True Seams: Modeling Seams in Digital Garments

ALEJANDRO RODRÍGUEZ and GABRIEL CIRIO, SEDDI, Spain

Fig. 1. Applying True Seams to a denim dress dramatically increases the realism of the garment. As input, our algorithm takes the 2D sewing pattern and the assembled 3D garment, as well as seam and stitch construction information. It then automatically computes the geometry of seams, dealing with overlaps and asymmetries, by folding and stacking together layers of fabric following the same construction process as true, real-life seams. The inset shows the dress before applying True Seams to it.

Seams play a fundamental role in the way a garment looks, fits, feels and behaves. Seams can have very different shapes and mechanical properties depending on how fabric is overlapped, folded and stitched together, with garment designers often choosing specific seam and stitch type combinations depending on the appearance and behavior they want for the garment. Yet, virtually all 3D CAD tools for fashion and visual effects ignore most of the visual and mechanical complexity of seams, and just treat them as joint edges, their simplest possible form, drastically limiting the fidelity of digital garments. In this paper, we present a method that models seams following their true, real-life construction. Each seam brings together and overlaps the fabric pieces to be sewn, folds the fabric according to the type of seam, and stitches the resulting assembly following the type of stitch. To avoid dealing with the complexities of folding in 3D space, we cast the problem into a sequence of simpler 2D problems where we can easily shape the seam and produce a result free of self-intersections, before lifting the folded geometry back to 3D space. We run a series of constrained optimizations to enforce spatial properties in these 2D settings, allowing us to treat asymmetric seams, gatherings and overlapping construction orders. Using a variety of common seams and stitches, we show how our approach substantially improves the visual appearance of full garments, for a better and more predictive digital replica.

CCS Concepts: • **Computing methodologies** → **Mesh geometry models**; *Physical simulation.*

Additional Key Words and Phrases: seams, cloth, modeling, geometry, optimization

ACM Reference Format:
Alejandro Rodríguez and Gabriel Cirio. 2022. True Seams: Modeling Seams in Digital Garments. *ACM Trans. Graph.* 41, 4, Article 62 (July 2022), 16 pages. https://doi.org/10.1145/3528223.3530128

Authors' address: Alejandro Rodríguez, alejandro.rodriguez@seddi.com; Gabriel Cirio, gabriel.cirio@seddi.com, SEDDI, Madrid, Spain.

0730-0301/2022/7-ART62 $15.00
https://doi.org/10.1145/3528223.3530128

1 INTRODUCTION

Modeling, simulating and rendering digital garments has become a staple in Computer Graphics, often making us wonder if we are not looking at a photograph instead of a synthetic image. However, a key ingredient is missing to achieve a truthful and trustworthy depiction of a real garment's draping and visual appearance: seams. Seams define the interface between fabric pieces, and are therefore

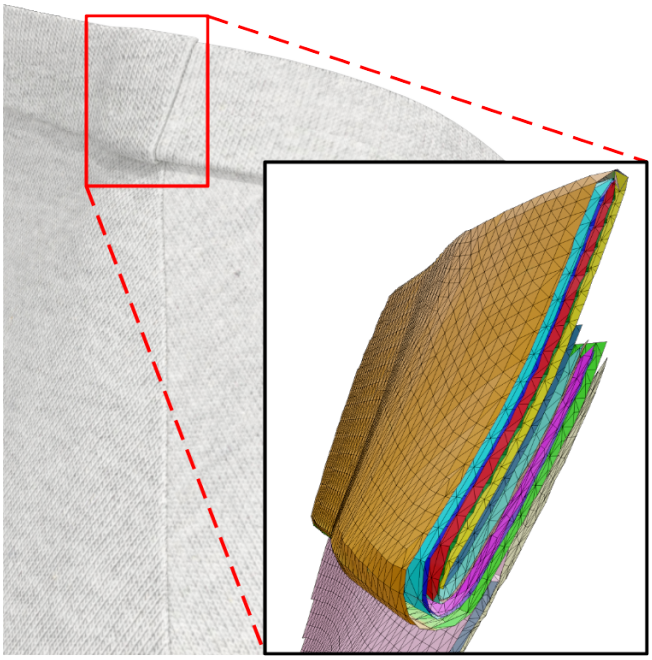


Fig. 2. A rather innocent looking neck can actually hide an awful lot of complexity: 14 layers stacked on top of each other after three simple sewing operations (a piece fold and two SSa 1.01 seams)

centerpiece in the design and construction process of a real garment. Seams play a major role in the appearance and even the style of a garment, as well as in its drapability, wearability and durability.

Yet, modern 3D CAD software and related garment design tools largely ignore seams, treating them as mere junctions between pieces in the simplest possible way. Currently, seams are defined as shared edges between piece boundaries, using constraints or shared degrees of freedom, and are often governed by a specific bending energy at these boundary hinges with its own bending stiffness [Pabst et al. 2008]. In reality, seams are complex assemblies of intertwined and overlapping layers of fabric held together by a variety of folds and stitching threads. The myriad of possible combinations between fabric, seam and stitch types produce an endless range of geometric shapes and mechanical properties which are specifically sought after (and sometimes explicitly avoided) by the garment designer because of their impact on the appeal, the fit, the technical properties or the comfort of the garment. Even the order in which the sequence of seams are applied during garment construction can significantly influence the resulting garment, but this aspect is also neglected in existing solutions. Correctly modeling seams to capture their complex behavior is therefore paramount. Trustworthy garment draping and appearance are needed to take digital fashion tools beyond the production of marketing material and visual effects, and into the actual process of garment design where key decisions can be made without resorting to physical prototypes.

Unfortunately, modeling seams can quickly become a combinatorial nightmare. There are eight different classes of seams according to the technical literature [ISO 4916:1991 1991], each for a different use-case, with many seam types within each class, amounting to more than 100 different types of seams. There are also more than 20 stitch types spread across six different classes [ISO 4915:1991 1991]. In addition, even the simplest garment (e.g. a t-shirt) can have different *overlapping* seaming operations, each with a different seam and stitch type. Notably, the complexity in the armpit and neck areas can sometimes escalate to more than 10 layers of fabric stacked on top of each other, as shown in Figure 2, and half a dozen folds and stitch threads holding everything in place. Even worse, the tightly packing together of narrow strips of fabric with many folds and layers is the perfect setting for the emergence of devastating self-intersections that would not only generate visual artifacts, but would also prevent the robust simulation of the resulting geometry. Finally, both sides of a seam are often asymmetric due to different lengths, different shapes, or both, adding to the overall complexity.

In this paper, we present the first approach that models all the seams in a garment, including overlapping and asymmetric seams, following the same construction process as true, real-life seams. We generate seams by overlapping, folding and stitching together each set of fabric pieces involved in a seam, including previously sewn pieces. The key idea behind our method, which makes seam modeling tractable and efficient, is the casting of a complex 3D problem into a set of simpler and smaller 2D problems. We work directly with the garment's 2D sewing pattern, where the patternmaker has drawn the pieces of the garment, and lift the geometry to 3D only once all the seams have been processed. We enforce specific spatial properties on the sewing pattern through a set of constrained optimizations, which ultimately allow us to treat seam folding as very simple geometric operations safe from self-intersections.

Another important feature of our method is the treatment of seams in a garment as an ordered sequence of operations, where the order matters. By dealing with the combinatorial complexity of having different sewing orders produce different results, we allow designers to explore, correct and validate unique construction orders.

The final output of our algorithm, and the main contribution of this paper, is the generation of complex multi-layer seams that emerge naturally from a set of sewing instructions, dramatically improving the appearance of digital garments. In addition, for sufficiently fine discretizations the resulting geometry is intersection-free and therefore simulation-ready.

2 RELATED WORK

2.1 Seam Modeling and Simulation

In textile engineering, the interest in modeling and simulating seams has its roots in a large body of work that studies the effect of seams on draping. Most of this work relies on real experiments: studying static and dynamic drapes for seamed and unseamed fabrics to explain differences in drape coefficient [Chung 1999; Hu et al. 1997; Pabst et al. 2008], observing and understanding the interdependence between the bending stiffness of fabric around the seam and the type of seam or stitch [Sukran 2020], or determining the parameters that influence the bending behavior of a seamed strip of fabric [Hu

and Chung 2000], such as fabric thickness, seam thickness, and seam allowance.

In more recent years, with draping and seam puckering experiments increasingly relying on numerical simulations, seam modeling naturally started to receive more attention. Inui and collaborators [2001; 1998] model seams as two overlapping strips of fabric using a Finite Element (FEM) discretization with tensile, shear and bending energies. They study the visual appearance of seam puckering under different mechanical properties of the fabric. While puckering is simulated by shortening one seamline, other approaches [Mousazadegan et al. 2012] apply forces along the seamline or directly model the stitching thread as 1-d elements [Roland et al. 2015]. Yang [2014] has also relied on a FEM discretization to model seams as a single layer in order to study fabric draping for different seams in different positions, while changing fabric stiffness and mass to account for multiple layers.

Closer to our work, Hu and collaborators [2006] explicitly model fabric folds and stitching due to seams, and simulate the resulting geometry using mass-spring systems [Provot 1995] for both fabric and stitching thread. They compare three different seam types using lockstitch stitching and study the resulting assembly after applying different amounts of shrinkage. Their seams are limited to two pieces joining at a unique, flat, straight and symmetric seam, which is the simplest case and far from most seams in real-world garments. They later extend this work to whole garments [Ma et al. 2006], using remeshing to increase the resolution around seams. While they are able to produce garment drapings, their work suffers from the same limitations as the original paper, i.e. oversimplistic seams. A clear example is their treatment of overlapping seams: only the first seam is generated correctly, while the other one has to be treated as joint edges.

Modeling seams as joint edges has been the traditional way of treating seams in Computer Graphics. Virtually all cloth modeling and simulation papers treat seams as joint edges. Notably, Pabst and collaborators [2008] capture the bending stiffness of fabric seamed with three different types of seams, from which they derive a scaling factor for a custom bending energy that increases bending stiffness depending on the distance to the seamline. Nodal mass is increased to account for the different fabric layers that would result from the actual folding of the seam. The authors show evidence of the effect of seams on draping. Montes et al. [2020] use a specific energy at seam edges to simulate tensile stiffening due to seams. Since our work explicitly models the overlap and the folds of fabric in seams, seam-specific mechanical behaviors emerge naturally, and can generalize to any recursive combination of fabric, seam and stitch type. In addition, we do not need to introduce or modify any elastic energy formulation for seams, since the folded geometry of the seam naturally takes care of fabric stiffening and other nonlinear behavior.

2.2 Garment Editing

There has been a growing interest in the last decade around digital garment editing to automatize common and traditionally manual patternmaking tasks [Jhanji 2018]. There are a number of papers that modify or augment the sewing pattern and/or its corresponding 3D garment to solve specific problems. Our approach is in line with this body of work, since we augment the assembled 3D garment with seams using (and working on) the sewing pattern.

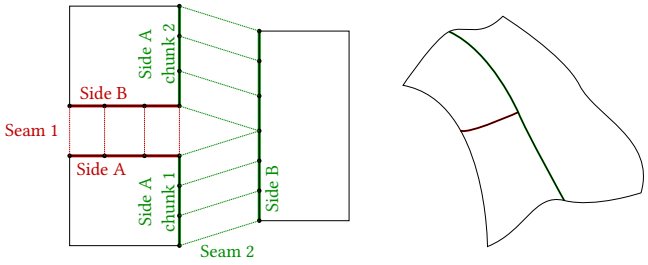


Fig. 3. In most CAD solutions, pieces are defined in the 2D *pattern space* (left). Seams are then defined by two sides of one or more chunks. This assembly is brought together in *world space* (right) during simulation by merging the seam sides, composed of a matching number of segments.

Lu et al. [2017] allow the placement and adjustment of prints directly on the 3D garment, with an automatic transfer to the sewing pattern. Similarly, Wolff and Sorkine-Hornung [2019] adjust the sewing pattern to allow for a better, more pleasing fit of the print along seams. They run a constrained optimization on the sewing pattern to satisfy symmetry and texture alignment constraints at seams. We also perform an optimization of the sewing pattern seeking spatial continuity along seams, but only as an internal step required for our algorithm, thus the original sewing pattern of the garment remains untouched. Their work is later improved to take into account reflection symmetries in pattern pieces [Wolff et al. 2019].

Combining garment editing with physically-based simulation, Keckeisen et al. [2004] allow sewing and cutting the assembled 3D garment in real time while the modifications are automatically applied to the underlying sewing pattern. Umetani et al. [2011] takes it a step further by allowing editing the other way around, from pattern to 3D, therefore enabling bidirectional garment design. Montes and collaborators [Montes et al. 2020] automatically alter digital tight-fitting clothing by optimizing sewing patterns to satisfy geometric and mechanical criteria such as body shape, pressure distribution and seam traction, among others. In FoldSketch [Li et al. 2018], users can specify folds and pleats through sketching, and the system then modifies both the 2D pattern and the 3D garment to match those specifications.

Pattern grading, i.e. adjusting the sewing pattern for different body sizes while preserving the style of the garment, has also been the focus of digital automation. Brouet et al. [2012] grade existing sewing patterns by reformulating pattern-grading criteria as a set of geometric constrains, where previous papers required the user to provide the set of geometric features that needed to be preserved [Meng et al. 2012]. These constraints are enforced in conjunction with physical plausibility constraints through the optimization of the 3D garment. Similarly, Bartle and collaborators [2016] allow grading as well as pattern mixing, and can additionally automatically compute the resulting sewing pattern right after optimizing the 3D garment. Wang [2018] improves upon these ideas by running

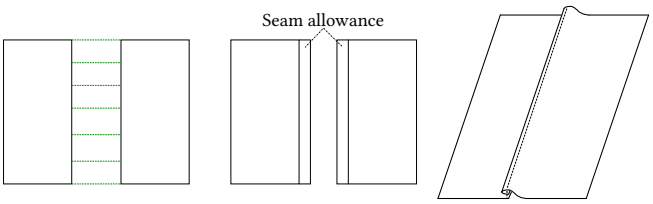


Fig. 4. A real-world seaming operation between two pieces (left) involves the *seam allowance*, an additional strip of fabric that extends beyond the seamline (middle), which is folded and stitched together (right).

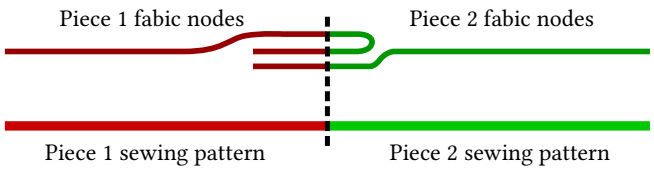


Fig. 5. Cross section view of two pattern pieces (bottom) meeting at a seam represented by the dotted line. Their corresponding fabric nodes (top), including the seam allowance, undergo folding and deformation. Then, a new support piece is found for the fabric nodes depending on which side of the seamline they fall, represented by matching colors. Fabric nodes can go through different support pieces several times depending on the folding complexity.

a constrained optimization for grading on the 3D garment and the sewing pattern simultaneously, therefore ensuring the existence and the correctness of the latter. Berthouzoz et al. [2013] can read existing sewing patterns and automatically create their corresponding 3D garment, assembled and simulated around an avatar, using machine learning and optimization techniques.

Seam allowance generation is a standard case of sewing pattern augmentation, and is a well documented process [Assembil 2013]. Since any professional patternmaking software knows how to add seam allowance to a sewing pattern, only a few approaches that require seam allowance generation provide guidance on how to do it on a discretized piece [Igarashi et al. 2008; Lu et al. 2017].

3 OVERVIEW

3.1 Definitions

In this section we define some common concepts of digital garments, many of which can be considered as standard in the field. A garment is composed of one or more *pieces*, following a sewing pattern made by a pattern maker. These pieces are typically defined in a 2D setting, that we call the *pattern space* (Figure 3, left). Each piece has a specific shape, size and position in this space. Since in the real world these pieces are cut out from fabric rolls, the geometry in *pattern space* also encodes the rest shape of each piece. A *seamline* groups together different piece edges (or sections of them) into one side of a seam. *Seams* can then be defined by matching two seamlines, which are then brought together during the assembly and simulation of the garment. A seamline can be as simple as one piece edge (e.g. each side of seam 1 in Figure 3), or can be made of several disconnected piece edge sections from one or more pieces that were previously sewn (e.g. side A of seam 2 in Figure 3). We call these sections *chunks*. During the virtual assembly and simulation of a garment, the pieces are discretized as triangle meshes. Seamlines become a sequence of discrete boundary edges, each of which we call a *seamline segment*, paying special care to have a correspondence between segments on each side of a seam.

Therefore, each seamline is a sequence of seamline segments and each segment has a corresponding segment on the other seamline of the seam. The simulation takes place in *world space* (Figure 3, right), where all segment pairs are brought together and overlap each other to give the garment its final shape. In the remainder of this paper, *along the seam* means parallel to the seamlines, while *across the seam* means in the direction of seamline normals, i.e. from one side of the seam to the other.

3.2 Seam Allowance and Seam Types

While the above definitions are generally true for most CAD garment software, they drastically simplify the real sewing process of garments and, while convenient, they neglect several important aspects of real seams.

In reality, two pieces are sewn through their *seam allowance*, an extra amount of fabric extending across each seamline (see Figure 4). The seam allowance specifically allows fabric folding and stitching at the seam without consuming fabric intended for the piece. To produce a seam, seamlines on both sides of the seam are overlapped, both seam allowances are folded following the seam type, and stitched together along and across the seamline following the stitch type.

The folds, overlaps, stitching threads and amounts of seam allowance are precisely what give each seam its particular properties, with different seam and stitch types used for different purposes. A flat Felled Seam (LSc 2.04.06) with two Lockstitches (301) is strong and durable, and is ideal for heavy fabrics in workwear garments, while a General Seam (SSa 1.01) with a 3 Thread Overedge (504) stitch is light and flexible, but somewhat weak. Sewing order is important as well: without proper care, overlapping seams could result in uncomfortable and visually noticeable bumps at seam junctions, affecting appearance and body feel. Correctly modeling and simulating seams is therefore paramount to obtain accurate and predictive virtual garments. In most existing solutions, all of these elements are either completely neglected or partially treated as an afterthought.

3.3 Method overview

The True Seams algorithm takes as input the output of a standard CAD garment software: a set of 2D sewing pattern pieces and their assembled shape in 3D space. Each piece is a triangle mesh, often coarse due to real time simulation constraints, and has a 1:1 mapping between its 2D and its assembled 3D geometry. In addition, and following the construction process of real-life garments, some information is required for each seam: the seam type and its specific variation (see the supplementary document accompanying this

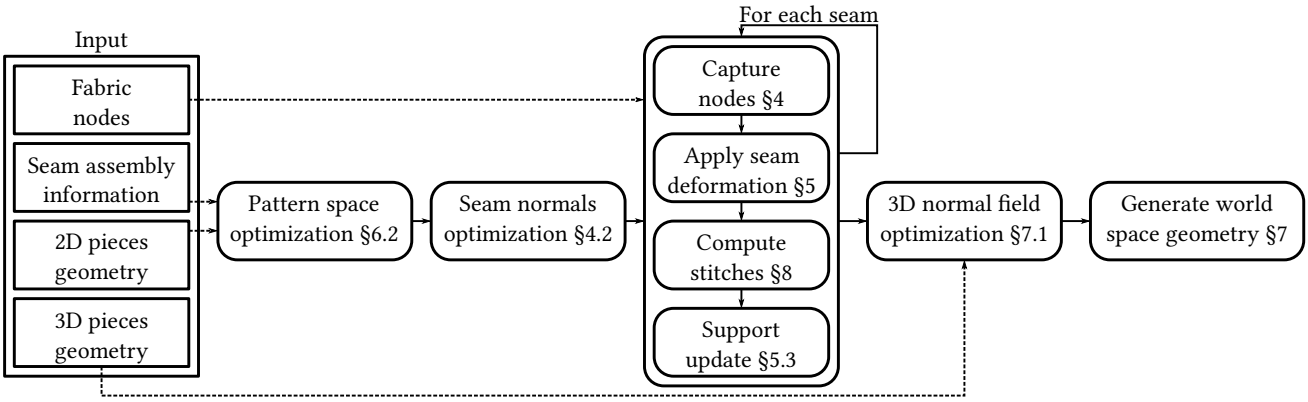


Fig. 6. Overview of the True Seams algorithm. Inputs are shown on the left, with dotted arrows pointing to the step where they are first needed. The initial steps of the algorithm are two geometric optimizations to prepare the 2D pieces and seams for the folding operations. Next, the seams are applied in order, each deforming the fabric nodes as left by the previous seam and adding the stitches. After all seams have been applied, the final geometry is lifted to 3D *world space*.

paper), the stitch type, the amount of seam allowance to be generated, and the construction order. With this information, how can we produce a geometric output with all the seams and stitches of a real garment?

Fabric Nodes. In the real world, fabric pieces are cut out of fabric rolls following the outline of sewing pattern pieces, previously extended by their seam allowances. Since seam allowance generation is standard in the patternmaking industry (see §2.2), we will not discuss it in this paper. We mimic the fabric cutting process and generate the 2D fabric pieces by starting from the outline of sewing pattern pieces with seam allowance. Then, for yarn-level cloth this outline is discretized as yarns and yarn nodes. For triangle mesh cloth, the outline is discretized as triangles and nodal vertices. We call these nodes *fabric nodes*. The True Seams algorithm is independent from the choice of fabric discretization as long as there is a nodal discretization with positions in *pattern space* and a high enough resolution to properly resolve fabric folds.

Support. Initially, by construction each fabric node falls inside or close to a sewing pattern piece: we say that sewing pattern pieces act as *support* of fabric nodes. We can compute the barycentric coordinates of a fabric node with respect to the triangle of the piece it is supported by. Subsequently, the 2D position of a fabric node in *pattern space* can be computed from the position of the vertices of the supporting triangle and the barycentric coordinates.

A key concept of our approach is that fabric nodes can travel from one piece to another in *pattern space* after being sewn, i.e. they can be supported by other sewing pattern pieces throughout the sewing process. Seam allowance nodes are the most common case: they are initially supported by the pattern piece they were cut out from, but after sewing and eventual folding they often end up on the other side of the seam, and therefore supported by another piece as shown in Figure 5.

Working in 2D space. It is tempting to use the 3D garment in *world space* as sole input to generate the seams. However, manipulating and folding surfaces in *world space* on top of a complex 3D mesh with arbitrary in-plane and out-of-plane deformation is far from trivial, especially when trying to avoid self-intersections. In addition, seam folding deformation is usually *plastic* due to ironing, while mesh draping deformation is mostly *elastic*. Correctly separating the two when working directly with the 3D mesh becomes very hard. Instead, we cast this complex 3D problem into a sequence of much simpler 2D problems. Key to our approach is the idea to work directly in *pattern space*, and lift the geometry to *world space* only once all the seams have been processed. In this way, we can guarantee that the input geometry is flat and undeformed, and produce intersection-free geometry after processing each seam. The 3D garment, however, has the advantage of forming a continuous space: one can naturally go from one piece to the other through a seam, since piece edges perfectly match at seams in 3D. *Pattern space*, on the other hand, is discontinuous. We therefore design a way to enforce continuity across pieces in *pattern space*, so that we can easily and safely overlap and fold fabric at piece edges. We call this continuous 2D space the *seamline space*, and we describe it in §4.

Pipeline. Figure 6 gives an overview of the different steps of our approach. Initially, all fabric nodes are supported by their respective pieces, including seam allowance nodes. Before treating the seams, we perform a series of optimizations in *pattern space* (§4.2 and §6.2) to enforce *seamline space* continuity. Then, we proceed seam by seam, following the garment's construction order. A seam will compute its *seamline space* and identify all fabric nodes that fall within the seamline area (§4), i.e. the nodes that will be deformed by the seam. We call this process *capturing* nodes. Captured nodes are then folded and deformed (§5) following the type of seam, and stitches are created on the folded geometry according to the type of stitch (§8). The support of fabric nodes is then updated, since some nodes might now be supported by a different sewing pattern piece after folding (§5.3), and the algorithm proceeds to the next seam. Once all seams are processed, the 1:1 mapping between pieces in *pattern space* and pieces in *world space* makes it very easy to

lift to 3D each fabric node (§7), using barycentric coordinates on supporting triangles and mapping that position to the 3D mesh.

4 CAPTURING IN *SEAMLINE SPACE*

Fabric nodes that are close enough to the seamlines will go through significant transformations due to folding and fabric overlap. For each seam, before folding or deforming any geometry, we must first find which nodes will be affected by the seam itself, and prepare those nodes for the folding operation. We call this process *node capturing*. In this section, we introduce the concept of *seamline space* to greatly simplify the capture and manipulation of nodes along and across seams. For the sake of clarity, we describe the algorithm for seams joining two pieces. We later extend it in §6 to support more complex seams with an arbitrary number of chunks and pieces on each side.

4.1 Seamline Space

While the two seamlines of a seam overlap in *world space* once the garment is assembled, they do not overlap in *pattern space*: each seamline usually belongs to a different piece, and pieces are laid flat in *pattern space* with some distance between each other to leave room for seam allowances and facilitate cutting. Therefore, at best, a rigid transformation is required to match both seamlines. Most of the time, however, there is no rigid transformation to align one seamline with the other, such as in the case of asymmetric seams, and only an elastic deformation can force them to match: in real life, the seamster would stretch or gather the fabric manually during stitching to force the overlap. The *pattern space* appears unsuitable to work at the seam level: it would be difficult to fold one seam side onto the other, because there is no *continuity* across seam sides. Instead, we need to define a continuous space around the seam that allows us to freely navigate along and across the seam, supporting rigid and, more importantly, non-rigid transitions between sides.

To this end, we design a parametric space along the seam that we call *seamline space*. We leverage the fact that, by construction, each segment of a seamline is paired to a segment of the other seamline of the seam, with both segments overlapping in *world space* as described in section §3.1. Each segment has endpoints $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$, with respective normals $\boldsymbol{N}_0$ and $\boldsymbol{N}_1$. We define the coordinate α to parameterize the position $\boldsymbol{p}(\alpha)$ of a point along the segment:

$$\boldsymbol{p}(\alpha) = \boldsymbol{x}_0 + \alpha(\boldsymbol{x}_1 - \boldsymbol{x}_0) \tag{1}$$

To parameterize the space around the segment, we define the coordinate μ as the signed distance to the segment and along the direction of the normal $\boldsymbol{N}$ resulting from the interpolation of endpoint normals $\boldsymbol{N}_0$ and $\boldsymbol{N}_1$ at α. Then, any point $\boldsymbol{p}_{2d}$ in *pattern space* on either side of the segment can be uniquely defined using *seamline space* coordinates (α, μ):

$$\boldsymbol{p}_{2d}(\alpha, \mu) = \boldsymbol{p}(\alpha) + \mu\boldsymbol{N}(\alpha) \tag{2}$$

By extension, any point around the *entire seam* can be uniquely defined using the segment number i and the *seamline space* coordinates (α, μ).

Using the *seamline space*, a point with (α, μ) coordinates for one seamline becomes $(\alpha, -\mu)$ for the other seamline of the seam, effectively defining a continuous space across the seam as shown in Figure 7. By ensuring segment normals are continuous along the seams, the resulting parametric space is also continuous along the entire seam.

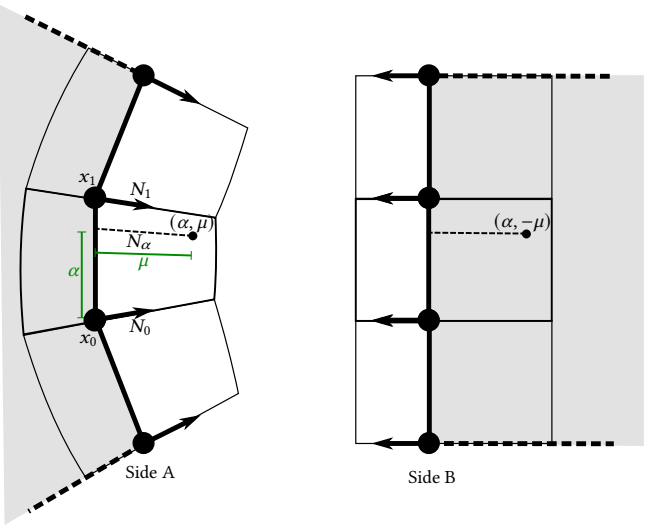


Fig. 7. The *seamline space* is defined by the seamline segments endpoints and their normals. Any point around the seamline, both towards the inside of the piece (highlighted in grey) or towards the seam allowance, can be defined by a segment index and two parameters α and μ, encoding respectively the position along the segment and the signed distance to the segment along the normal at α. Any point in a seamline has a direct, bijective mapping to the other seamline of the seam.

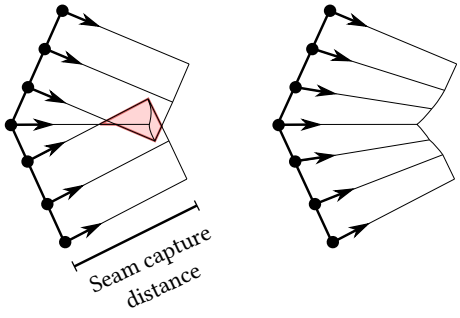


Fig. 8. Left: A naïve computation of the seamline normals can lead to ill-defined areas spanning several segments (in red) due to the intersection of the capture lines along the segment normals. Right: We address this issue by optimizing the seamline normals to prevent such intersections.

In order to track changes in height due to folds and overlaps, we add a third coordinate to the *seamline space*, the height h, which always follows the out-of-plane direction z. This height coordinate will be used in sections §5 and §7.

4.2 Seamline normal optimization

While the *seamline space* is continuous *across* the seam by construction, it is not the case for continuity *along* the seam. Since

Eq. (2) defines an offset curve from the seamline at a given μ, the offset curve is valid only if lines along the segment normals do not intersect each other, as illustrated in Figure 8.

This is a known and well studied problem in the context of offset curves and surfaces [Maekawa 1999; Pham 1992]. To avoid intersections, offset distances are often shortened in localized areas [Cohen et al. 1996; Peng et al. 2004; Porumbescu et al. 2005], which would result, in our context, in unrealistic compression of the fabric. Instead, we run a constrained optimization on the seam normals N to prevent intersections within the maximum capture distance μ_{max}. Let $\mathcal{I}$ be the set of all pairs of consecutive nodes in a seamline. For each pair $j \in \mathcal{I}$ of consecutive nodes v, w, we compute θ_{max} as the maximum allowed angle between their respective normals (see Eq. (10) in Appendix A.1) and define $f_j(N) = \text{acos}(N_v \cdot N_w) - \theta_{max}$. We then solve

$$\begin{aligned} &\min_N \sum_i \left\| N_i - \bar{N}_i \right\|^2 \\ &\text{s.t.} \quad f_j(N) < 0, \quad j \in \mathcal{I} \end{aligned} \tag{3}$$

to avoid intersections in the seamline normals while deviating as little as possible from the reference normals $\bar{N}$ of the nodes, defined by averaging the normals of incident segments. We obtain a well defined, intersection-free *seamline space* (Figure 8, right).

4.3 Capturing nodes

In order to capture a node to include in the seaming process, we iterate over the seamline segments of each seamline. Each seamline segment belongs to a single piece, and we therefore look for nodes supported by that piece that fall inside the segment's capturing region in pattern space. The capturing region of a segment is the offset surface swiped by $\boldsymbol{p}_{2d}(0 \le \alpha \le 1, -\mu^{int}_{max} \le \mu \le \mu^{ext}_{max})$ in Eq. (2). Both μ_{max} values (interior and exterior) are given by the seam type, since different seam types require different widths. There are therefore two capture regions per segment: one towards the inside of the piece (with negative μ) that captures interior nodes, and one towards the outside of the piece (with positive μ) that captures seam allowance nodes (see Figure 7).

In some cases, the capture region of a segment can reach across a previous seam, and requires capturing nodes supported by other pieces. Since the pattern space is not continuous, we use the *seamline space* of the previous seam to provide continuity across its sides. This effectively connects both *seamline spaces*, allowing to reach other pieces and capture their nodes.

4.4 Computing seamline space coordinates

If a node falls inside the capture region of a segment, we can compute its (α, μ) coordinates from its 2D position $\boldsymbol{p_{2d}}$ in *pattern space*. We recall that $\boldsymbol{p}(\alpha)$ is the parametric position along the segment, while μ is the signed displacement from $\boldsymbol{p}(\alpha)$ in the direction of $\boldsymbol{N}(\alpha)$. $\boldsymbol{N}(\alpha)$ results from the interpolation of endpoint normals $\boldsymbol{N}_0$ and $\boldsymbol{N}_1$ at α:

$$\boldsymbol{N}(\alpha) = \frac{(1-\alpha)\boldsymbol{N}_0 + \alpha\boldsymbol{N}_1}{\|(1-\alpha)\boldsymbol{N}_0 + \alpha\boldsymbol{N}_1\|} \tag{4}$$

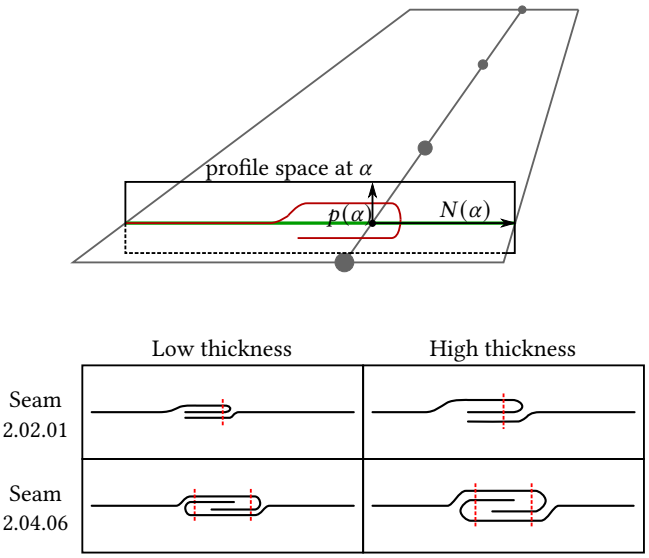


Fig. 9. Top: A seam profile at $p(\alpha)$ along the seamline defines a 2D space orthogonal to the *seamline space*, called *profile space*, easing the deformation of nodes captured at α during the folding operation. The green line represents the centerline, and becomes the red line when transformed through ρ. Bottom: Example of two seam types as defined by ISO 4916 parametrized to accommodate layers with different thicknesses.

Noting that $\boldsymbol{p}_{2d} - \boldsymbol{p}(\alpha)$ and the normal direction $(1-\alpha)\boldsymbol{N}_0 + \alpha\boldsymbol{N}_1$ are colinear and are both a function of α, we can solve the following system for α:

$$(1-\alpha)\boldsymbol{N}_0 + \alpha\boldsymbol{N}_1 = \lambda(\boldsymbol{p}_{2d} - (\boldsymbol{x}_0 + \alpha(\boldsymbol{x}_1 - \boldsymbol{x}_0))) \tag{5}$$

with two equations and two unknowns λ and α, which amounts to finding the roots of a quadratic polynomial on α. Once we know α, we find μ through Eq. (2).

5 FOLDING IN *PROFILE SPACE*

Once all relevant nodes have been captured by the seamline segments, we can proceed to the folding of the geometry according to the type of seam.

To this end, we generate reference frames along the seamline that we call *seam profiles*. These profiles define 2D spaces orthogonal to the *seamline space* that run across the seamline as shown in Figure 9. The origin of a *seam profile* at α is located at $\boldsymbol{p}(\alpha)$, the x axis is aligned with the seamline normal $\boldsymbol{N}(\alpha)$, and the y axis points upwards (matching the z axis of the *seamline space*). Therefore, a point in *seamline space* with coordinates (α, μ, h) has *profile space* coordinates (μ, h) in a *seam profile* at α. It is therefore trivial to go from one space to the other.

All folding operations happen in *profile space*. For any node captured by the seamline and with *seamline space* coordinates (α, μ, h), we generate a *seam profile* at α and compute the new coordinates (μ', h') resulting from the folding operation.

5.1 Folding operation

The *seam profile* allows us to focus on a slice of the seam across the seamline, reducing a 3D problem (folding a surface) to a 2D problem (folding a line) in *profile space*, which is a more manageable task.

However, properly folding both sides of the seam can become quite challenging when seam sides can have arbitrarily complex topologies. Indeed, seam sides can be flat, homogeneous pieces of fabric if never seamed before, but can also be the result of one or multiple previous seaming operations. In this case, a seam side can already have multiple complex folds and many layers of fabric from different pieces with varying height along and across the seam. Keeping track of these folds and layers would be a very complex endeavour. As it turns out, it would also be a pointless one.

We make the key decision to treat each side of the seam, before folding, as a flat plate with uniform thickness across the seam (but not along it, as explained in §5.2). We therefore reduce each side of the seam to its centerline ($h = 0$) plus a thickness, akin to the treatment of thin plates in continuum mechanics. Treating and folding seam sides as plates with uniform thickness has two key advantages. First, profiles become agnostic of the specific geometry (folds, layers, connectivity, etc) within each side of the seam: only the centerline and the thickness matter. Second, this guarantees that if the geometry within the plates was intersection-free, the resulting folded geometry will also be intersection-free.

The folding operation then simply becomes a transformation of the flat centerline coordinate μ to the folded centerline coordinates (μ', h') through a function ρ from $\mathbb{R}$ to$\mathbb{R}^2$ specific to each seam type. In order to account for the additional displacement due to the height h along the thickness direction of a point at μ, we also need the normal η to the function ρ at μ. The folding operation then transforms the input position (μ, h) on either side of the seam into its deformed, folded position (μ', h'):

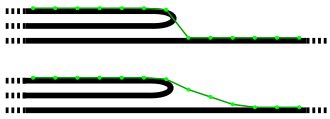
$$(\mu', h') = \rho(\mu) + h\eta(\mu) \tag{6}$$

Both ρ and η are parametric functions defined by the type of seam, and parameterized by the thickness of both plates.

Examples of seam types and their respective ρ functions in *profile space* are shown in Figure 9. Each side of the seam undergoes a different transformation, depending on the seam type. For seam type 2.02.01 [ISO 4916:1991 1991], for example, the left side is folded onto itself and over the right side. For seam type 2.04.06, both sides are folded onto themselves while clutched together, with the left side on top of the right side. In practice, ρ can be represented as a piece-wise continuous set of parametric primitives (lines, arcs), or as parametric spline for more artistic direction, while η can be derived from ρ.

5.2 Thickness heightmap

As explained in the previous section, we treat seam sides as flat plates with uniform thickness. While we do this at every seam profile, it does not mean that the thickness has to be uniform over the entire seamline. Quite the contrary: height usually varies significantly as the seamlines traverse previously folded seams. On the other hand, we must ensure some sort of smoothness in terms of thickness along the seamline. Otherwise, we would end up with sudden changes in thickness from one profile to the next, resulting in unnatural jumps in the folded geometry.



In order to compute the thicknesses along the seamline, we build a heightmap by sampling the height of the seamline at regular intervals along α. Specifically, for each sampling location we find the captured node with the largest value $h + t$ among the nodes whose α lie within the previous and next sampling locations (inset figure, top), and use it to set the heightmap at that location. Here t is half the fabric thickness of the captured node. Then, when a *seam profile* is generated at any given point along the seamline, its thickness is computed through a simple linear interpolation of the two closest heightmap samples.

To enforce the smoothness of profile thicknesses along the seamline, we simply smooth out the heightmap (inset figure, bottom) until all gradients satisfy a given *smoothness threshold*. We constrain the smoothing to only allow increases in height, since decreases could result in geometry intersections during folding. We used a fixed value of 0.3 for the *smoothness threshold*, although a more sophisticated approach could make it dependent on the bending stiffness of the fabric, since a higher stiffness will tend to flatten the surface.

5.3 Support update

Once all captured nodes have been folded in *profile space*, we can update their position in *seamline space* using their newly computed μ' and h' coordinates.

As a consequence, many nodes will have switched from one side of the seam to the other, and will be falling inside a piece belonging to the other side of the seam. If this is the case, we simply invert the sign of the μ' coordinate and reassign the node to the corresponding seamline segment on the other seamline. This doesn't change the position of the node in *seamline space*. However, it effectively moves the node from one piece to the other in *pattern space*, and therefore which piece is now supporting the node. When we compute its position $\boldsymbol{p}_{2d}$ using Eq. (2), the node will have switched pieces in *pattern space*.

Similar to what happens in the node capture process (§4.3), if a node ends up moving beyond a previous seam, we use the *seamline space* of the previous seam to compute the new position of the node in the other piece.

Once all support relationships have been updated, we can move on to the next seam.

6 MULTI-CHUNK SEAMS

Seams are not typically applied in isolation. After two pieces are sewn together, the resulting assembly can be sewn to another piece, or even another assembly. The simplest instance of this operation is shown in Figure 3 (top), but more complex cases are actually quite common and appear in most garments: the sleeve and neck pieces sewn to the torso of a sweatshirt (Figure 18) or the waistband seam of many pants (Figure 19). Thus, as already described in §3.1, a seamline will be typically composed of several chunks belonging

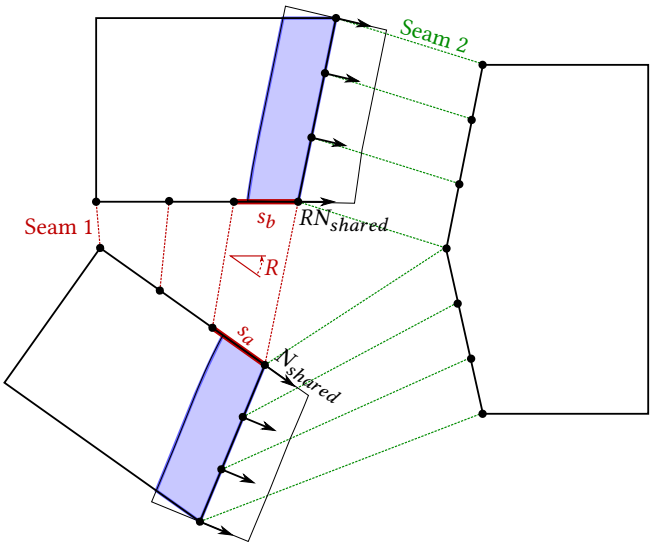


Fig. 10. The optimization of multi-chunk seam normals requires the merging of the normals at the transitions between chunks. The segments s_a and s_b of the previous seam connecting both chunks are used to compute the rotation R that aligns the normals.

to one or more pieces, leading to additional considerations for our method.

6.1 Chunks in normal optimization

Applying (3) directly when there is more than one chunk in a seamline leads to a well defined *seamline space* for each chunk independently, but it is not guaranteed to hold in the transitions between chunks. To guarantee continuity and proper definition of the *seamline space* between chunks, the last normal of one chunk must match the first normal of the next chunk under the corresponding rigid transformation. Otherwise, in the example of Figure 10, the optimization would lead to a discontinuity in the transition between chunks, failing to capture and fold the nodes in that area. To compute the rigid transformation R, we take the two segments s_a and s_b of the previous seam incident on the current seam (highlighted in red in Figure 10) and compute the rotation that aligns them, so that $R\frac{s_a}{\|s_a\|} = \frac{s_b}{\|s_b\|}$. During the optimization procedure, we treat both normals as one, initialized as $N_{shared} = \frac{N_1+R^T N_2}{\|N_1+R^T N_2\|}$, and adapt the constraints involving the shared normal generated by the first and second chunk to act on N_{shared} and RN_{shared} respectively. Figure 10 shows the optimized normals once the shared nodes are accounted for.

The same procedure is applied to cyclical seams (seams that start and end at the same location), where the first normal of the first chunk and the last normal of the last chunk must undergo the same treatment.

6.2 *Pattern space* optimization

When two pieces are joined by a seam, their seamlines become aligned. If the seamlines are asymmetrical (e.g., they have different lengths, as seam 1 in Figure 12), seamlines would be manually compressed and/or stretched by the seamster (or by the draping simulator in digital fashion) to get them aligned. Let's call this deformation *alignment deformation*. The *seamline space* handles these deformations implicitly for single seams. However, if the resulting assembly is affected by a latter seam (see seam 2 in Figure 12), this latter seam has no way to anticipate the *alignment deformation* of the previous assembly in pattern space (2D), since the *alignment deformation* exists only in the assembled garment (3D).

Ignoring this *alignment deformation* leads to discontinuities across chunks in *seamline space*, causing nodes to get miscaptured and receive deformations inconsistent with those of neighboring nodes (see Figure 12 left, where two neighboring nodes across seam 1 would get different μ values by the capture process of seam 2), or receive no deformation at all (see Figure 11), leading in both instances to visual artifacts.

To address this issue, we perform an optimization on the sewing pattern pieces as the first step of the True Seams algorithm, aiming to enforce a 2D version of this *alignment deformation*. Since this is only needed for the areas of a seam that overlap areas of a preceding seam, we first flag all seamline segments across all seams that require *pattern space* optimization, i.e. seam segments that fall within capture distance of seamline segments of latter seams. These segments (and their corresponding coupled segments on the other side of the seam) define the seamline sections that are required to be symmetrical in *pattern space*, as they will be traversed by the *seamline space* of latter seams.

From all the flagged segments, we define $\mathcal{I}_{norm}$ as the set of all pairs of coupled segments and $\mathcal{I}_{angle}$ as the set of all quadruplets of two contiguous segments and their corresponding coupled segments. For each pair $i \in \mathcal{I}_{norm}$ of coupled segments s and $\tilde{s}$ we define $f_i(x) = \|s\| - \|\tilde{s}\|$. Similarly, for each quadruplet $j \in \mathcal{I}_{angle}$ of two contiguous segments s_a and s_b and their corresponding coupled segments $\tilde{s}_a$ and $\tilde{s}_b$ we define $g_j(x) = angle(s_a, s_b) - angle(\tilde{s}_a, \tilde{s}_b)$. We then define an energy term $E_{shape}(x)$ measuring the deviation from the original *pattern space* geometry and solve

$$\begin{aligned} \min_x \quad & E_{shape}(x) \\ \text{s.t.} \quad & f_i(x) = 0, \quad i \in \mathcal{I}_{norm} \\ & g_j(x) = 0, \quad j \in \mathcal{I}_{angle} \end{aligned} \tag{7}$$

effectively enforcing symmetrical sides for the flagged sections while minimizing *pattern space* deformation. We set E_{shape} to simply measure deviations in edge length and triangle area with respect to the original geometry, but more complex energy terms could be devised, e.g. accounting for different fabric mechanical properties. In practice, only small sections at the end of some seamlines are flagged and require very little deformation. We have found this approach to be effective in all the examples we have tested so far, including exaggerated test scenarios, with the optimization converging rapidly even for high resolution pattern pieces (see section §9 for broken down timings). The asymmetric example of Figure 11, which uses curved pieces to create volume in the garments, shows the results

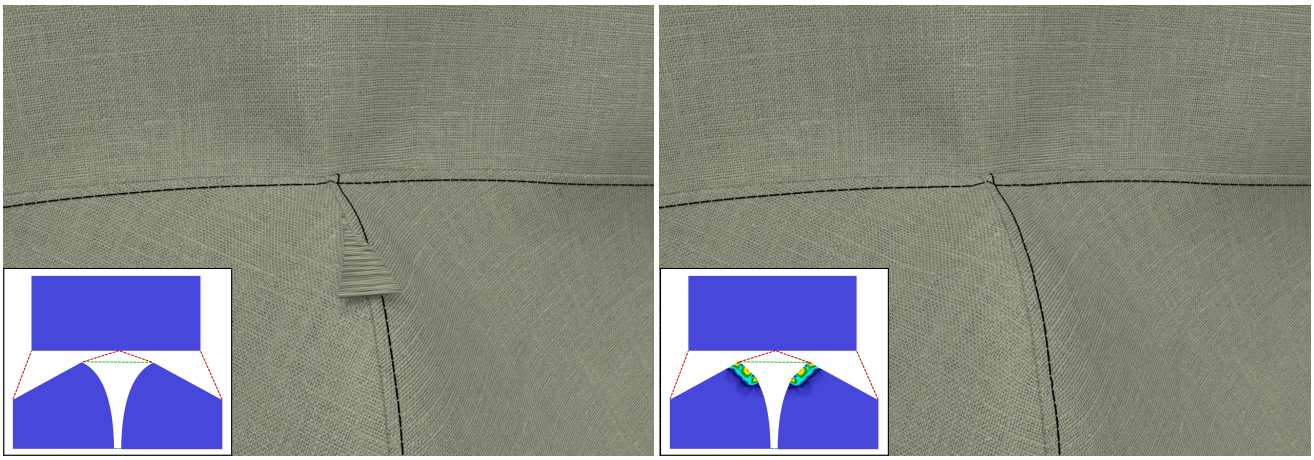


Fig. 11. The True Seams algorithm is applied to an assembly with opposing curved pieces. On the left, artifacts appear if the original pattern is used. On the right, the optimized pattern allows for a continuous *seamline space* and removes the artifacts. The inset images show the deformation of the pattern pieces during the optimization, ranging from 0mm (blue) to 2mm (red).

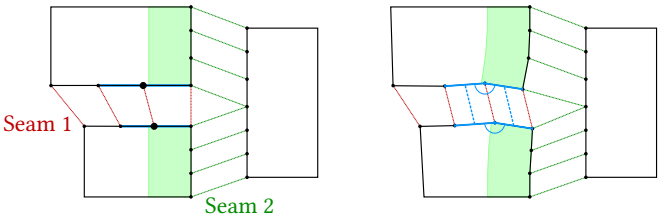


Fig. 12. Left: the asymmetric seamlines of Seam 1 leads to discontinuities in the *seamline space* of the next seam (Seam 2), e.g., the area around the thicker node of the bottom piece will get captured by Seam 2 and receive folding deformation, while the area around the corresponding thicker node in the top piece will not get captured, thus remaining undeformed. Highlighted in blue are the segments of Seam 1 that are to be symmetrical for Seam 2 to span a well defined *seamline space*. Right: Our *pattern space* optimization forces all required seamline sections to have matching segment lengths and angles (shown in blue) while minimizing the deformation of the pieces.

both with and without the pattern optimization step. We note that this modified sewing pattern is used exclusively by the True Seams algorithm: the original sewing pattern remains untouched, and so does the rest shape of the sewing pattern pieces later used for simulation.

7 LIFTING TO *WORLD SPACE*

Once all seams have been treated, each piece has a final list of supported nodes with positions $\boldsymbol{p}_{2d}$ in *pattern space*. In addition, every node that went through a seaming operation now has an h coordinate specifying the height of the node along the normal direction. In *pattern space* this normal direction is always the out-of-plane direction z. In *world space*, however, the pieces (without seam allowance) are assembled together in a garment mesh and deformed through simulation until they reach their final draping position. Therefore, the normal direction in *world space* is a vector field orthogonal to the surface of the garment.

In order to construct the seams in *world space*, we need to find the *world space* position of each fabric node. For each node, we find the triangle containing its $\boldsymbol{p}_{2d}$ position in *pattern space*, and compute the barycentric coordinates of the node with respect to the triangle. We then compute the *world space* position of the node on the surface of the garment $\boldsymbol{p}_{surf}$ using the barycentric coordinates and the position of the triangle in *world space*. To compute the final *world space* position $\boldsymbol{p}_{3d}$ of the node, we offset the node from $\boldsymbol{p}_{surf}$ along the normal vector field $\mathcal{N}$ at $\boldsymbol{p}_{surf}$ by the amount of h, the height coordinate of the node:

$$\boldsymbol{p}_{3d} = \boldsymbol{p}_{surf} + h\mathcal{N}(\boldsymbol{p}_{surf}) \tag{8}$$

If the node doesn't have a height coordinate, it wasn't touched by the seaming operations and there is no offset to apply.

7.1 Normal field optimization

Properly computing $\mathcal{N}$ is key to avoid geometric intersections when offsetting nodes along the normal field. A naïve approach would just compute the normals at each vertex of the surface and use barycentric interpolation for the rest of the field, without any guarantees that an offsetted surface will not self-intersect.

To ensure a smooth and suitable field along the surface, we therefore run a constrained optimization algorithm on surface normals with constraints to prevent intersections within the offset volume. This problem is similar to the seam normal optimization described in section §4.2, but in a 3D setting. Let $\mathcal{T}$ be the set of all triangles of the garment. For each triangle $t \in \mathcal{T}$, we compute θ_{max} as the maximum allowed angle between its vertex normals (see Eq. (11) in Appendix A.2), and define $f_{t,j}(x) = acos(\mathcal{N}_v \cdot \mathcal{N}_w) - \theta_{max}$ for the normals of every pair j of vertices (v, w) in the triangle. Then we solve

$$\begin{aligned} &\min_{N} \sum_i \left\| \mathcal{N}_i - \bar{\mathcal{N}}_i \right\|^2 \\ &\text{s.t.} \quad f_{t,j}(\mathcal{N}) < 0, \quad t \in \mathcal{T}, j \in \{0, 1, 2\} \end{aligned} \tag{9}$$

Fig. 13. Comparison of six assemblies with three pieces and two seams forming a T shape. The two seams have different variations of the General Seaming (SSa 1.01) seam: seam allowance *direction* (Inside, Outside), and seam allowance *towards* (SideA, SideB, Open). The fabric is a two-sided blue denim. Even with such limited variations, the six constructions are strikingly different.

Fig. 14. Photographs of four different three-piece assemblies with combinations of General Seaming (SSa 1.01) seams, matching some of the assemblies of Figure 13 and highlighting the realism of our approach.

avoiding intersections in the normal field while deviating as little as possible from the reference normals $\bar{N}$ of the surface vertices, defined by averaging the normals of incident triangles.

Many of the constraints will be redundant, as two triangles sharing an edge will likely impose different angle constraints, so we simply keep the most restrictive ones. Seamed 3D pieces are treated as a continuous surface, merging the corresponding boundary vertices and their normals.

7.2 Ironing rest angle

Most of the seams are ironed right after being sewn to make sure the folded fabric stays as flat as possible, avoiding visual and tactile discomfort. As such, ironing strongly influences the final appearance of the seam and can prevent it from unfolding.

We reproduce the plastic deformation due to ironing by using the folded angle of the seam as rest angle during simulation. Since our approach computes the geometry of seams before lifting the resulting geometry to *world space*, it inherently dissociates seam deformation from draping deformation. All the deformation that appears in *seamline space* is exclusively due to seaming. Working directly in 3D would have made ironing much more difficult.

8 STITCHING

Stitching is performed for each seam in *seamline space*, after folding the fabric. Different stitch types have different number of threads that interlock with the fabric and with themselves in a wide variety of ways. We discretize each stitch thread as a piecewise linear curve, with consecutive segments joining at stitching nodes, akin to a

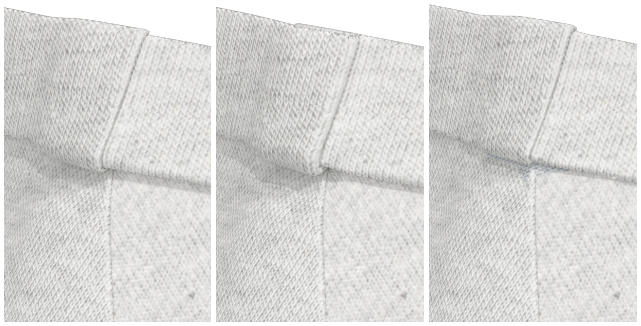


Fig. 15. The neck of a white jersey shirt with different construction orders for the same 3 sewing operations on the collar piece, resulting in noticeable differences in bulkiness and edge finishing. Left: sewing to itself, then piece folding, then sewing to the body. Middle: piece folding, then sewing to itself, then sewing to the body. Right: sewing to itself, then sewing to the body, then piece folding.

mass-spring system. A stitching node is generated when the thread intersects the fabric due to a needle puncture, or when two or more threads interlock. The smoothness and curviness of the threads are recovered during rendering, when the resulting set of segments are converted into splines to exhibit a more natural curvature.

The discretized stitching threads are parameterized using (α, μ) coordinates in *seamline space* along the seam. Since fabric is now folded along the seam, threads usually traverse many layers of fabric each time the needle punctures the fabric. To find the exact puncturing position on each layer, we use raycasting on the geometry around the puncturing coordinates (α, μ) to retrieve all the height coordinates h where the needle intersects a layer of fabric. This process is trivial since we perform it in *seamline space*, with the folded geometry readily available on both sides of the seam as a local continuum and without requiring to track each individual layer in the seam. We place a stitch node at each layer intersection and can move on to the next needle puncture.

Stitching nodes are generated as new fabric nodes, with their own *seamline space* coordinates, and become supported by pattern pieces when updating the support of captured nodes. They can then be moved around in *pattern space* if they become involved in other seams later on, just like any other fabric node. As a consequence, stitching threads can also undergo folding and deformation due to further seaming operations, as shown in Figure 1 where the stitches around the neck area get folded along with the pieces.

9 RESULTS

In this section, we show examples of the True Seams algorithm for two different use-cases: a post-process to an already simulated garment to dramatically improve its appearance, or a pre-process for the resulting geometry to influence the mechanical behavior of a subsequent simulation. The algorithm is identical in both cases. In the post-process case, we used a triangle mesh discretization and first simulated the garments using StVK [Volino et al. 2009] and Discrete Shells [Grinspun et al. 2003] energy models *before* applying True Seams. For the pre-process case, we used a yarn-level discretization and simulated the seam-enhanced geometry *after*

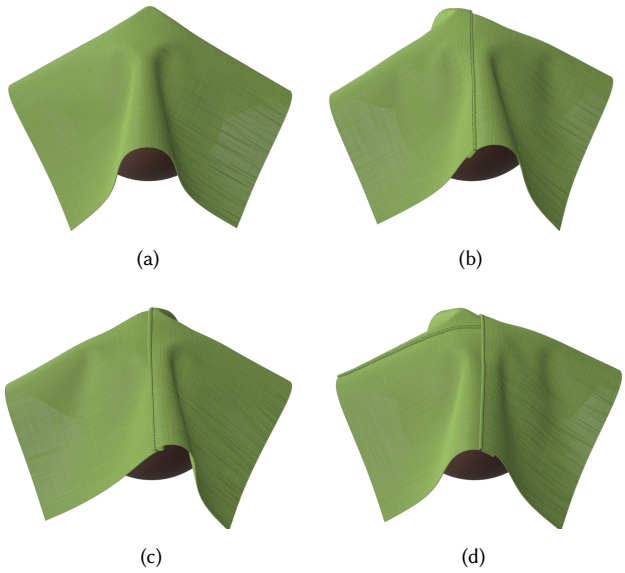


Fig. 16. Comparison of yarn-level simulations of four 20x20cm cloth arrangements made of the same woven linen fabric, draped on top of a sphere after applying the True Seams algorithm. We compare (a) a single piece, (b) two pieces sewn with a 2.02.01 seam, (c) two pieces sewn with a 2.04.01 seam and (d) two pieces sewn with a 2.02.01 seam and then sewn to a third piece with a 2.02.01 seam. The differences in drape emerge naturally, showing the mechanical influence of the different seam arrangements.

applying True Seams using the persistent contact model of Cirio et al. [2014; 2017] and a mass-spring system for the stitches. The results presented in this section were generated using commodity hardware (4.2GHz Quad-core Intel Core i7-7700K CPU with 32GB of memory and an Nvidia GTX 1080 GPU), with a breakdown of computation times shown in Table 1. We refer the reader to our accompanying video for animated results. The seam types used in these examples, including their specific variations, are described in the supplementary document accompanying this paper.

9.1 Seam comparisons

By modeling seams following their true geometric construction, and with dozens of seam and stitch types according to the technical literature [ISO 4915:1991 1991; ISO 4916:1991 1991], the combinations are almost endless. Even variations of the same seam type on

Table 1. Computation time breakdown (in seconds) for examples of full garments, where True Seams is applied as a post-process on top of an already simulated triangle mesh.

	Input Vertices	Output Vertices	Optimizations	Fabric Folding	Total
Dress	12307	1971168	4	14.2	19.5
Sweatshirt	10154	966003	1.2	8.3	10.9
Tank Top	15695	461975	0.5	4	4.9
Pants	78223	1110255	0.8	6.2	7.8

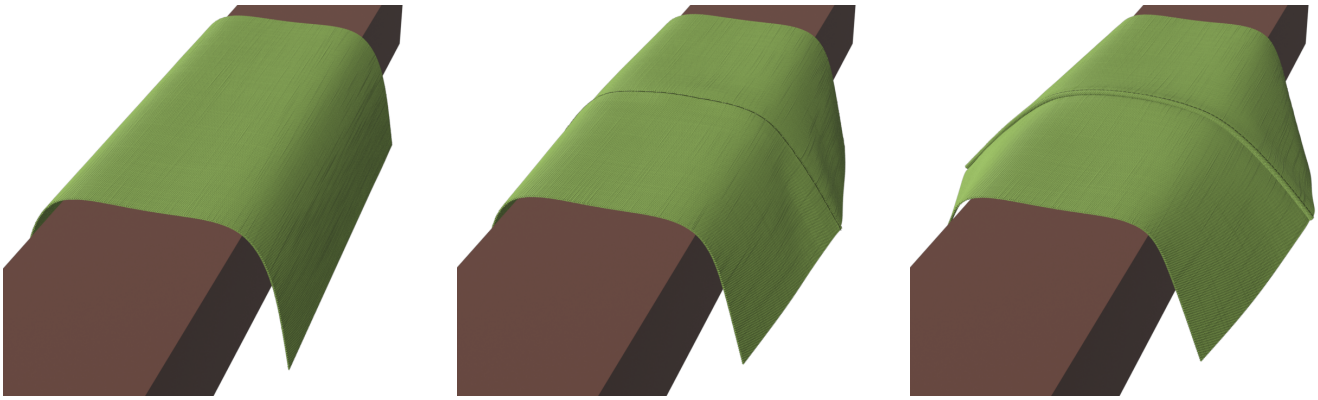


Fig. 17. Comparison of yarn-level drape simulations of three 20x20cm cloth arrangements made of the same woven linen fabric on top of a box. We compare a single piece of cloth (left), two pieces sewn with a 2.02.01 seam (center) and two pieces sewn with a 2.04.01 seam (right). We observe a bending stiffening effect depending on the seam type, naturally emerging from the stacking of fabric layers along the seams.

the simplest assembly of pieces can produce very different visual results.

We illustrate this in Figure 13, where we compare different constructions of the same three pieces forming a T-shaped assembly, with two pieces sewn through one of their edges and then sewn to the third piece. The triangle mesh assembly is already draped and True Seams is applied as a post-process. Both seams use General Seaming (SSa 1.01) as seam type, and Single Thread Chainstitch (101) and Zig Zag Lockstitch (304) respectively as stitch types. The only change is in the variations of the General Seaming: the seam allowance "out of plane" direction once folded (towards the inside or the outside of the garment), and its "in-plane" direction (towards either side of the seam, or folded open). We used a two-sided denim fabric to easily identify the front and the back of the fabric. Even with such limited variations, the six constructions are strikingly different, ranging from an almost seamless appearance to prominent back-face fabric stripes running along the seams. Some folds can be bulky due to the accumulation of layers, while others are not, with stitch lines clearly visible in some cases but hidden otherwise.

These results can be compared to their real counterparts in Figure 14, where four of the above combinations were sewn together using real denim fabric and photographed on top of a spherical support. All salient features are captured by our approach: the increased thickness due to folded layers, the back side of the fabric popping through and the visible stitching threads.

Seam order is another important feature of our algorithm. Different construction orders will produce different results, as in a real garment. In Figure 15, we show different construction orders for the same three sewing operations on the collar of a jersey shirt: piece folding, sewing to itself, and sewing to the body pieces. A piece folding is the operation of folding the entire piece in half along a symmetry line, which we treat as a type of seam. The results are different and depend exclusively on the construction order, going from bulky and open at the top of the collar, to thin and smoothly wrapping around the edge. A designer can use our algorithm to play with different options, as well as catch and correct design defects early on in the garment creation process.

9.2 Draping comparisons

We compare several yarn-level draping simulations to show the effect of different seam types on the drape. Here True Seams is applied as a pre-process to the yarn-level simulation. In Figure 16 we compare four 20x20cm woven linen fabric patches with different seam assemblies: a single piece, two pieces sewn with a Topstitching Seam (LSb 2.02.01), two pieces sewn with a Single Needle Felled Seam (LSc1 2.04.01), and four pieces sewn with Topstitching Seams. We ran the simulations until the patches came to a rest. From the resulting drapes we can observe that the presence of seams as well as the seam type affect the mechanical properties of the patch. Differences in the appearance of the seam foldings themselves are also apparent.

A similar comparison is shown in Figure 17, with the simulation of three 20x20cm fabric patches made of either a single piece, two pieces sewn with a Topstitching Seam (LSb 2.02.01), and two pieces sewn with a Single Needle Felled Seam (LSc1 2.04.01). We observe a bending stiffening effect depending on the seam type, naturally emerging from the geometric assembly produced by the True Seams algorithm. The single piece is the reference, with an homogeneous bending along its length. The Topstitching seam folds fabric in one of the pieces, stacking 3 layers along the seam line, stiffening the bending behavior in that section of the arrangement. The Single Needle Felled seam folds fabric in both pieces, stacking 4 layers along the seamline, leading to further bending stiffening.

Regarding performance, the single piece patch (Figure 16.a and Figure 17 left) consist of ∼200k simulation nodes with an average cost of 2.2s per 1ms simulation step, while the other patches, which have seams, consist of ∼225k simulation nodes with an average cost of 2.6s per step. The cost increase of ∼20% is due both to the simulation nodes added by the seam allowances and to the more challenging collision scenarios.

9.3 Full garments

We applied True Seams as a post-process to a set of triangle mesh garments, and compared the results to the original garment without True Seams. In both cases, input patterns and their corresponding

Fig. 18. A knit fleece sweatshirt with V-inserts and a folded collar piece. The shoulder and collar seams add depth and relief to the garment, compared to the original garment in the inset image.

3D meshes were subdivided using Loop subdivision [Stam 1998] in the vicinity of seams to have 1mm edge lengths (see Figure 2 for a discretization example). The size of the vicinity is given by the seam type of each seam (in particular μ_{max} as described in §4.3). Having a fine grained discretization around seams is necessary to properly resolve folds, otherwise our algorithm cannot produce intersection-free geometry. Adding seams and stitches dramatically increases the realism of the garments.

Denim dress (Figure 1). The upper body of the dress is made of 3 pieces for the front and 4 pieces for the back in a thick two-sided denim fabric, and is sewn together using Felled Seams (LSc 2.04.06). In the middle of each back shoulder piece there is a dart sewn using General Seaming (SSa 1.01). The neck is hemmed towards the inside using a Clean Finish Hem (EFb 6.03.01). All seams and hems use Lockstitch (301) in beige and black. Notice the bumps created by overlapping layers of fabric where the hem meets vertical seams, and where the darts meet the shoulder seams. In the accompanying video, we show how True Seams can be used in a animation as a postprocess for every frame. Since most of the work is done using the 2D sewing pattern of the garment, it can be reused for all frames of the animation, with implicit temporal coherence between frames even under fast motion and significant deformation.

Sweatshirt (Figure 18). The sweatshirt is made of knit fleece body pieces with a front and a back V-insert, and a neck that is first sewn to itself (creating a cylinder shape), then folded in half (a piece fold) and finally sewn to the body. All seams are General Seaming (SSa 1.01), and stitching is done with Lockstitch (301) and Coverstitch (605). Notice how the shoulder seams run towards the neck and fold above it, creating depth and relief, while in the original garment everything is flat. The inside of the garment is also clearly visible, with the Coverstitch sewing the V-insert to the back piece.

Tank top (Figure 19, left). The tank top is made of a front and a back body piece using black single jersey fabric and sewn together using General Seaming (SSa 1.01). Thin white jersey fabric stripes are then sewn to the collar and the side openings of the body using Seam & Cord Seam (LSq 2.02.03) and a combination of 3 Thread Overedge (504) and Lockstitch (301). All edges are hemmed towards the inside using Raw Edge Hems (EFa 6.02.01) and Single Thread Chainstitch (101). There is a clear bulkiness on the shoulders due to multiple pieces meeting each other. Notice the delicate black Overedge and white Chaintstitch stitching threads in the interior of the garment.

Pants (Figure 19, right). The pants are made of two front and two back pieces of Milano knit fabric joined together by thick jersey fabric stripes and a rib waistband, using a combination of General Seaming (SSa 1.01) and Single Needle Felled Seams (LSc1 2.04.01). The waistband creates noticeable relief all around the waist opening.

10 CONCLUSION

We have presented an algorithm to model seams in digital garments by overlapping, folding and stitching together the fabric pieces of sewing patterns, following the true construction of seams in real-life garments. Key to our approach is the splitting of a complex 3D problem into a sequence of simpler 2D problems by working in pattern, seamline and profile spaces before lifting the geometry to 3D space. We showcased different seams and stitch types and their effect on draping and appearance, as well as full garments with

Fig. 19. Left: a black single jersey tank top with white fabric stripes sewn to the collar and the side openings of the body. We encourage the reader to zoom in to better appreciate all the details in the interior of the garment. Right: Milano knit pants with thick jersey fabric stripes and a rib waistband.

complex asymmetric seamlines and overlapping sets of seams, all correctly resolved and intersection-free.

Our approach and implementation are not without limitations, and there are many opportunities for future work. We have limited our seaming support to boundary seams, each seam joining two sides. While these are the most common seam types and cover a wide range of garments, other common seams, such as interior pocket seams involving the interior of a piece, or multi-layer seams where more than two sides are joined together in a single seaming operation, are also required to allow faithful virtual replicas of many other types of garments. We plan to extend our method to support these and other seam types as well.

Our method produces simulation-ready geometry, and we have shown examples of macroscopic mechanical effects that can emerge naturally. However, this is true only for sufficiently fine geometry: layers can intersect each other at folds solely due to a locally inadequate discretization. With a 1mm edge resolution around seams, we have not observed any self-intersection even in complex cases such as Figure 2 with many layers and multiple folds. However, if self-intersections were detected, one could use a backtracking approach where the problematic triangles are further subdivided until all folds are resolved properly. In addition, given the large number of vertices that are required to properly resolve fabric folds and produce smooth geometry, efficiently simulating such a high resolution geometry is a challenging task. Designing a new simulation method was outside the scope of this paper. Nevertheless, we believe that our method, once coupled to an efficient seam simulation technique, will open the possibility to explore systematic testing of seaming properties in the mechanical behavior of garments. Ultimately, True Seams could help in the study not only of the aesthetics of garments, but also of their dynamic behavior such as wear comfort and performance in motion.

ACKNOWLEDGMENTS

The authors would like to thank Harrison Johnson and Miguel A. Otaduy for many insightful discussions, Sofía Domínguez and Loreto Pérez for their help with real fabric assemblies, Javier Fabre and Víctor Arellano for seam and stitch rendering, and the SEDDI teams in general for their help and support. This work was funded in part by the Spanish Ministry of Economy, Industry and Competitivity with Torres Quevedo grants FashionScale (PTQ-17-09154) and FashionSkin (PTQ-17-09156), and by the Spanish Ministry of Science and Innovation with Retos Colaboración grant FashionAvatar (RTC2019-007480-6).

REFERENCES

Assembil. 2013. *How patterns work: The fundamental principles of pattern making and sewing in fashion design.* Createspace Independent Publishing Platform.

Aric Bartle, Alla Sheffer, Vladimir G. Kim, Danny M. Kaufman, Nicholas Vining, and Floraine Berthouzoz. 2016. Physics-driven pattern adjustment for direct 3D garmentediting. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 35, 4 (2016).

Floraine Berthouzoz, Akash Garg, Danny M. Kaufman, Eitan Grinspun, and Maneesh Agrawala. 2013. Parsing sewing patterns into 3D garments. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 32, 4 (2013).

Remi Brouet, Alla Sheffer, Laurence Boissieux, and Marie-Paule Cani. 2012. Design preserving garment transfer. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 31, 4 (2012).

J. Chung. 1999. *The effect of assembly methods of a garment on fabric drape.* Ph.D. Dissertation. Institute of Textiles and Clothing, The Hong Kong Polytechnic University.

Gabriel Cirio, Jorge Lopez-Moreno, David Miraut, and Miguel A. Otaduy. 2014. Yarn-level simulation of woven cloth. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia)* 33, 6 (2014).

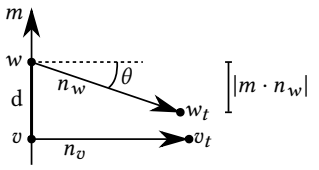
Gabriel Cirio, Jorge Lopez-Moreno, and Miguel A. Otaduy. 2017. Yarn-Level Cloth Simulation with Sliding Persistent Contacts. *IEEE Transactions on Visualization and*

Computer Graphics 23, 2 (2017), 1152–1162.
Jonathan Cohen, Amitabh Varshney, Dinesh Manocha, Greg Turk, Hans Weber, Pankaj Agarwal, Frederick Brooks, and William Wright. 1996. Simplification envelopes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 119–128.
Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. 2003. Discrete Shells. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 62–67.
JL Hu and Siuping Chung. 2000. Bending Behavior of Woven Fabrics with Vertical Seams. *Textile Research Journal* 70 (02 2000), 148–153.
J. Hu, S. Chung, and M. Lo. 1997. Effect of seams on fabric drape. *International Journal of Clothing Science and Technology* 9 (3 1997), 220–227.
Liang Hu, Jinlianand Ma, George Baciu, Wingo Sai-Keung Wong, and Weiyuan Zhang. 2006. Modelling Multi-layer Seam Puckering. *Textile Research Journal* 76, 9 (2006), 665–673.
Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. 2008. Automatically adding seam allowance to cloth pattern. In *ACM SIGGRAPH 2008 Posters*.
S. Inui, H. Okabe, and T. Yamaraka. 2001. Simulation of seam pucker on two strips of fabric sewn together. *International Journal of Clothing Science and Technology* 13 (02 2001), 53–64.
S. Inui and T. Yamaraka. 1998. Seam pucker simulation. *International Journal of Clothing Science and Technology* 13, 2 (1998), 128–142.
ISO 4915:1991. 1991. *Textiles – Stitch types – Classification and terminology*. Standard. International Organization for Standardization, Geneva, CH.
ISO 4916:1991. 1991. *Textiles – Seam types – Classification and terminology*. Standard. International Organization for Standardization, Geneva, CH.
Yamini Jhanji. 2018. Computer-aided design – garment designing and patternmaking. In *Automation in Garment Manufacturing*, Rajkishore Nayak and Rajiv Padhye (Eds.). Woodhead Publishing, 253 – 290.
Michael Keckeisen, Matthias Feurer, and Markus Wacker. 2004. Tailor Tools for Interactive Design of Clothing in Virtual Environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. ACM, 182–185.
Minchen Li, Alla Sheffer, Eitan Grinspun, and Nicholas Vining. 2018. Foldsketch: Enriching Garments with Physically Reproducible Folds. *ACM Trans. Graph.* 37, 4, Article 133 (jul 2018), 13 pages.
Shufang Lu, P.Y. Mok, and Xiaogang Jin. 2017. A new design concept: 3D to 2D textile pattern design for garments. *Computer-Aided Design* 89 (2017), 35 – 49.
Liang Ma, Jinlian Hu, and George Baciu. 2006. Generating Seams and Wrinkles for Virtual Clothing. In *Proceedings of the 2006 ACM International Conference on Virtual Reality Continuum and Its Applications (VRCIA)*. 205–211.
Takashi Maekawa. 1999. An overview of offset curves and surfaces. *Computer-Aided Design* 31, 3 (1999), 165–173.
Yuwei Meng, Charlie C.L. Wang, and Xiaogang Jin. 2012. Flexible shape control for automatic resizing of apparel products. *Computer-Aided Design* 44, 1 (2012), 68 – 76.
Juan Montes, Bernhard Thomaszewski, Sudhir Mudur, and Tiberiu Popa. 2020. Computational Design of Skintight Clothing. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 39, 4 (2020).
Fatemeh Mousazadegan, Siamak Saharkhiz, and Masoud Latifi. 2012. Prediction of tension seam pucker formation by finite-element model. *International Journal of Clothing Science and Technology* 24 (06 2012), 129–140.
Simon Pabst, Sybille Krzywinski, Andrea Schenk, and Bernhard Thomaszewski. 2008. Seams and Bending in Cloth Simulation. In *Workshop in Virtual Reality Interactions and Physical Simulation (VRIPHYS)*. 31–38.
Jianbo Peng, Daniel Kristjansson, and Denis Zorin. 2004. Interactive Modeling of Topologically Complex Geometric Detail. In *ACM SIGGRAPH 2004 Papers* (Los Angeles, California) *(SIGGRAPH '04)*. Association for Computing Machinery, New York, NY, USA, 635–643.
Binh Pham. 1992. Offset curves and surfaces: a brief survey. *Computer-Aided Design* 24, 4 (1992), 223–229.
Serban D Porumbescu, Brian Budge, Louis Feng, and Kenneth I Joy. 2005. Shell maps. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 626–633.
Xavier Provot. 1995. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour. In *Proceedings of Graphics Interface (GI)*. 147–154.
Scott Roland, Mathew D. Janda, and Charles Lowry. 2015. Implementation of Modeling and Simulation of Textile Seam and Joints for Parachute Design Applications. In *Aerodynamic Decelerator Systems Technology Conferences*.
Jos Stam. 1998. Evaluation of loop subdivision surfaces. In *Proceedings of ACM SIGGRAPH 98*.
Kara Sukran. 2020. Comparison of sewn fabric bonding rigidities obtained by heart loop method: effects of different stitch types and seam directions. *Industria Textila* 71, 2 (2020), 105–111.
Nobuyuki Umetani, Danny M. Kaufman, Takeo Igarashi, and Eitan Grinspun. 2011. Sensitive Couture for Interactive Garment Modeling and Editing. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 30, 4 (2011).
Pascal Volino, Nadia Magnenat-Thalmann, and François Faure. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Transactions on Graphics* 28, 4 (2009).
Huamin Wang. 2018. Rule-free sewing pattern adjustment with precision and efficiency. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 37, 4 (2018).
Katja Wolff, Philipp Herholz, and Olga Sorkine-Hornung. 2019. Reflection Symmetry in Textured Sewing Patterns. In *Vision, Modeling and Visualization*. The Eurographics Association.
Katja Wolff and Olga Sorkine-Hornung. 2019. Wallpaper Pattern Alignment along Garment Seams. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 38, 4 (2019).
Yunchu Yang. 2014. Investigating Seamed Woven Fabric Drape Using Experimental and Virtual Approaches. *Fibers and Polymers* 15 (10 2014), 2217–2224.

A COMPUTING θ_{max}

A.1 For seamline normal optimization in Eq. (3)

Let v and w be two vertices separated by a distance d in a seamline and let m be the unit vector aligned with the seamline. Let the vector n_v be perpendicular to m and let the vector n_w be rotated by θ radians with respect to n_v. The projection of n_w onto m satisfies $|m \cdot n_w| < |\theta|\,\|n_w\| \;\; \forall \theta$. We can then define a maximum rotation rate per unit displacement as $\frac{1}{\|n_w\|}$ that guarantees that when $\theta < \frac{d}{\|n_w\|}$ then $d > |m \cdot n_w|$, i.e., the segments $\overline{vv_t}$ and $\overline{ww_t}$ do not intersect, with $v_t = v + n_v$ and $w_t = w + n_w$.



In practice we also want to prevent excessive cramming of material in any area, so we define the maximum allowed angle between the normals of two nodes v, w separated by d that belong to a seamline with maximum capture distance μ_{max} as

$$\theta_{max_{vw}} = 0.5\frac{d}{\mu_{max}} \tag{10}$$

A.2 For 3D normal optimization in Eq. (9)

For each triangle defining the *world space* configuration of the garment, we want to limit the rate of change of the normal field inside it as a function of the maximum offset that will be applied to the nodes contained by the triangle. We define a conservative limit for each triangle in *world space* by taking d as the minimum distance between each of the triangle vertices and their opposing edge, and h_{max} as the maximum absolute height coordinate of the nodes supported by it, and compute

$$\theta_{max} = 0.5\frac{d}{h_{max}} \tag{11}$$