Weavecraft: An Interactive Design and Simulation Tool for 3D Weaving

RUNDONG WU, Cornell University, USA
JOY XIAOJI ZHANG, Cornell University, USA
JONATHAN LEAF, Stanford University, USA
XINRU HUA, Stanford University, USA
ANTE QU, Stanford University, USA
CLAIRE HARVEY, T.E.A.M. Inc., USA
EMILY HOLTZMAN, Rhode Island School of Design, USA
JOY KO, Rhode Island School of Design, USA
BROOKS HAGAN, Rhode Island School of Design, USA
DOUG JAMES, Stanford University, USA
FRANÇOIS GUIMBRETIÈRE, Cornell University, USA
STEVE MARSCHNER, Cornell University, USA

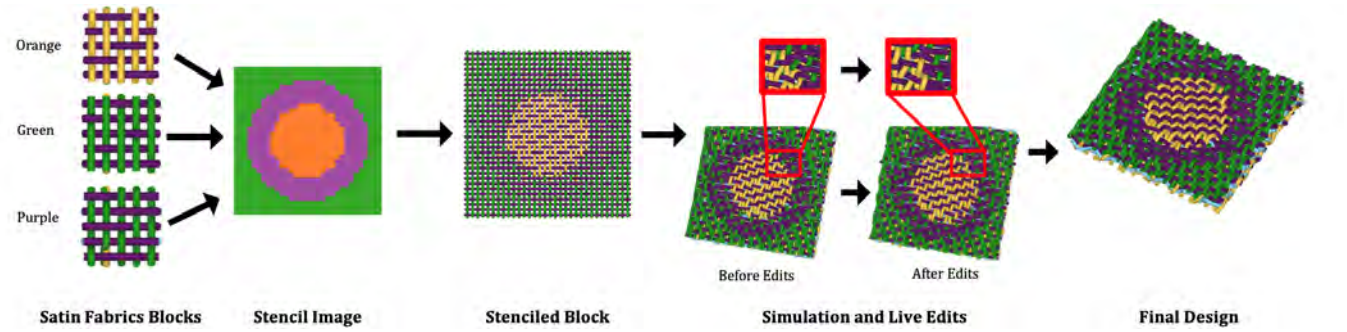


Fig. 1. With Weavecraft, textile designers interactively simulate and design woven fabrics in 3D. In this simple two-layer example shown in the supplementary video, a user plans to make a 3 color fabric shown in the "Stencil Image". The user combines the three satin patterns into a large block using the stenciling operation. The user then simulates and makes a few edits along boundaries between colored regions to fix the unexpected yarn interactions. The wall-clock time for this interactive design session was five minutes.

3D weaving is an emerging technology for manufacturing multilayer woven textiles. In this work, we present Weavecraft: an interactive, simulation-based design tool for 3D weaving. Unlike existing textile software that uses 2D representations for design patterns, we propose a novel weave block representation that helps the user to understand 3D woven structures and to create complex multi-layered patterns. With Weavecraft, users can create blocks either from scratch or by loading traditional weaves, compose the blocks into large structures, and edit the pattern at various scales. Furthermore, users can verify the design with a physically based simulator, which predicts and visualizes the geometric structure of the woven material and reveals potential defects at an interactive rate. We demonstrate a range of results created with our tool, from simple two-layer cloth and well known 3D structures to a more sophisticated design of a 3D woven shoe, and we evaluate the effectiveness of our system via a formative user study.

CCS Concepts: • **Computing methodologies → Shape modeling**.

Additional Key Words and Phrases: 3D Weaving, Yarn-level Cloth Simulation, Computer Aided Design

ACM Reference Format:
Rundong Wu, Joy Xiaoji Zhang, Jonathan Leaf, Xinru Hua, Ante Qu, Claire Harvey, Emily Holtzman, Joy Ko, Brooks Hagan, Doug James, François Guimbretière, and Steve Marschner. 2020. Weavecraft: An Interactive Design and Simulation Tool for 3D Weaving. *ACM Trans. Graph.* 39, 6, Article 210 (December 2020), 16 pages. https://doi.org/10.1145/3414685.3417865

Authors' addresses: Rundong Wu, Cornell University, Ithaca, USA; Joy Xiaoji Zhang, Cornell University, Ithaca, USA; Jonathan Leaf, Stanford University, Stanford, USA; Xinru Hua, Stanford University, Stanford, USA; Ante Qu, Stanford University, Stanford, USA; Claire Harvey, T.E.A.M. Inc. Woonsocket, USA; Emily Holtzman, Rhode Island School of Design, Providence, USA; Joy Ko, Rhode Island School of Design, Providence, USA; Brooks Hagan, Rhode Island School of Design, Providence, USA; Doug James, Stanford University, Stanford, USA; François Guimbretière, Cornell University, Ithaca, USA; Steve Marschner, Cornell University, Ithaca, USA.

0730-0301/2020/12-ART210 $15.00
https://doi.org/10.1145/3414685.3417865

1 INTRODUCTION

Weaving is a technology that fabricates cloth by interlacing two groups of yarns, weft and warp, on a loom (shown in Figure 2). Unlike conventional woven materials, in which yarns usually lie one to three layers thick, 3D woven fabrics can take significant 3D form, with dozens of layers stacked into a thick, solid material. The multi-layered fabrics can be created with numerous yarn structures, resulting in a wide variety of shapes and appearances. 3D woven fabrics are extensively used in demanding technical applications such as composites for automotive and aerospace applications, because they provide control over the placement of reinforcing fibers to achieve mechanical properties such as high stiffness and resistance against fatigue, fracture, and delamination. There is also an emerging trend in using 3D weaving as a general fabrication technique, which has the potential to create complicated functional objects [Harvey et al. 2019] and arbitrary shapes [Wu et al. 2020]. Despite the great potential of 3D weaving, its applications are limited by the difficulty in designing weaving patterns with existing textile software.

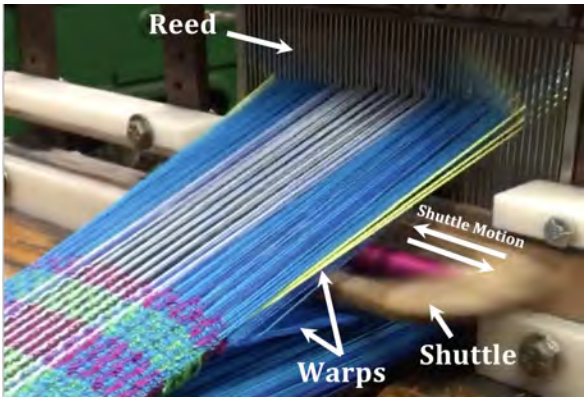


Fig. 2. **3D weaving loom in action:** The shuttle carries the weft yarn through the "shed" formed by the warp yarns, which pass through the openings in the comb-like reed that in turn battens down weft yarns.

3D fabrics can be manufactured using traditional Jacquard looms, where the raising and lowering of warp yarns are controlled by a *card image*, a binary pattern indicating the relative positioning between the weft and warp yarns, with one column for each warp, one row for each weft, and each pixel specifying if one warp yarn passes above (black) or below (white) one weft yarn. Figure 2 illustrates a typical weaving process, where a single weft yarn is carried by a shuttle that moves back and forth, while the warp yarns are arranged into an array by the reed, and raised or lowered as the shuttle moves so that they interlace with the weft yarn before it is pressed into place by the reed.

While traditional software such as PointCarre (discussed in Section 2) supports the design of repeating 3D structures and large-scale 2D patterns, designing large-scale and spatially varying 3D structures is extremely difficult. Existing tools represent weave patterns as 2D card images, which are unintuitive to read and lead to tedious and error-prone design processes even for experienced designers.

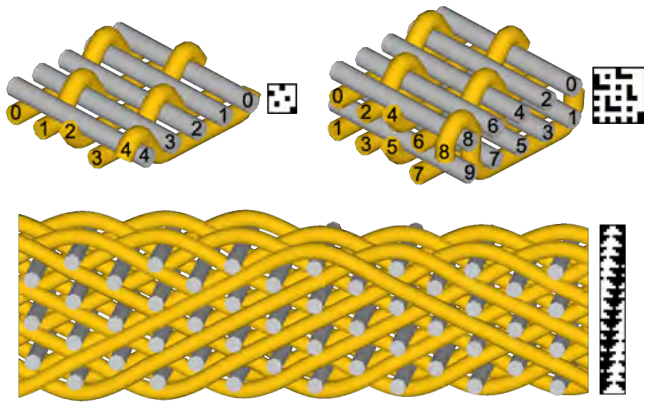


Fig. 3. Simple weave structures and their card images. In the two examples in the top row, each weft yarn (grey) is marked with the corresponding row number in the card image, and each warp yarn (orange) is marked with the corresponding column number.

With card images, users are expected to not only establish the spatial relationship of the different layers in their minds, but also carefully maintain the compatibility of the structures being juxtaposed to achieve the desired shape and appearance. This gets increasingly difficult as the patterns become more complex (Figures 3). Furthermore, existing software fails to accurately predict the geometric structure of a design pattern upon being woven, which can deviate from the expected result due to inter-yarn forces. The only way to verify the design is to physically manufacture the fabric, which is a time-consuming and expensive iterative process. The constraints of existing software call for tools catering to the design of large-scale, multi-layered fabrics.

In this paper, we present Weavecraft: an interactive, simulation-assisted tool for 3D weaving design. Weavecraft introduces a novel *weave block representation* as shown in Figure 3, which supports easy comprehension and efficient creation of large-scale, spatially varying 3D woven fabrics. A typical design process in Weavecraft is as follows: In a 3D interface, a user creates simple 3D blocks and assembles them into complex patterns using stenciling. The user can then adjust the design at the yarn level while quickly observing the effect of the changes on the final design by running physically based simulation. At the end of the design process, the software automatically generates a card image to be used as an input to the loom. Our system is closer to a modern CAD system and resolves the barriers to 3D weave design in existing software via three major improvements: (1) direct control and editing of weave structures at all scale levels, from local yarn flips to large-scale arrangements of block structures; (2) tight integration of simulation into the design loop to reveal the flaws in the patterns, verify the tweaks, and allow faster design iterations; and (3) 3D visualization of a yarn structure which naturally conforms with both the physical appearance and the mental model of a multi-layered weave pattern.

We demonstrate some examples of 3D weaving designs and 3D woven materials created with our system in Section 6. We CT

scanned some of the samples and compare the simulation results against the CT data. The results show that the simulation can predict crucial features of the 3D structure, such as deflected yarns in the deformed structure. The predictive power of the simulation helps reveal the defects of the design without actually fabricating the sample, and significantly accelerates the design process by reducing the number of iterations. In addition, we performed a formative study among three users to verify that our approach is understandable by textile designers with little prior experience in 3D weaving, and that the simulator does uncover the potential flaws in the patterns to assist the design process.

2 RELATED WORK

2.1 3D Weaving

3D weaving technology has existed for decades [Chen et al. 2011], and is most commonly used to create composite materials for high-performance industries. These composites exhibit many favorable mechanical properties (such as resistance to delamination) as they are created by laying many layers of yarns and binding the layers together using z-tow yarns

The vast majority of 3D weaving for composites consists of making big blankets with simple repeating weave patterns, bending the fabric to fit the shape of a mold, and then adding the resin. Some examples of this approach are tubular fabrics [Geerinck et al. 2016], spacer fabrics [Yip and Ng 2008], and canisters [Chen et al. 2011]. For parts that are difficult to make with repeating structures, preforms of various 3D shapes [Mouritz et al. 1999; Umair et al. 2015] are created to fit a particular part. Such fabrics are usually composed with hand-crafted patterns, which either only have simple shapes or need post processing to be molded into a desired shape. This is a manual, rare, and expensive process. While 3D weaving is capable of fabricating objects of very general shapes, the applications that require intricate part-specific designs are bottlenecked by the lack of design tools.

2.2 Woven Textile Design Software

Various software exists to support the creation of card images for woven materials.

ScotWeave [2019] provides an interface for creating repeatable weave structures in a slice view and can produce the corresponding card image. However, it can only create repeated patterns rather than spatially varying structures. PointCarre [1988] is a mature system for 2D weaving, which provides features to create fabrics with complicated graphical patterns by substituting different card images according to an input image. However, it does not provide 3D visualization of the weave strctures and does not reason about 3D structure compatibility when combining the card images, making 3D weaving design difficult. EAT [2015] demonstrates a 3D visualization interface, but manipulations of the weave structure are still based on a 2D card image interface. MultiMech [2018] provides a menu which allows the user to specify yarn topology with numbers and generate small scale weave structures. TexGen [Sherburn 2007] is an open source package focusing on simulating small scale weave structures using finite element method, which also provide basic structure editing capability.

In contrast with these previous systems, Weavecraft provides designers the ability to control the structure at both local and global scales and integrates physically-based simulation deeply into the design loop. It is designed to help the user understand the relationship between small- and large-scale design structure and the resulting physical cloth behavior, which is crucial for designing customized 3D shapes as opposed to regular fabrics. Interactive simulation and editing also provide rapid feedback about design choices and tweaks, enabling faster iterations.

2.3 Yarn-level Cloth Modeling and Simulation

Yarn-level cloth simulation has been an active topic in both the computer graphics and textile communities. In computer graphics, the seminal work of Kaldor et al. [2008, 2010] used a spline-based model to represent individual yarns of knitted cloths and got realistic deformation from simulation. Volino et al. [2009] applied adapted strain-stress laws to efficiently and accurately simulate the nonlinearity of cloth. Sueda et al. [2011] combined Lagrangian and Eulerian approaches to generate efficient and accurate simulations of strands. Cirio et al. [2014, 2015, 2016] use persistent contacts to simplify collision handling, allowing for faster animations of single-layer woven fabrics, though at the cost of not supporting general contact configurations. Leaf et al. [2018] implemented a GPU simulator for knitted and woven cloth based on Kaldor's model, which significantly accelerates the simulation and makes an interactive design tool possible. Our system uses an implementation of a similar GPU simulator to predict deformation of 3D woven models.

Aside from simulation, yarn-level cloth modeling has also been an active topic of research in graphics. Yuksel et al. [2012] built a tool to design knits on mesh surfaces, and Wu et al. [2018] generates knit topology on surfaces automatically and provides a corresponding 3D interface for knit structures. Zhao et al. [2016] uses a procedural model to generate fibers on yarn curves for realistic rendering.

In the textile community, works on woven cloth modeling and simulation focus on the mechanical properties of composites [Ansar et al. 2011; Dash et al. 2013; Huang et al. 2013; Lin et al. 2008; Miao et al. 2008; Sherburn 2007]. For example, Huang et al. [2013] simulates a repeatable unit of 3D woven composite with periodic boundary conditions. The yarn is iteratively split into more and more virtual fibers after relaxation converges in each level, so cross sectional deformation of yarns can be simulated with multiple fibers per yarn. Chen [2011]; Isart et al. [2015]; Nauman and Cristian [2015] present various modeling approaches for 3D woven textiles, with a focus on reproducing the geometric shapes of yarns in the woven material.

2.4 Textile Design in Graphics and HCI

There have been numerous works in the graphics and HCI communities on creating tools to assist the garment and textile making process. For example, McCann et al. [2016] and Narayanan et al. [2018, 2019] generate knit topology on a surface and knitting machine instructions to fabricate the cloth. Instead of constructing weave structures from the yarn level, many works have designed flat fabrics that are cut and sewn into clothing. Umetani et al. [2011] and Li [2018] have developed interactive interfaces with shaping

features such as darts and folds to fit 3D models, and integrated with various simulators.

Another stream of work has focused on customizing garments for both hand knitting [Igarashi et al. 2008a,b; Wu et al. 2019] and machine knitting [Hofmann et al. 2019; Kaspar et al. 2019], with applications in creating soft actuate objects [Albaugh et al. 2019].

Other works have also explored integrating textiles with 3D printing [Pérez et al. 2017; Rivera et al. 2017] and multiple-material fabrication [Vidimce et al. 2016]. Recently, the design and manufacture of interactive, touch-sensitive textiles has become increasingly prevalent [Friske et al. 2019; Hamdan et al. 2018; Klamka et al. 2020; Poupyrev et al. 2016], which opens up a new direction in the realm of wearable computing.

3 DESIGN OVERVIEW

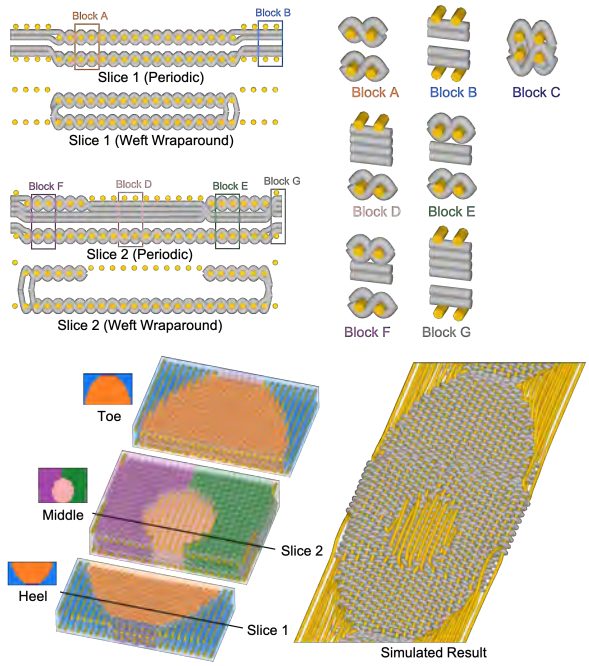


Fig. 4. **Design of a simple shoe.** The shoe has three components: The toe, the middle, and the heel. For each component, several weave blocks (Blocks A to F) are created from 2D slices and merged into a larger block based on a given stencil image. Weavecraft provides two modes for weft yarn boundaries, periodic mode where each weft yarn repeats itself, and wraparound mode where consecutive wefts are connected and wrap around warp yarns catching the connections (details in Section 4.2). The fully assembled shoe can be simulated with periodic boundary conditions in the warp direction, and the result is shown in the bottom right with weft wraparound.

Once the yarn types and loom setup are chosen, designing woven fabrics essentially reduces to arranging the weft and warp yarns to achieve the desired shape, appearance, and mechanical properties for the final fabric. Designing 3D woven objects follows the same process in principle, but the structures are much more complex. Weavecraft makes it easy for users to control the design at varying scales, from yarn arrangement to block construction, as well as supporting the user's understanding of the material's logical and physical structures. The core of supporting 3D weaving design is the introduction of *weave blocks*, which are tilable units of varying 3D weave structures. In our system, the simplest blocks are usually created from slices (see Blocks A to G in Figure 4). These blocks can be composed into larger blocks hierarchically, ultimately leading to a complex final design such as the shoe. During the design process, users can easily compose and modify the structures at varying levels of detail.

To illustrate how Weavecraft supports the design of complicated fabrics such as footwear, we provide a step-by-step walkthrough for designing a simplified shoe (Figure 4). The shoe design consists of three stages: Creating blocks from 2D slices, composing and editing the blocks, and running simulation to verify the design.

Creating blocks. There are three approaches to create a weave block in Weavecraft. The simple ways are loading either a card image or one of the basic preset patterns directly. Another way is to draw a weft or warp slice from scratch in the 2D slice editor and load into the 3D viewer. Blocks A to G in Figure 4 are essentially single-slice blocks resulting from this approach, where the yarn paths are drawn manually with a mouse, and adjustments are made by duplicating, removing, or moving yarns. Each yarn is also renumbered to reflect the weft insertion order on an actual loom. In addition, the weft (in gray) ends can be connected to resemble the turnaround of a single, continuous weft yarn in shuttle weaving, described in Section 4.2.

Editing blocks. Once the small weave blocks are created and loaded in the 3D viewer, the user can apply block operations including attaching one block to another on an arbitrary face (Figure 5(b)), tiling the block in any direction, and merging different blocks with a *stencil*. The design of the shoe is structured as the assembly of 3 different blocks for the toe, the middle part, and the heel. In turn, each block is created using a stencil operation, where blocks corresponding to each color in the stencil image are specified to achieve the required design (Figure 4(a)). The three stenciled blocks are then attached to form a complete shoe. Although not illustrated in the shoe, the user can apply minor tweaks by flipping a pair of weft and warp yarns upon completing the block composition, explained in Section 4.4.

Running the simulation. Once the design is completed, the user can relax the pattern using simulation. Depending on the simulation result, the user can also make yarn edits directly on the simulated model. The simulation can be paused whenever a tweak is desired, and simulation parameters describing the fabric such as the yarn radius and tension can be easily tuned. The simulated result in Figure 4 demonstrates how the shoe would look like in an actual loom production.

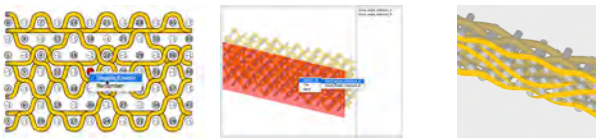
Weavecraft has two modes for weft yarn boundaries, periodic and wraparound, as shown in the left of Figure 4. In the periodic mode, each weft yarn always runs fully from edge to edge, and the mode is called "periodic" because the weft yarns will be automatically connected upon attaching several copies of the same block. This mode is useful to visualize and simulate repeated patterns efficiently. The wraparound mode is used for a block that takes up the whole

width, so the weft yarns wrap around at the last warp they interlace with and do not go all the way to the edge. This mode better describes the weft paths in the fabric considering the actual weaving process. More details on the weft wraparound mode can be found in Section 4.2.

A key component of the Weavecraft interface is the integration of the 2D and 3D views, so that users who are accustomed to designing slices and card images can easily understand the correspondence with the block structures before transitioning into 3D weave design. 2D slice editing is convenient for specifying the detailed structures of weave blocks, and can be found in traditional manual weave drafting and conventional textile design software. 3D visualization assists in understanding the complete logical structure of the material, as well as the 3D deformations in the physical structure. An illustration of the user interface can be seen in Figure 5.

While the user is designing and editing the weave structures, the corresponding card images are automatically created and updated, allowing the user to focus on design rather than low-level machine instructions. Once the design is finished, the card image is ready to be fed to the loom for manufacturing.

The weave block representation and operations are explained in more detail in Section 4. Section 5 elaborates our improvements upon the existing simulation techniques to support relaxation of 3D weave structures in the design process. We create a variety of examples in Section 6, including a more complicated shoe, to demonstrate the range of fabrics that Weavecraft is capable of designing.



(a) Slice View (b) 3D View (c) Clipping Plane

Fig. 5. **Screenshots of our design tool interface.** (a) The slice editor shows the grid of weft yarns and curves of warp yarns with numbers representing their order in the card image and loom. The user can click on the yarns and see a list of available operations (e.g. enabling/disabling a weft yarn). (b) The 3D viewer shows the 3D model of weave block, and a list the available blocks on the right. The user can click on a face of the block and do the assemble operations (e.g. attaching another block to the current block). (c) A clipping plane can hide part of the 3D model and highlight the cross section of yarns that intersect the plane, to help the user understand the internal structure.

4 WEAVE BLOCK REPRESENTATION

4.1 Definition and Representation

A weave block is a tilable unit of 3D weave structure, with several weft yarn segments and warp yarn segments interlaced together. Each weave block has an associated card image that defines how the yarn segments are interlaced. By convention, `'1'` (white) in the pattern means that the corresponding weft segment is on top of the warp segment, and `'0'` (black) means the warp is on top of the weft. The structure of a woven composite is determined by both the card image and the setup of the loom. It is often hard to discern the 3D structure solely from the card image, especially when the structure

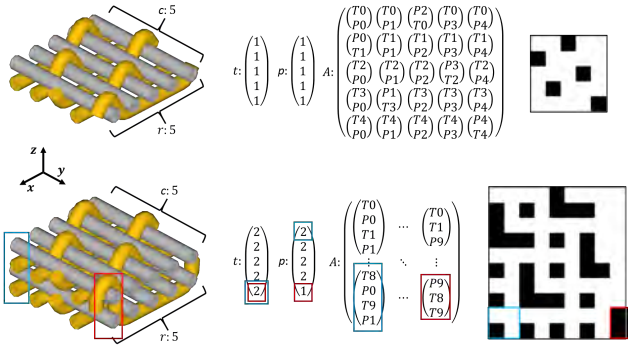


Fig. 6. **Examples of our weave block representation.** Top: a one layer satin weave. Bottom: a two layer weave example. We show the 3D structure of the weave block and corresponding block representations. Weft yarns are in grey and warp yarns are in orange. Stacks in the weave blocks, as well as corresponding representations and card image blocks are marked in color. In the card image, white represents weft above warp and black represents the opposite.

has a significant number of layers. For instance, it is difficult for a novice to tell the relationship between the card image and the structure of the fabric even for the two-layer example in Figure 6. Although experienced designers are capable of reading card images, a more structured representation is desired for 3D weave design. In Weavecraft, we interpret the logical structure of a weave block as a 2D array of *stacks*, which can be seen as multiple warp-weft yarn crossings aligned vertically at a certain point. We represent a stack as a 1D array of symbols that specify the order of weft and warp yarns, in which `'P'` represents warp, `'T'` represents weft and `'E'` represents empty space. Each `'P'` and `'T'` is followed with an index specifying which warp or weft yarn it corresponds to. For example, `a` = $\begin{pmatrix}\texttt{'P0'}\\ \texttt{'P1'}\\ \texttt{'T0'}\end{pmatrix}$ represents a stack of three yarns, where the top two yarns are the first and second warps in this block and the bottom yarn is the first weft. In our convention, warp yarns travel in the x or column direction, while weft yarns go in the y or row direction.

More precisely, we represent a weave block with a set of parameters {`r`, `c`, `h`, `t`, `p`, `A`} , where `r` and `c` are the number of rows and columns and `h` is the height; `t` is a vector of `r` integers where `t[i]` is the number of wefts in the i-th row; `p` is a vector of `c` integers where `p[j]` is the number of warps in the j-th column; and `A` is an `r`-by-`c`-by-`h` array where `A[i][j]` is the stack that lies in the i-th row and j-th column (see Figure 6).

Yarn index constraints. A valid weave block must obey some constraints on the yarn indices in the array of stacks. We use the operation `sum(x)` to denote the sum of all elements in a numerical array `x`, and `sum(x[:i])` to denote the sum from the first up to (but not include) the $\texttt{i}^{th}$ element in `x`. The total number of warp yarns in the block is the sum of the entries of `p`, and every warp index in the range `[0, sum(p))` must occur exactly once. Furthermore, though warps within a stack may appear in any order, all warps

that appear in a particular column must be higher numbered than all warps in earlier columns. This means warp yarns can move up and down within their column, from one row to the next, but they cannot cross into other columns. The analogous statements are true of the wefts. Formally, in stack `A[i][j]`, the range of weft indices is `[sum(t[:i]), sum(t[:i+1]))` and the range of warp indices is `[sum(p[:j]), sum(p[:j+1]))`, and each index appears exactly once in this stack.

Card image. The number associated with each `'P'` and `'T'` determines which column/row the warp/weft yarn corresponds to in the `sum(t)`-by-`sum(p)` card image `I`. Therefore, it is straightforward to derive the card image from the weave block representation. In each stack, if `'Tx'` is above `'Py'`, then the corresponding element `I[x][y]` in the card image is `'1'`, otherwise it is `'0'`. The yarn index constraints laid out above guarantee that the whole card image will be covered by all the stacks.

Yarn Curve Generation. Given a block {`r`, `c`, `h`, `t`, `p`, `A`}, we need to generate the centerline curves for all yarns in order to do yarn-level rendering and simulation. The initial yarn curve model is regular based on the block structure. We first derive a set of key points from the block representation and then interpolate the key points to create the complete curves. Figure 10 shows the initial models generated for several basic 3D weave structures. More details are in the appendix.

4.2 Weft wraparound

In shuttle weaving, the weft in the entire fabric is a single continuous yarn, which is carried by the shuttle that moves back and forth. After each pick, the shuttle turns in the opposite direction, letting the weft yarn wrap around the first warp yarn that catches it. By arranging the position of warp yarns during these two consecutive picks, it is possible to make the weft wrap around in the middle of the fabric. This is an important feature of shuttle weaving and is very useful for creating complicated shapes in 3D weaving. For example, this mechanism is used to create the thin middle part of the I-beam in Figure 14, and to close the sides of the shoe in Figure 4. We refer to [Wu et al. 2020] for more details about weft wraparound. Most of existing design softwares does not consider this mechanism, so the designers need to figure it out on their own, which is very difficult without good visualization. In Weavecraft, we provide a mode which figures out the connections between consecutive weft yarns based on the order of their numbers, and then calculates the right position for the weft wraparounds (Figure 7).

4.3 Assembling Weave Blocks

Once the designer builds some blocks, they can assemble them to create more complicated structures. We define several useful operations for assembling weave blocks. Yarn indices are carefully rearranged to ensure that the assembled block satisfies the yarn index constraints.

Tiling a block. A block can be repeated along any of the three axes to create a larger tile. When a block is tiled, the ranges of yarn indices are expanded, and the numbers are regenerated to satisfy the yarn index constraints (see details in appendix). In each repeat, the relative order of yarn numbers is maintained to be the same as in the original block.

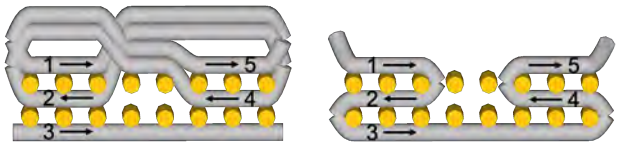


Fig. 7. Weavecraft has two modes for visualizing weft (gray) yarn paths. In this example, the numbers indicates the order of the wefts, and the arrow indicates the direction of the shuttle for each pick. Left: in periodic mode, the boundary of each weft yarn is periodic. Right: in weft wraparound mode, Weavecraft figures out the connections between two consecutive weft picks based on yarn index, and calculates the right location for the wraparounds.

Attaching two blocks. Two blocks can be attached to each other as long as they satisfy certain constraints. For example, to attach two blocks {`r1`, `c1`, `h1`, `t1`, `p1`, `A1`} and {`r2`, `c2`, `h2`, `t2`, `p2`, `A2`} in the x direction, they must satisfy the constraints that `c1 == c2` and `p1 == p2`. A similar constraint applies when attaching two blocks in the y direction. For attaching in z direction, the constraints are `r1 == r2` and `c1 == c2`. The user is expected to ensure these constraints when doing the operation, and the system reports errors upon incompatible block compositions. Algorithm 2 and Algorithm 3 in the appendix shows how to compute yarn indices when attaching in the x and z directions; the y direction is similar to x.

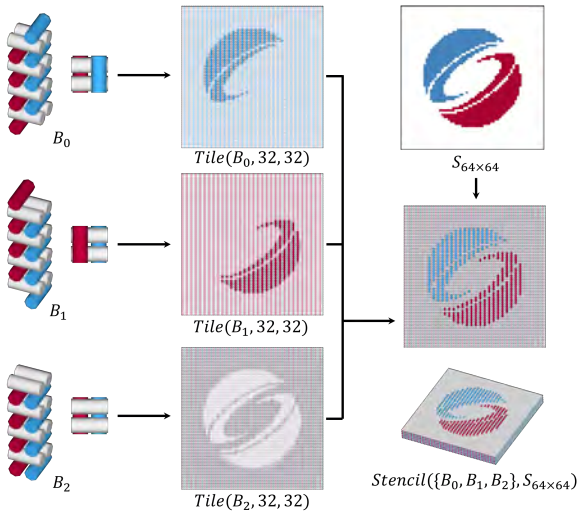


Fig. 8. **Illustration of stenciling.** To merge several blocks with a stencil, we first tile every block respectively so that the x and y sizes of the tiled blocks are no smaller than the stencil. Then for each pixel in the stencil, we grab a stack from the corresponding tiled block according to the index value. The merged block is a combination of these stacks.

Stenciling blocks. Bringing a core tool of 2D jacquard design to the 3D context, our stenciling operation provides a way to create

fabrics with complex, spatially varying structure. Stenciling uses an image to merge different blocks with a certain pattern.

The inputs for the stenciling are N different weave blocks {Bi={ri, ci, hi, ti, pi, Ai} , i=0,...,N-1}, and an x-by-y stencil image S. The stenciling operation first tiles all blocks to cover the size of the stencil image, then it grabs stacks from the tiled blocks according to the block indices specified in the stencil image. Algorithm 4 in the appendix and Figure 8 illustrate how this operation works.

To create a valid stenciled block, the indices in the stencil image must be within range, and the tiled blocks must have matching numbers of wefts and warps in corresponding stacks. Let x be a m-by-1 vector and n be a scalar. Define repeat(x, n) as an m×n-by-1 vector that makes n copies of x. Then repeat(ti, rj) == repeat(tj, ri) and repeat(pi, cj) == repeat(pj, ci) need to be satisfied for all i and j.

The results of the above operations are valid weave blocks themselves and can used in other operations. For example, several narrow blocks that are easy to build can be attached to form a repeatable weave structure. Subsequently, stenciling or tiling can be used to expand one or several such blocks to an entire fabric, so the user can create a hierarchy of weave blocks using the assembling operations. The hierarchical block representation allows the user to propagate changes on the child blocks to the parent blocks (see supplemental video).

4.4 Yarn Flipping

Our interactive tool allows the user to make changes to existing weave blocks while running physically-based simulation (Sec. 5.2). To provide a simple user interface and reliable simulation updates, we only allow the user to make one kind of change to the topology: Each yarn edit can only flip one pair of weft and warp yarns, i.e. invert one pixel on the card image.

When a yarn flip is requested by the user, we identify the indices of the corresponding weft and warp yarns. The weft and warp index pair exists in exactly one stack in the block, due to the yarn index constriants mentioned in Sec. 4.1. The positions of the two corresponding elements in the stack need to be swapped. To ensure that swapping the target pair does not affect other yarn pairs, we use Algorithm 1 in the appendix to check whether the flipping is valid and reject invalid flipping requests.

5 SIMULATION

The geometric structure of a woven pattern often differs considerably from the schematic, grid-based plans used for deriving the card image. The final position of the yarns in a fabric is determined by the inter-yarn forces during and after weaving, and it is easy to create designs that cause yarns to deviate from their expected positions, impacting the quality of the woven result. Recent work on yarn-based cloth simulation [Leaf et al. 2018] demonstrates rod simulations with large numbers of contacts running at interactive rates, and our experiments show this type of simulation is able to predict many aspects of the behavior of yarns in 3D woven materials. Having this kind of simulation available during the design process can drastically reduce the need for iterations of test weaving, since many errors and problems can be corrected before test production, and the simulation can provide useful guidance in adjusting the model to correct problems that do arise in testing.

Our simulation is based on a GPU-based rod simulator implemented according to [Leaf et al. 2018] to do yarn-level relaxation, which is able to relax models containing thousands of control points in minutes (see Table 1).

5.1 Technical Improvements

Our key improvements on the yarn simulation from [Leaf et al. 2018] include a novel strain balancing algorithm, support for heterogeneneous yarn types, and a mechanism for changing yarn topology while simulation is running.

Strain balancing. The rest length of the rods is an important parameter that reflects the amount of yarn taken up by the fabric during weaving. Since 3D weaving uses a separate yarn supply per warp end, warp yarns are individually kept at a constant tension. This ensures that as warp yarns wrap above and below weft yarns, enough material is fed into the fabric as necessary to construct the pattern; in the simulation this manifests as warp yarns of different length, but the same tension. In contrast, weft yarns are straight when inserted into the fabric, so they are closer to constant length, with variable tension. Because of the way warp take-up works, the warp rest lengths are not known a priori; the pattern initialization from the block representation is designed to ensure that the topology is correct, but does not reflect the yarn lengths that would actually be present in the fabricated material; these lengths must be discovered through simulation.

To address this issue, we employ a strain balancing algorithm that updates the rest-lengths of yarn segments during simulation to match a design strain value. That is, the following relationship is enforced during the simulation:

$$L_i = \hat{L}_i / S$$

Where $\hat{L}_i$ is the current length of spline segment i, S is the target strain, and L_i is the rest length of spline segment i. In our implementation, we require the target strain satisfy $S >= 1.0$ to prevent a positive feedback loop. We also require that boundary conditions are present, either periodic or frozen, to prevent yarn segments from shrinking indefinitely. Using this approach, the pattern can re-balance to a configuration that normalizes the relative strain per spline segment, which corresponds to a uniform tension—the condition under which warp yarns are woven.

Heterogeneous yarn types. Unlike [Leaf et al. 2018], our simulator permits heterogeneous yarn types. Each yarn can have a different radius and different physical parameters.

To support collisions across a variety of yarn radii, we implement spatial hashing and set the grid cell size to be the largest yarn diameter in the simulation. This is sub-optimal, as having a single yarn with a large radius will cause the spatial grid to be large, reducing the efficiency of culling potentially colliding contact primitive pairs. However, for the patterns we simulated, this compromise in speed was not so large as to prevent interactive design.

In our experiments, we found that all yarn parameters need to be tuned for the type of yarn material present in the model. For each of the examples presented, we allow the user to define physically based

parameters separately in the weft and warp directions, a feature that is critically necessary for simulation to match real 3D woven fabrics, which very often use quite different yarns in the warp and weft. Details about the parameters and simulation performance can be found in Section 6.

5.2 Integrating with Weavecraft

The application of yarn simulation in Weavecraft intends to predict the flaws of a design pattern and verify the fixes in real time. This requires not only resolving the intra-yarn collisions upon initialization so that the model can reach a relaxed state with minimal parameter tuning, but also validating the yarn-level edits without reinitializing the entire weave pattern in the simulator.

Simulation setup. After generating the yarn curves, we use them to define the initial state of a set of rods in the simulator, and let the model relax under both intra-yarn (stretching and bending) and inter-yarn (collision) forces.

The rod radii are important parameters that affect the collision forces used to model interactions between yarns. Although the yarns are usually tightly packed together in 3D weaving, initializing the simulation with such tight configurations can create excessive contact energy, because in our regular initial model the yarns often have paths right next to each other. Therefore, we run a short simulation to resolve collisions and place yarns in a configuration where they are no longer strongly intersecting. We then allow the user to scale up the radii of the yarns using the radius scaling technique described in [Leaf et al. 2018] to adjust the global amount of material used by the pattern.

In the periodic mode, each weave block is a repeatable unit by nature so we can apply periodic boundary conditions. The periods in the x and y directions are set as the size of the initial model and remain fixed during the simulation. We can also merge all weft picks in to a single continous yarn to simulate weft wraparounds discussed in Section 4.2

We allow the user to change some parameters during the simulation, including the target strain, stiffness and yarn radius, etc. To avoid unstable simulation, we limit the range of parameters that the user can set, and we also adopt adaptive timestep as in [Leaf et al. 2018]. When the user changes the target strain, the rest length is gradually updated in each timestep until the target strain is reached, to increase stability. With these measures, we do not encounter unstable simulation for the results presented in this paper.

Fig. 9. **Transition Pairs in Action:** Two pairs are defined: (1) the blue yarn should be above the red yarn, and (2) even farther above the green yarn. (Left) The initial configuration of the yarns before the transition pairs spring forces are applied; (Middle) transitions in progress; and (Right) the final configuration after the transition pair spring forces have done their work.

Interactive yarn flipping. A key challenge in interactive design is to allow users to modify the pattern *during* relaxation. Suppose a user simulates a complex pattern and wishes to change the structure of the fabric by modifying the stacking of weft and warp yarns. Prior simulation methods would require the user to edit the topology of the pattern in the block representation, reinitialize the simulation state, and re-run the simulation, resulting in a delay even for a tiny change to the pattern. To avoid this scenario, a way to modify the pattern during simulation is needed.

The yarn movement during relaxation depends on the topology of the pattern, the simulation parameters, as well as the external forces applied on the pattern, and is therefore difficult to predict. As yarns are free to slide during the simulation, the spline segments might steer away from their initial positions, and the yarn collisions cannot be maintained as a consequence. We propose a set of methods to define the topology of the pattern, transition yarn segments through each other, and ensure that the transitions remain reliable throughout the simulation.

The topology of a weave pattern can be encoded as the crossing of yarns projected onto a 2D plane. We propose a method for swapping the stacking of crossing yarns during simulation, which can be applied to an arbitrary pattern. To initiate swapping the stacking of two yarn segments, we introduce a projection axis $\hat{v}$ that defines the 2D projection plane, the relative target height h, and the yarn segments s_i, and s_j. In Weavecraft, $\hat{v}$ is defined along the z-axis and perpendicular to the top layer of a weave pattern. A transition pair is defined as a pair of yarn segments (s_i, s_j). When a transition pair is active, collisions are disabled between the two segments in the pair. Subsequently, the yarn segments are projected onto the 2D plane normal to $\hat{v}$. If a crossing is found, then the height $\hat{h}$ between the yarn segments is calculated. If $|h - \hat{h}| > \epsilon$, then a spring force is applied to move the yarn curves to the target height h. Otherwise, the transition pair is disabled. See that in Figure 9 this method produces a successful set of transitions. The blue yarn is successfully moved above the green and red yarns in this example.

Section A.4 in the appendix discusses the edge cases of this approach.

6 RESULTS

In this section, we demonstrate a range of examples created with our tool, from simple two-layer woven cloths to 3D woven composites and a complicated 3D woven shoe.

Two-layer satin and twill cloths. Figure 11 shows the simulated results of two two-layer satin and twill cloths, compared with CT scan data. More two-layer examples can be found in the supplemental materials. Note that the simulation predicts the diagonal dent in the Twill G example. Figure 1 and the supplemental video show the procedure of interactively designing a three-color cloth using different two-layer patterns. With the help of simulation and live edits, the user is able to fix defects in the design.

Basic 3D Weave Structures. Common 3D weaving structures can be conveniently designed and simulated with our tool. In Figure 10 we demonstrate several common 3D weave structures created with our tool. We show the slice sketch design of the structures, the card

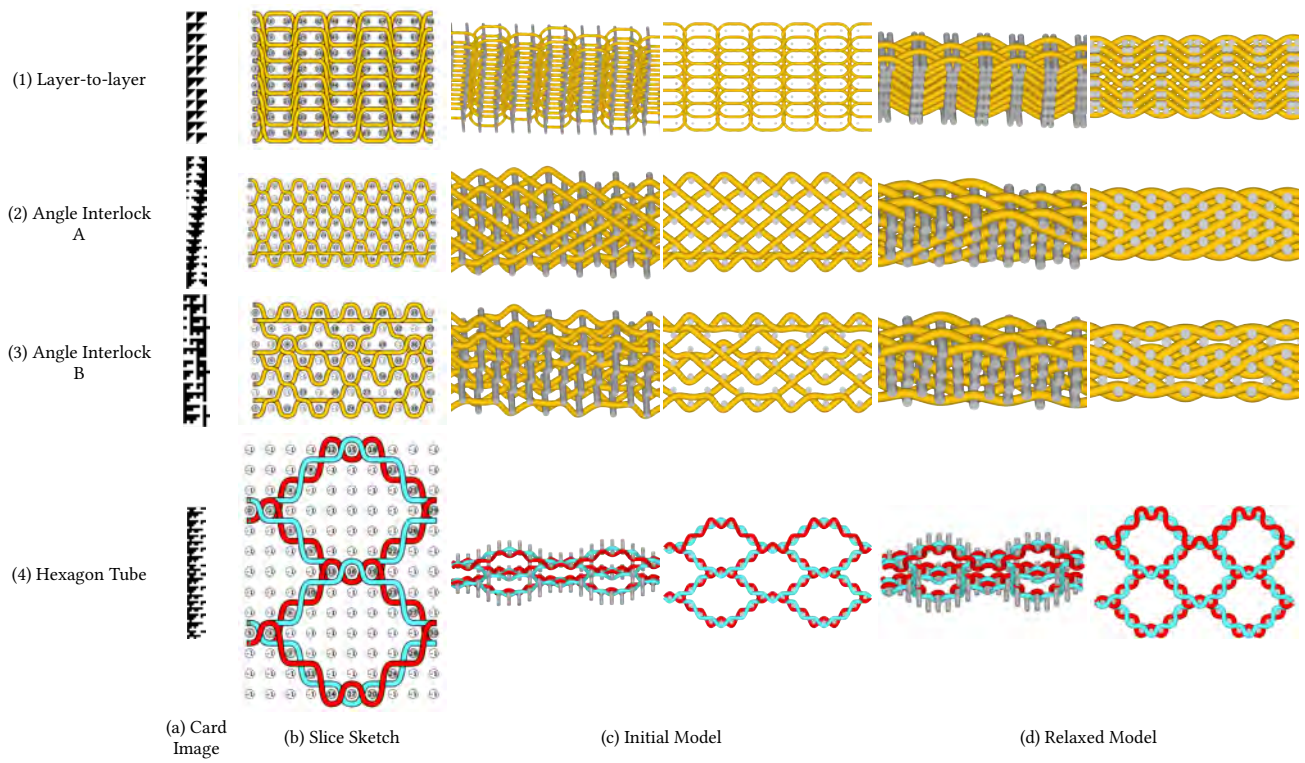


Fig. 10. **3D Weaving Examples**

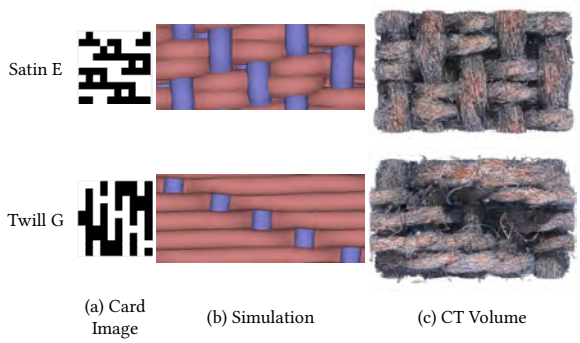


Fig. 11. **Two-layer woven cloth examples** with simulated results shown beside CT scans for reference.

Table 1. **Simulation Statistics:** For each pattern, a simulation was run with a target strain of 1% to convergence. We report the simulation time, time per step, and number of control points below. Some patterns, such as Angle Interlock A, have initial configurations far from the minimum, requiring a large number of step for convergence. TL013 has a particularly high time per step because it has weft yarns that have 3x the radius of warp yarns. The additional cost of contacts for variable radii yarns is described in §5.1.

Structure	Simulation Time [s]	Time per step [ms]	Number of Control Points
Satin E	3.16	1.21	692
Twill G	5.1	1.19	711
Layer-To-Layer	10.5	2.84	3075
Angle Interlock A	29.54	1.8	1867
Angle Interlock B	60.49	1.61	1444
Hexagon Tubes	81.13	1.76	2127
TL013	62.49	14.88	4491
Pocket	15.5	7.75	12751

image, as well as the simulation results. For the hexagon example, weft yarns in the top and bottom are pinned to a fixed position, to simulate external force preventing the fabric from collapsing under warp tension. Table 1 shows the performance statistics of the simulation. Interesting yarn motion and deflection are predicted by the simulation. For instance, in *layer-to-layer*, two neighbor weft yarns are pushed close to each other by the warp yarn structure.

A 3-layer fabric (TL013). This fabric, consisting of a zigzag layer connected with two double-cloth surface layers, was originally designed manually without Weavecraft. During the design process the designer made an unnoticed error, causing the upper section of the zigzag layer to lean sideways when the fabric was woven. The error is revealed by a CT scan (Figure 12(1c)). We replicated the erroneous design in Weavecraft and simulated it. The simulation predicts the sideways leaning in the middle section as shown in Figure 12(1b). In the supplemental video we demonstrate how this error could have been fixed during the design process with simulation and interactive editing. The corrected simulation result, as well as the card image automatically updated by our tool, is shown in Figure 12(2).

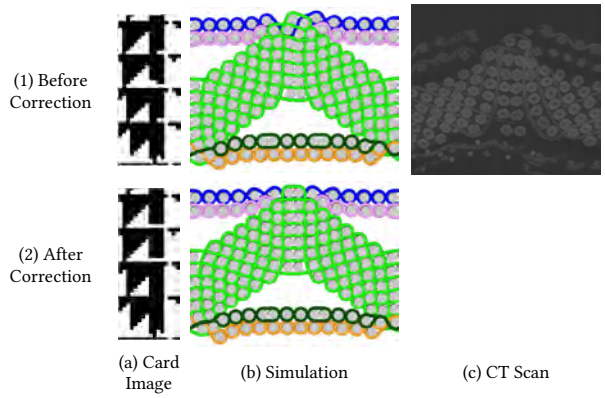


Fig. 12. **CT Scan of TL013 and simluation result.** The CT image is a single slice of the 3D volume, so not all warp yarns are shown in it.

Pocket. Our simulation methodology is not limited strictly to periodic relaxations or yarn flipping. A user has designed a pocket in Figure 13. Using user-defined spherical force-fields, the user visualizes the pocket once it has been pulled open. In this simulation, periodic boundary conditions are removed, but inverse mass filtering is applied to prevent significant yarn sliding along the boundaries. Allowing users to visualize their products in a physical use case is key for ensuring the pattern behaves as expected.

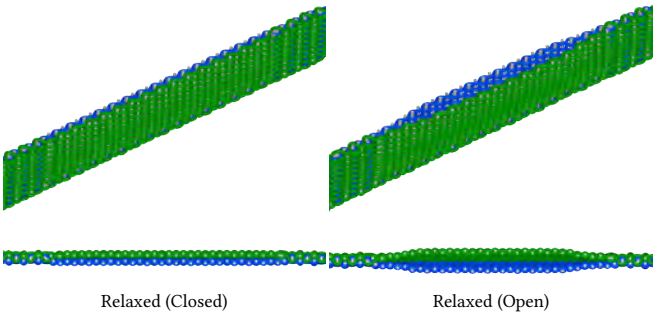


Fig. 13. **Pocket Simulations.** Simulation without periodic boundary conditions can visualize physical environments for patterns, such as opening and closing a pocket. (Left) The pocket is relaxed, but no forces are applied to open it. (Right) Spherical force fields are placed in the pocket to open it.

I-beam. The development of Weavecraft is in close collaboration with professional designers working in an industrial facility. They have already been using our prototype in their production work. Figure 14 demonstrates an example carbon fiber I-beam composite created with Weavecraft for their production. Note that the purple warp yarns do not interlace with any weft yarn, and are only place holders to accomodate the loom setup. Weavecraft can figure out the right weft wraparound locations and correctly visualize the thin part in the middle.

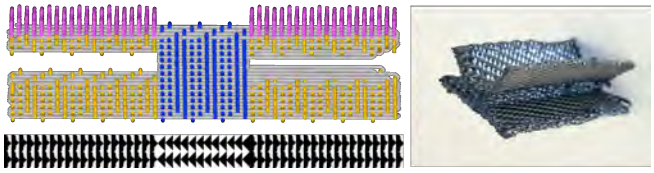


Fig. 14. **An I-beam composite designed with Weavecraft.** We show the block model before being tiled in the warp direction, as well as the card image and the fabricated result using carbon fiber yarns.

6.1 Case Study: Footwear

With our tool, large scale 3D woven objects can be designed, such as the shoe created by [Harvey et al. 2019]. To demonstrate how the features of our system support such complicated design, we use two toy examples to illustrate the design process, both of which were created within a few hours in Weavecraft.

Baby shoe. In Figure 4, we demonstrate a much simplified shoe with only two layers: the top layer and the sole layer, both being plain weave patterns. Six blocks and three stencil images are used to create the three parts of the shoe: the heel, the middle, and the toe. We choose a perspective that better shows the weft (grey) path in each block. Slice 1 is composed with Block A-C, each of which has four weft yarns, two in the top layer and two in the sole layer. Note that the wefts in Block B do not interlace with any warp yarns, so when it is put on the two sides, the part of weft will not remain in the fabric because of the weft wraparound mechanism, and the two layers will be connected by weft wraparounds. Slice 2 is from the middle part, which has a hole in the middle. Each of the Blocks D to G has six wefts: two for the left top layer, two for the right top layer and two for the sole layer. The weft wraparound mechanism creates two separate top layers on the left and right respectively. Many warp yarns in the result do not interlace with weft yarns, which is common in complicated 3D woven shapes, as loose warps are trimmed during post processing. In particular, the middle part has a circular opening for the ankle after trimming the unwoven warp segments.

Toddler shoe. Figure 15 demonstrates a slightly more complicated shoe, which is made from nine blocks and three stencil images in a similar way as the baby shoe. The weft yarns in the blocks are carefully renumbered (colors of yarns indicates correspondence among different blocks), so that they take advantage of weft wraparound simlar to the previous example. Figure 15 uses a different perspective to better illustrate the warp paths. The warps are arranged into three layers (shown in Slice 3): the sole, the insole, and the top, each of which has a different structure chosen specifically for its functionality. The warp yarns in the sole layer (dark blue) form a zigzag shape to increase friction. The sole layer (red) is a thick layer-to-layer structure for cushioning. The top layer (orange) is made thinner for air permeability. Note that block B, G and H only have two thin layers and are placed on the edge of the shoe, so they can be unfolded afterwards to form the sides of the shoe.

Adult shoe. The sophisticated shoe in [Harvey et al. 2019] was also created in Weavecraft following a similar approach. The design

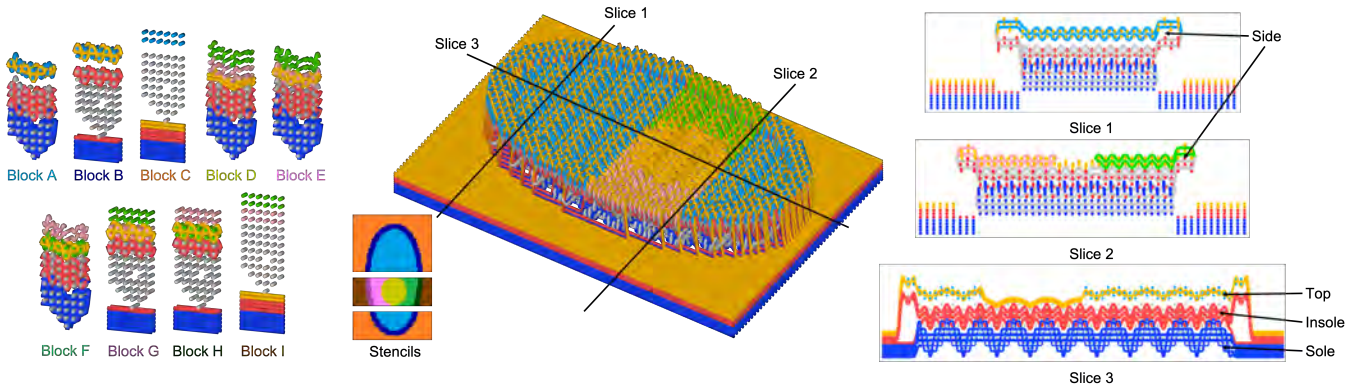


Fig. 15. **Design pipeline of a simplified shoe:** Left: 9 different blocks are used to create the shoe. The colors of block names correspond to the colors on the stencil. Middle: three stencil images are used to make the heel, the middle and the toe parts, which are then attached to create the full shoe. Right: Intersection planes are used to show the internal structures of the shoe.

(a) (b)

Fig. 16. **Details of two areas on the shoe.** (a) Viewing from the side, the upper layer is on the left and the sole layer is on the right. (b) On the left part, warp yarns are designed not to interlace with weft yarns, allowing easy trim in post processing.

pipeline is shown in Figure 17. Similar to the previous example, the warp yarns are roughly divided into three sets (See Figure 16(a) and Figure 17 B67): the sole, the insole and the upper, each with specific material to allow for localized design of rigidity. For example, the warp yarns for the sole are thick PVC coated polyester yarns, and the sole is made into a zigzag shape to increase rigidity and friction. The warp yarns for the insole and upper are soft cotton yarns for comfort, and areas of the insole are customized to specify areas with more or less cushion. Customization of the upper fabric incorporates woven seams, and eliminate the need for sewn reinforcement required to finish the shoe. Regions of the upper fabric are designed with different patterns for visual appeal, as well as functional regions such as the shoelace eyelets. In the sole layer, warp yarns outside of the sole shape are designed not to interlace with weft yarns (See Figure 16(b) and Figure 17 B0), and are trimmed away while the cut ends are reinforced with glue during post processing. Throughout the different weave structures, attachments between layers are customized to allow for the ultimate folding of the shoe into its final volume.

The design of the full shoe was the motivating problem that drove Weavecraft's development and was done with previous versions of the system. It took about 300 hours of design time, and 20 iterations of actual fabrication (around 100 hours including setup and weaving time). The whole process spanned three months. Redoing it with the final system would be much faster.

As we have presented, this example is very complicated and spatially varying, with different structures designated in different parts of the fabric. It would be nearly impossible to design such a complex form without the control and convenience provided by our tool. Although the full shoe cannot be simulated interactively due to simulator performance limitations, the idea of verifying the pattern with simulation can still help the design process: many design iterations are dedicated to establishing basic weave structures and fitting them together and can be addressed by small simulations. The simulation can also inform decisions on warp setups, which are very costly (days per iteration) to experiment with on the loom.

7 FORMATIVE USER EVALUATION

We conducted a formative user evaluation to better understand how novice 3D weavers react to our system of 3D visualization and simulation in the design process. The study was conducted as part of a textile class running once a week, so the training session and the actual study were one week apart, while the students tried out the software during the week. Our study group consists of three students, with weaving experience ranging from one to five years, and some experience with traditional textile CAD software such as PointCarre.

7.1 Procedure

Prior to the study, we provided our participants with a brief introduction to Weavecraft alongside a 1-hour tutorial of the software introducing the weave block operations and the interactive yarn simulation. We also provided example weave blocks created using Weavecraft as well as a user manual (attached in supplemental materials). The study took place one week later and lasted 30 minutes. Given the requirement for social isolation due to COVID-19, the study was conducted on the participants' own computers, which accessed a simulation server hosted on Amazon Web Services. We asked the participants to design a woven pocket in four steps, namely, creating a one-layer rectangular fabric of a given graphical

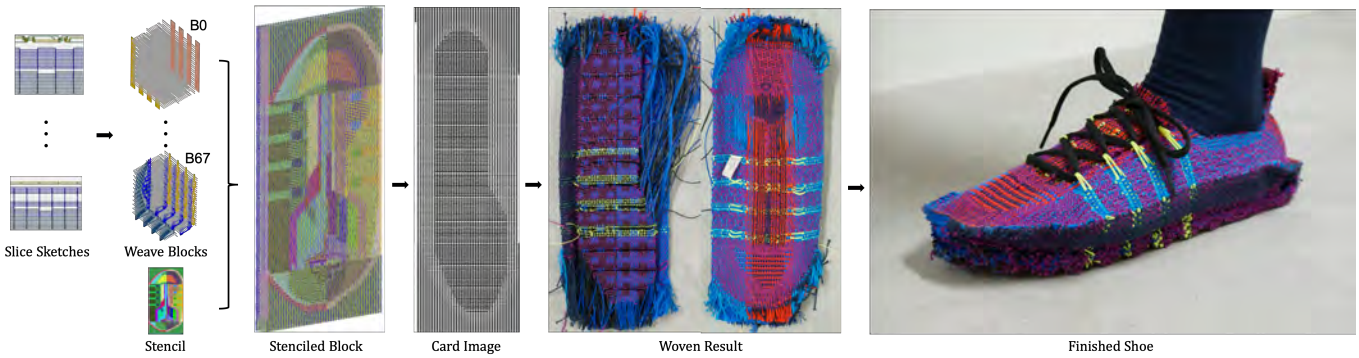


Fig. 17. **Design pipeline for a 3D woven shoe.** The shoe is created by stenciling 68 different blocks.

pattern, simulating the fabric, adding a second layer of the same size, and attaching the two layers to form a pocket. Finally, we conducted a group interview with the three participants after the study and gathered their feedback. We did not ask the participants to fabricate their patterns due to restricted access to the loom.

7.2 Results

We observed the behavior of our participants through screen-sharing in remote conference calls, and took notes with an emphasis on the operations used for each task and the places where the participants got stuck. All three participants completed the pocket design task within the 30-minute interval.

Patterning. For the first subtask, the participants were given a graphical image and the card images for two 2-3 satin patterns, and were asked to create a one-layer fabric of 25 warps and 35 wefts of the given graphics from the two satin patterns. Two of our three participants successfully created the pattern without guidance using different approaches: one loaded the given card images to create the stencil, the other used our preset satin patterns, and both were able to change the yarn colors and complete the stenciling operation. Our third user attempted to create the structure in the slice editor and failed, but still completed the subtask upon being directed to the stenciling feature.

Simulation. Upon creating the one-layer fabric, the participants were asked to simulate the pattern and make yarn edits where necessary. All three participants were able to simulate the pattern to a converged state and identify flaws in the pattern such as a long yarn floating out of the pattern or yarns bumping into each other, before applying yarn flip operations correspondingly to resolve the issues. Participants also quickly learned how to tune the parameters to make the yarn appear "tighter".

Attaching two layers. In the third subtask, the participants were asked to create a second layer of the same size from a 4-1 twill pattern whose card image was provided. The two layers had to be attached and sealed on the two long sides by turning on weft wraparound. All our participants had trouble finding the block attachment and weft turnaround options by themselves, and completed the operations under oral guidance.

Sealing. Finally, we asked the participants to seal one short side of the pocket by connecting the two layers using weft yarns at the seam. Our participants struggled with the yarn flipping operation, before realizing that they were only allowed to flip a single pair of weft and warp yarns at a time.

7.3 Post-Study Interview Feedback

All our participants found Weavecraft to be helpful in assisting the design and understanding of multi-layered patterns. With the slice and block visualizations, participants were able to see how the yarns interact from different perspectives: "I had a hard time understanding the gauze weave pattern when designing in PointCarre because it was too flat, but I finally understood the pattern when playing with it in WeaveCraft. It is a good visual simulation tool that displays the exchange of yarn layers." The notion of building from blocks provided a different mindset for textile design, and our participants were inspired to create patterns consisting of more layers as well as more complicated surface designs.

Our participants stated that the yarn simulation results generally conformed with their expectations of the patterns being woven on an actual Jacquard loom, and particularly found the target strain parameters useful in controlling the tension on the yarns: "The simulation could be quite powerful once the parameters are fully understood, and provide approximate mappings to the actual loom setup." The ability to flip yarns during the simulation was also helpful in identifying and resolving flaws in the design patterns.

Our participants did not find the UI very inviting, and were overwhelmed by the blank windows upon launching as well as the unfamiliar terminology used in the software. Additionally, participants suggested that the navigation of the visualizer could be improved: "Directional hints that tell you where you are in 3D space and the ability to freeze the space to prevent unintentionally moving around can be really helpful." Nevertheless, they stated that the learning curve of Weavecraft was comparable to traditional textile software, and that the design process became clearer with some practice. In particular, operations such as slice editing and block stenciling were intuitive and familiar, while operations including block attachment and yarn flipping were harder since they were less familiar from the experience in existing textile software. The difficulty in grasping

the controls was expected from exposure to a prototype software, and could be easily addressed.

8 CONCLUSION

In conclusion, we believe that 3D weaving, beyond its traditional application to composite materials, is a new and general way of making soft objects with complex geometric structures, whose application has been restricted by the lack of design tools. Weavecraft is the first tool that tightly integrates multi-scale modeling, 3D visualization, physically based simulation, and interactive editing for 3D weaving design, which opens up the possibility to explore novel 3D woven objects.

Limitations and Future Work. While our tool has several limitations, it opens numerous opportunities for future development and has the potential to fundamentally change the way that 3D woven fabrics are designed.

Simulation is one aspect of the system that can always be improved. Due to limitations on GPU memory and convergence speed, currently we cannot simulate large-scale models such as the full-sized shoe at an interactive rate. Future development could decompose a large model into small pieces and use distributed computation resources for simulating them. Additionally, incorporating more complex yarn cross-section deformation models would help to enrich the simulation results to match even more closely to reality, such as the I-beam example (Figure 14) where the untwisted carbon fibers are flatten to thin elliptical shapes. Predicting the physical and functional properties of the woven results is also a promising future direction. For example, shoe manufacturers might be interested in how rigid the shoe sole is and how soft the other parts are. By combining various yarns with different physical properties, one can obtain an object with specific physical property, as described in [Bickel et al. 2010].

Another exciting direction is to further automate the design process of 3D weaving. Currently, our system is designed for professional 3D weavers who want complete control over the details of weave structure. One could imagine building a library of weave structures with various properties and appearances. With such a library, a novice user only gives high-level instructions on the shape or appearance of the fabric, while an algorithm selects the right building blocks from the library and assembles the final artifact automatically.

ACKNOWLEDGMENTS

We would like to acknowledge the management of T.E.A.M. Inc. (teamtextiles.com) for supporting the project through access to their advanced 3D weaving facility in Woonsocket, RI, as well as expert technical assistance. We thank the support from the RISD Virtual Textile Research Group. We also thank Eston Schweickart for the yarn rendering code. This work is funded by the National Science Foundation under grants IIS-1513954, IIS-1513967, IIS-1644523 and DGE-1656518, and by generous donations from Under Armour. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or Under Armour.

REFERENCES

Lea Albaugh, Scott Hudson, and Lining Yao. 2019. Digital fabrication of soft actuated objects by machine knitting. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

Mahmood Ansar, Wang Xinwei, and Zhou Chouwei. 2011. Modeling strategies of 3D woven composites: a review. *Composite structures* 93, 8 (2011), 1947–1963.

Bernd Bickel, Moritz Bächer, Miguel A Otaduy, Hyunho Richard Lee, Hanspeter Pfister, Markus Gross, and Wojciech Matusik. 2010. Design and fabrication of materials with desired deformation behavior. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–10.

Xiaogang Chen. 2011. Mathematical modelling of 3D woven fabrics for CAD/CAM software. *Textile Research Journal* 81, 1 (2011), 42–50.

Xiaogang Chen, Lindsay Waterton Taylor, and Li-Ju Tsai. 2011. An overview on fabrication of three-dimensional woven textile preforms for composites. *Textile Research Journal* 81, 9 (2011), 932–944.

Gabriel Cirio, Jorge Lopez-Moreno, David Miraut, and Miguel A. Otaduy. 2014. Yarn-Level Simulation of Woven Cloth. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH Asia)* 33, 6 (2014). http://www.gmrv.es/Publications/2014/CLMO14

Gabriel Cirio, Jorge Lopez-Moreno, and Miguel A Otaduy. 2015. Efficient simulation of knitted cloth using persistent contacts. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 55–61.

Gabriel Cirio, Jorge Lopez-Moreno, and Miguel A Otaduy. 2016. Yarn-level cloth simulation with sliding persistent contacts. *IEEE transactions on visualization and computer graphics* 23, 2 (2016), 1152–1162.

BP Dash, BK Behera, Rajesh Mishra, and Jiri Militky. 2013. Modeling of internal geometry of 3D woven fabrics by computation method. *Journal of The Textile Institute* 104, 3 (2013), 312–321.

GmbH EAT. 2015. 3DWeave | EAT GMBH - The Designscope Company. http://designscopecompany.com/3dweave/

Mikhaila Friske, Shanel Wu, and Laura Devendorf. 2019. AdaCAD: Crafting Software For Smart Textiles Design. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

Ruben Geerinck, Ives De Baere, Geert De Clercq, Jan Ivens, and Joris Degrieck. 2016. Development and characterization of composites consisting of woven fabrics with integrated prismatic shaped cavities. In *3D fabrics and their applications*.

Nur Al-huda Hamdan, Simon Voelker, and Jan Borchers. 2018. Sketch&Stitch: Interactive Embroidery for E-textiles. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.

Claire Harvey, Emily Holtzman, Joy Ko, Brooks Hagan, Rundong Wu, Steve Marschner, and David Kessler. 2019. Weaving objects: spatial design and functionality of 3D-woven textiles. *Leonardo* 52, 4 (2019), 381–388.

Megan Hofmann, Lea Albaugh, Ticha Sethapakadi, Jessica Hodgins, Scott E Hudson, James McCann, and Jennifer Mankoff. 2019. KnitPicking Textures: Programming and Modifying Complex Knitted Textures for Machine and Hand Knitting. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 5–16.

Lejian Huang, Youqi Wang, Yuyang Miao, Daniel Swenson, Ying Ma, and Chian-Fong Yen. 2013. Dynamic relaxation approach with periodic boundary conditions in determining the 3-D woven textile micro-geometry. *Composite Structures* 106 (2013), 417–425.

Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. 2008a. Knitting a 3D model. In *Computer Graphics Forum*, Vol. 27. Wiley Online Library, 1737–1743.

Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. 2008b. Knitty: 3D Modeling of Knitted Animals with a Production Assistant Interface.. In *Eurographics (Short Papers)*. Citeseer, 17–20.

N Isart, B El Said, DS Ivanov, SR Hallett, JA Mayugo, and N Blanco. 2015. Internal geometric modelling of 3D woven composites: A comparison between different approaches. *Composite Structures* 132 (2015), 1219–1230.

Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2008. Simulating Knitted Cloth at the Yarn Level. *ACM T. Graph. (SIGGRAPH'08)* 27, 3 (2008), 65.

Jonathan M Kaldor, Doug L James, and Steve Marschner. 2010. Efficient yarn-based cloth with adaptive contact linearization. In *ACM Transactions on Graphics (TOG)*, Vol. 29. ACM, 105.

Alexandre Kaspar, Tae-Hyun Oh, Liane Makatura, Petr Kellnhofer, Jacqueline Aslarus, and Wojciech Matusik. 2019. Neural Inverse Knitting: From Images to Manufacturing Instructions. *arXiv preprint arXiv:1902.02752* (2019).

Konstantin Klamka, Raimund Dachselt, and Jürgen Steimle. 2020. Rapid Iron-On User Interfaces: Hands-on Fabrication of Interactive Textile Prototypes. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–14.

Jonathan Leaf, Rundong Wu, Eston Schweickart, Doug L James, and Steve Marschner. 2018. Interactive design of periodic yarn-level cloth patterns. In *SIGGRAPH Asia 2018 Technical Papers*. ACM, 202.

Minchen Li. 2018. *FoldSketch: Enriching garments with physically reproducible folds*. Ph.D. Dissertation. University of British Columbia.

Hua Lin, Martin Sherburn, Jonathan Crookston, Andrew C Long, Mike J Clifford, and I Arthur Jones. 2008. Finite element modelling of fabric compression. *Modelling and Simulation in Materials Science and Engineering* 16, 3 (2008), 035010.

James McCann, Lea Albaugh, Vidya Narayanan, April Grow, Wojciech Matusik, Jennifer Mankoff, and Jessica Hodgins. 2016. A compiler for 3D machine knitting. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 49.
Yuyang Miao, Eric Zhou, Youqi Wang, and Bryan A Cheeseman. 2008. Mechanics of textile composites: Micro-geometry. *Composites Science and Technology* 68, 7-8 (2008), 1671–1678.
AP Mouritz, MK Bannister, PJ Falzon, and KH Leong. 1999. Review of applications for advanced three-dimensional fibre textile composites. *Composites Part A: Applied science and manufacturing* 30, 12 (1999), 1445–1461.
MultiMechanics. 2018. Product. http://multimechanics.com/product/
Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. 2018. Automatic Machine Knitting of 3D Meshes. *ACM Trans. Graph.* 37, 3, Article 35 (Aug. 2018), 15 pages. https://doi.org/10.1145/3186265
Vidya Narayanan, Kui Wu, Cem Yuksel, and James McCann. 2019. Visual knitting machine programming. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–13.
Saad Nauman and Irina Cristian. 2015. Geometrical modelling of orthogonal/layer-to-layer woven interlock carbon reinforcement. *The Journal of The Textile Institute* 106, 7 (2015), 725–735.
Jesús Pérez, Miguel A Otaduy, and Bernhard Thomaszewski. 2017. Computational design and automated fabrication of kirchhoff-plateau surfaces. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.
Pointcarre. 1988. Pointcarre. http://www.pointcarre.com/.
Ivan Poupyrev, Nan-Wei Gong, Shiho Fukuhara, Mustafa Emre Karagozler, Carsten Schwesig, and Karen E Robinson. 2016. Project Jacquard: interactive digital textiles at scale. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 4216–4227.
Michael L Rivera, Melissa Moukperian, Daniel Ashbrook, Jennifer Mankoff, and Scott E Hudson. 2017. Stretching the bounds of 3D printing with embedded textiles. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 497–508.
ScotWeave. 2019. ScotWeave. http://www.scotweave.com/.
Martin Sherburn. 2007. *Geometric and mechanical modelling of textiles*. Ph.D. Dissertation. University of Nottingham.
Shinjiro Sueda, Garrett L Jones, David IW Levin, and Dinesh K Pai. 2011. Large-scale dynamic simulation of highly constrained strands. In *ACM SIGGRAPH 2011 papers*. 1–10.
Muhammad Umair, Yasir Nawab, Mumtaz Hasan Malik, and Khubab Shaker. 2015. Development and characterization of three-dimensional woven-shaped preforms and their associated composites. *Journal of Reinforced Plastics and Composites* 34, 24 (2015), 2018–2028.
Nobuyuki Umetani, Danny M Kaufman, Takeo Igarashi, and Eitan Grinspun. 2011. Sensitive couture for interactive garment modeling and editing. *ACM Trans. Graph.* 30, 4 (2011), 90.
Kiril Vidimce, Alexandre Kaspar, Ye Wang, and Wojciech Matusik. 2016. Foundry: Hierarchical material design for multi-material fabrication. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 563–574.
Pascal Volino, Nadia Magnenat-Thalmann, and Francois Faure. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. (2009).
Kui Wu, Xifeng Gao, Zachary Ferguson, Daniele Panozzo, and Cem Yuksel. 2018. Stitch meshing. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 130.
Kui Wu, Hannah Swan, and Cem Yuksel. 2019. Knittable stitch meshes. *ACM Transactions on Graphics (TOG)* 38, 1 (2019), 1–13.
Rundong Wu, Claire Harvey, Joy Xiaoji Zhang, Sean Kroszner, Brooks Hagan, and Steve Marschner. 2020. Automatic Structure Synthesis for 3D Woven Relief. *ACM Transcations on Graphics (TOG)* 39, 4.
Joanne Yip and Sun-Pui Ng. 2008. Study of three-dimensional spacer fabrics:: Physical and mechanical properties. *Journal of materials processing technology* 206, 1-3 (2008), 359–364.
Cem Yuksel, Jonathan M Kaldor, Doug L James, and Steve Marschner. 2012. Stitch meshes for modeling knitted clothing with yarn-level detail. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 37.
Shuang Zhao, Fujun Luan, and Kavita Bala. 2016. Fitting procedural yarn models for realistic cloth rendering. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 51.

A APPENDIX

A.1 Yarn Flipping in Weave Block

The yarn flipping operation as mentioned in Section 4.4 can be achieved using Algorithm 1. First, we sink the top yarn in the target pair to the bottom of all consecutive yarns of the same type in this stack, and float the bottom yarn similarly. After sinking and floating the two yarns, if they are neighboring in the stack, the flipping will be valid and we swap them. Otherwise, swapping the target pair will affect other pairs and the flipping is rejected. Figure 18 illustrates examples of valid and invalid flipping operations.

ALGORITHM 1: Flipping a pair of weft and warp yarns in a stack

Input: Stack `a`, Position of the top yarn in the target pair `top`, Position of the bottom yarn in the target pair `bot`
Output: New stack `a`

```
b = a
while top > 0 and sameType(b[top], b[top-1]) do
    swap(b[top], b[top-1])
    top--
end
while bot < len(b)-1 and sameType(b[bot], b[bot+1]) do
    swap(b[bot], b[bot+1])
    bot++
end
if top == bot+1 then
    swap(b[top], b[bot])
    a = b
end
```

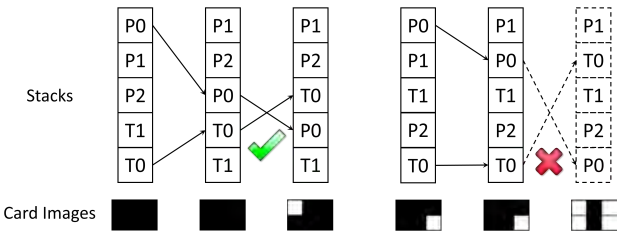


Fig. 18. **Flipping two yarns `'P0'` and `'T0'` in a stack.** If the flipping only affects one pixel on the card image, it will be accepted (left). Otherwise, the flipping will be rejected (right).

A.2 Algorithms for Block Operations

Following are the algorithms for computing the blocks for the operations mentioned in Section 4.3.

ALGORITHM 2: Attaching in x direction

Input: Block 1 {r1, c1, h1, t1, p1, A1} , Block2 {r2, c2, h2, t2, p2, A2}
Output: New Block {nr, nc, nh, nt, np, nA}

```
nr = r1 + r2
nc = c1
nh = max(h1, h2)
nt = concat(t1, t2)
np = p1
nA = EmptyArray(nr, nc, nh)
nA[:r1][:][:h1] = A1
nA[r1:][:][:h2] = Renumber(A2, sum(t1), 0)
```

ALGORITHM 3: Attaching in z direction

Input: Block 1 {r1, c1, h1, t1, p1, A1} , Block2 {r2, c2, h2, t2, p2, A2}
Output: New Block {nr, nc, nh, nt, np, nA}

```
nr = r1
nc = c1
nh = h1 + h2
nt = t1 + t2
np = p1 + p2
nA = EmptyArray(nr, nc, nh)
for i = 0, ..., nr-1 do
    for j = 0, ..., nc-1 do
        nA[i][j][:h1] = Renumber(A1[i][j], sum(t2[:i]),
        sum(p2[:j]))
        nA[i][h][h1:] = Renumber(A2[i][j],
        sum(t1[:i+1]), sum(p1[:j+1]))
    end
end
```

ALGORITHM 4: Stenciling

Input: N Blocks Bi= {ri, ci, hi, ti, pi, Ai} , i=0,...,N-1, x-by-y stencil image S
Output: New Block {nr, nc, nh, nt, np, nA}

```
nr = x
nc = y
nh = max(h0,...,h(N-1))
for i = 0,...,N-1 do
    {ri, ci, hi, ti, pi, Ai}  = Tile(Bi, 1+x/ri, 1+y/ci,
    1)
end
nt = t0[:x]
np = p0[:y]
nA = EmptyArray(nr, nc, nh)
for i = 0,...,x-1 do
    for j = 0,...,y-1 do
        nA[x][y][:h(S[x][y])] = A(S[x][y])[x][y]
    end
end
```

ALGORITHM 5: Tiling

Input: Block {r, c, h, t, p, A} , Number of tiles x, y, z
Output: New Block {nr, nc, nh, nt, np, nA}

```
nr = x*r
nc = y*c
nh = z*h
nt = repeat(z*t, x)
np = repeat(z*p, y)
nA = EmptyArray(nr, nc, nh)
for i = 0, ..., x-1 do
    for j = 0, ..., y-1 do
        for k = 0, ..., z-1 do
            for ii = 0, ..., r-1 do
                for jj = 0, ..., c-1 do
                    tOffset =
                    i*z*sum(t)+ii*(z-1)*sum(t[:ii])+k*t[ii]
                    pOffset =
                    j*z*sum(p)+jj*(z-1)*sum(p[:jj])+k*p[jj]
                    nA[i*r+ii][j*c+jj][k*h:k*h+h] =
                    Renumber(A[ii][jj], tOffset, pOffset)
                end
            end
        end
    end
end
Function Renumber(array, tOffset, pOffset)
    ret = array
    for Every element a in ret do
        if isWeft(a) then
            a += tOffset
        end
        if isWarp(a) then
            a += pOffset
        end
    end
    return ret
```

A.3 Details of Yarn Curve Generation

We use the key points generation for a weft yarn as an example. Consider a weft yarn `'Tx'` that lies in the i-th row of the block and appears in stacks `A[i][j], j=0,...,c-1`. For each stack in the row, we find the position `kj` of `'Tx'` in the stack, s.t. `A[i][j][kj] == 'Tx'`. Then we can calculate the positions of c key points for this yarn {`(dx*i, dy*j, dz*kj), j=0,...,c-1`}, where dx, dy and dz are the parameters controlling the spacing between yarns in the three dimensions. Note that we introduced a symbol `'E'` for empty spaces in Section 4.1. These empty spaces do not affect the card image generation or block assembly constraints, but only affect the initial yarn curves generated for rendering and simulation. The user can specify such empty spaces by disabling weft yarns in the user interface.

In our block representation, yarns in the same row or column can cross each other. Using two warp yarns `'P0'` and `'P1'` in the first column as an example, suppose their positions in the i-th row are `ki` and `li` respectively, i.e. `A[i][0][ki] == 'P0'` and `A[i][0][li] == 'P1'`. If `ki > li` and `k(i+1) < l(i+1)`, then this means that the relative position of these two yarns are flipping between row

i and row i+1, and there will be a crossing where the two yarn curves exactly intersect. Such an intersection between two yarn centerlines would cause the simulation to fail. Therefore, when we detect such a crossing, we will apply a subgrid displacement to the corresponding yarn curve segments, to route curves past each other and avoid the intersection. The amount of displacement for each warp yarn is different from any other warp in the same column, to ensure they are separated when they pass. Note that there can still be some overlap between the yarns, but such overlap is resolvable by collision energy during the simulation. Figure 19 shows an example of these sideways displacements.

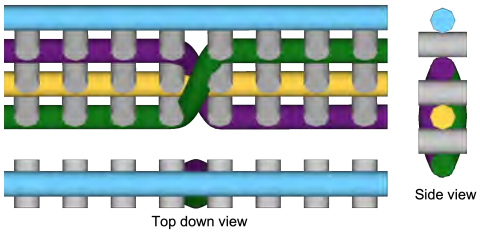


Fig. 19. **Warp yarns (shown in color) crossing each other in the same slice are displaced to avoid exact intersection.** Three different perspectives are included to show the displacement.

A.4 Discussion on Yarn Flipping Simulation

The proposed method in Section 5.2 for yarn flipping simulation has some problems during edge cases.

Edge Case: Small Yarn Segments. Suppose that yarn segment s_i is shorter than the radius of the yarns in question. When yarn segment s_j attempts to transition through the fabric, there will be collisions between s_j and neighboring segments s_{i-1}, s_{i+1}, etc. Transitions should be guaranteed to be smooth, not causing fights between collision forces and the transition spring force.

To address this issue, the method automatically generates all transition pairs around s_i and s_j if segments are too small. That is, if $2n$ is the number of nearby segments included in the set of transition pairs, then the list of pairs can be expressed as $S = (s_{i\pm k}, s_{j\pm \ell})$ where $k \in [0, n]$ and $\ell \in [0, n]$. Generating nearby pairs is equivalent to increasing the effective length of the transition pair segments s_i and s_j. Now as the spring force is applied to push the yarns through each other, collisions will not be triggered even under minor sliding conditions. The choice of n depends on the relative length of this effective length and the radius of the yarns. The effective/aggregate length should be at least twice the size of the radius of the thickest yarn, to ensure that there is enough space for the yarns to pass through each other.

Edge Case: Yarn Sliding. To fix the yarn sliding problem, a method to prevent other sections of the fabric from making the transition segments slide around is required. Our simulation model is integrated via a symplectic euler update of the following form:

$$v^{n+1} = v^n - \Delta t M^{-1} \nabla E$$

Here v^n is the velocity at the nth timestep, Δt is the timestep, M^{-1} is an inverse mass matrix, and ∇E is the energy gradient of the simulation. By setting elements of this inverse mass matrix to 0, some yarn segments have infinite mass and cannot move. All yarn segments that are not near a transition pair are frozen using inverse mass filtering to ensure that no sliding or separation occurs for the transition pairs in the pattern. After the transition pair has completed, the rest of the fabric is unfrozen and relaxation proceeds as normally.

Although the aforementioned approach can reliably handle many types of crossings, some pattern edits are challenging to do efficiently in this scheme. Two yarns that are parallel to each other would require a large number of transition pairs to exchange their positions. For a similar set of reasons, any edits that would require a large amount of material to move around will be possible, but may require scheduling multiple edit passes. We avoid these cases by restricting edits to only be between warp-weft yarn pairs; any required changes to the design that do not fall into this case can simply be done by revising the design and restarting the simulation.