

WEBER, O., SORKINE, O., LIPMAN, Y., AND GOTSMAN, C. 2007. Context-Aware Skeletal Shape Deformation. *EUROGRAPHICS 26*, 3.

XU, W., UMENTANI, N., CHAO, Q., MAO, J., JIN, W., AND TONG, X. 2014. Sensitivity-optimized Rigging for Example-based Real-time Clothing Synthesis. *ACM Transaction on Graphics, Proc. ACM SIGGRAPH 33*, 4.

ZHU, Y., SELLE, A., EMPEY, M., TAMSTORF, R., TERAN, J., AND SIFAKIS, E. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH 30*, 4.

ZURDO, J. S., BRITO, J. P., AND OTADUY, M. A. 2013. Animating Wrinkles by Example on Non-Skinned Cloth. *IEEE TVCG 19.*