Parsing Sewing Patterns into 3D Garments

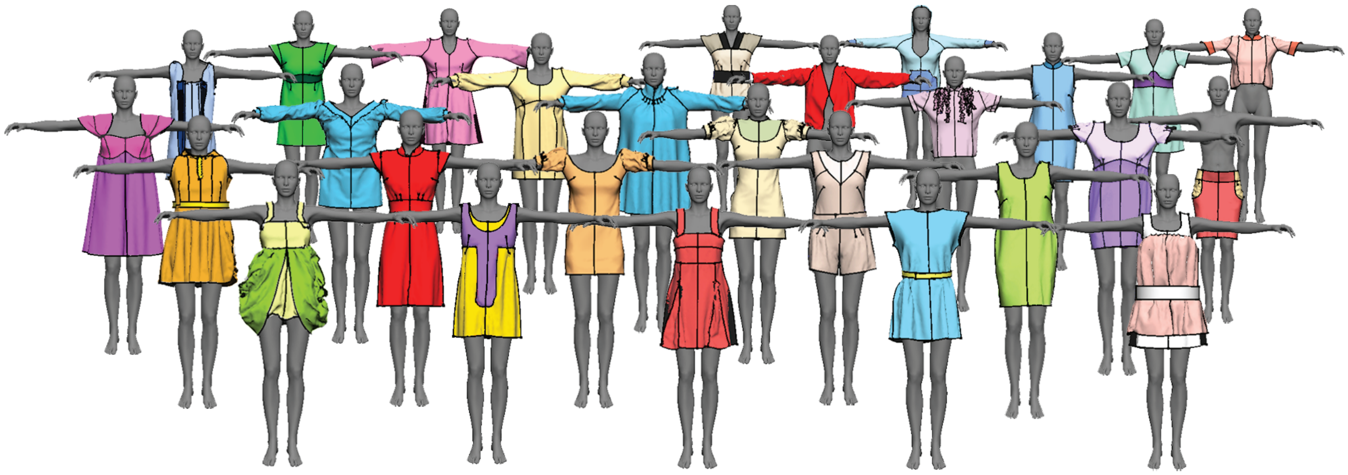


Figure 1: *Our parser automatically converted a diverse set of sewing patterns into 3D garment models for this small crowd of women.*

Abstract

We present techniques for automatically parsing existing sewing patterns and converting them into 3D garment models. Our parser takes a sewing pattern in PDF format as input and starts by extracting the set of panels and styling elements (e.g. darts, pleats and hemlines) contained in the pattern. It then applies a combination of machine learning and integer programming to infer how the panels must be stitched together to form the garment. Our system includes an interactive garment simulator that takes the parsed result and generates the corresponding 3D model. Our fully automatic approach correctly parses 68% of the sewing patterns in our collection. Most of the remaining patterns contain only a few errors that can be quickly corrected within the garment simulator. Finally we present two applications that take advantage of our collection of parsed sewing patterns. Our garment hybrids application lets users smoothly interpolate multiple garments in the 2D space of patterns. Our sketch-based search application allows users to navigate the pattern collection by drawing the shape of panels.

Keywords: Diagram parsing, garment modeling

1 Introduction

Sewing patterns describe the cutting, folding and stitching operations required to physically fabricate clothing. While websites such as burdastyle.de and voguepatterns.com provide ready access to thousands of such patterns online, the patterns themselves are terse and encode many of the sewing operations implicitly (e.g. how pieces of the garment are stitched together). To identify the complete sequence of operations required to construct a garment, skilled human tailors usually have to rely on their experience and understanding of the conventions of sewing patterns.

Garment designers for virtual characters in films and games do not exploit the rich collection of sewing patterns available online to generate 3D clothing. Instead, they manually create virtual clothing using special-purpose garment modeling and sculpting tools. This process requires significant expertise and is very time-consuming. Recent sketch-based garment design systems [**?**; **?**; **?**; **?**; Umetani et al. 2011] facilitate this process, but usually produce garment models that are simpler than real-world garments. Creating detailed 3D garment models remains a challenging task.

We present techniques for automatically parsing existing sewing patterns and converting them into 3D garment models. Given a sewing pattern in PDF format as input, our parser first extracts the panels or shaped pieces of cloth that form the garment. It then extracts styling elements such as darts, pleats and hemlines contained within the panels. The key step in parsing is to determine how the panels must be stitched together to form the garment. Our parser uses a combination of machine learning and integer programming to infer the stitching edge correspondences between panels. Our system includes an interactive garment simulator that uses these correspondences to generate a 3D model of the garment and drape it on a virtual mannequin. The simulator extends Sensitive Couture [Umetani et al. 2011] with a small set of features that allow it to support a larger variety of real-world input patterns.

Our fully automatic approach correctly parses 68% of the sewing patterns in our collection. Most of the remaining patterns contain only a few errors that can be quickly corrected within the garment simulator. Our system automatically generated all of the 3D garments in Figure 1 without any parsing errors.

Building a collection of parsed sewing patterns opens the door to myriad data-driven applications. To demonstrate this potential, we draw inspiration from recent work on the reuse, alteration, resizing and retargeting of garments [**?**; **?**; **?**; **?**; **?**; **?**] and present two applications that allow users to further explore the space of garment design. Our first application uses these parsed patterns to smoothly interpolate multiple garment panels in the 2D pattern space. Users can now rapidly create and explore multiple design variants via hybrids and combinations of exisiting garment designs. Our second application allows users to navigate available pattern collections by sketch-based search.

2 Previous Work

Parsing diagrams. People often use diagrams to communicate information. For example, sewing patterns are diagrams that describe the operations necessary to assemble a garment. Researchers have developed automatic parsers that can reconstruct the information contained in many different types of diagrams, including engineering drawings [**?**], cartographic road maps [**?**], sheet music [**?**], and data visualizations [**?**]. These parsers rely on image-processing methods and domain-specific knowledge to extract and interpret the

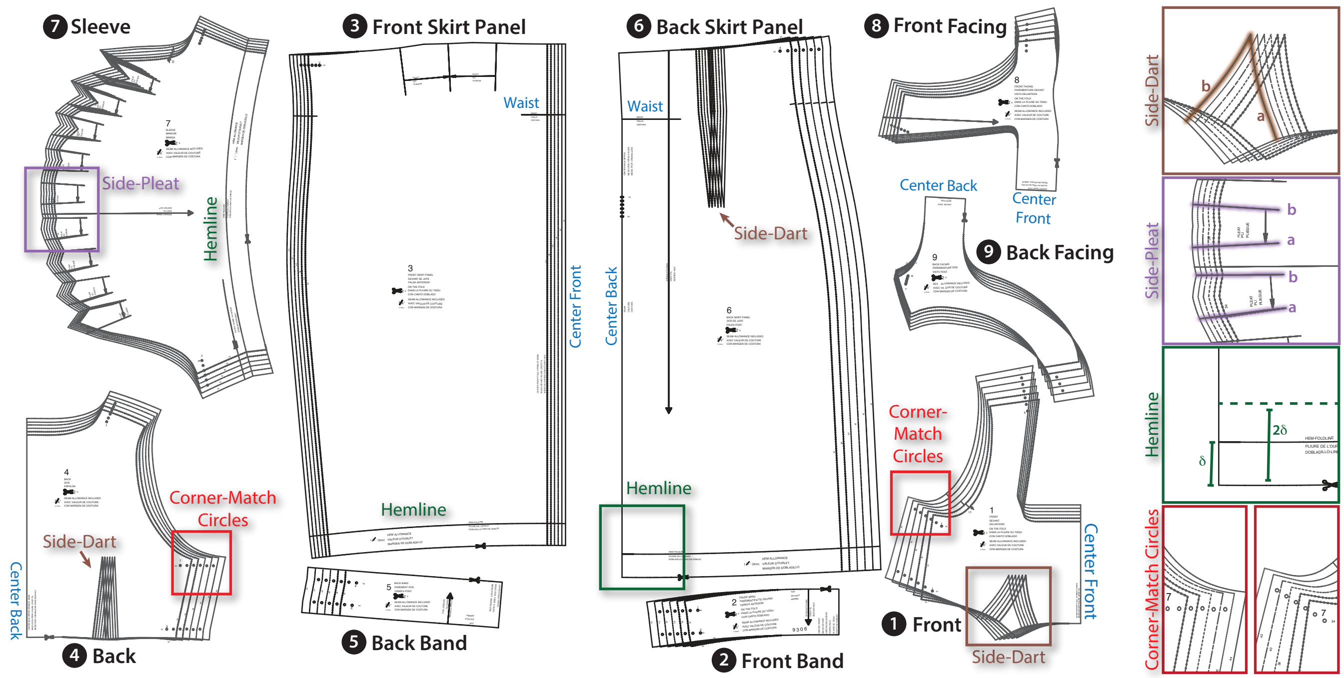


Figure 2: *A sewing pattern from our collection containing nine panels. Each panel contains multiple contour lines that represent different garment sizes and are drawn using different line styles (dotted, crosses, etc.) We have annotated some styling elements including darts (brown), pleats (purple), and hemlines (green) as well as placement labels (blue) and corner-match circles (red). The side-dart and side-pleat insets show how the edges marked* a *and* b *must be sewn together to properly form the styling element. The hemline inset shows how the panel must be folded at the line marked as a hemline stitched along an additional line (dotted) located at twice the distance* δ *from the nearby panel contour. The corner-match circles inset shows the number 7 circles from the Back and Front panels. These circles indicate matching panel corners. Readers can zoom into the PDF to see the text provided in the original pattern as well as the different line styles on panel contours.*

diagrams. To the best of our knowledge, our work is the first to apply a similar approach to parsing sewing patterns.

Sketch-based apparel design. Research in sketch-based interfaces considers the construction of an internal model representation from input strokes and annotations. In that sense, algorithms that parse diagrams share similarities to sketch-based apparel design. Turquin et al. [**?**; **?**] ask the user to oversketch a character's body with an intended garment's silhouette and infer the garment's geometry and drape. Robson et al. [**?**] further incorporate contextual knowledge of key factors that affect a garment's shape. Wang et al. [**?**] and Decaudin et al. [**?**] ask the user to sketch on the mannequin itself, dividing the body into extruded panels; the latter work then develops the panels into a design pattern of the kind we consider here.

There are also important differences between interactive sketch-based tools and our problem setting. First, most sketching interfaces oversketch a 3D mannequin, whereas we interpret the 2D line and text comprising a pattern. Second, sketching gives strokes an orientation and temporal ordering; it allows for gestures, or non-marking sketches. These additional data are not available in printed patterns. On the other hand, sketching interfaces must be interpreted online, constructing a model only from past strokes with no foresight; our patterns are interpreted in whole and offline, allowing for global analysis.

Computer-aided design and fitting of sewing patterns. Sewing patterns are the singular standard for specification of apparel designs in the fashion industry. For this reason, a considerable number of academic works and commercial software tools have been developed for computer-aided garment pattern design [**?**]; representative samples include ClothAssembler [**?**], Optitex PDS (Pattern Design Software), Marvelous Designer, and Pattern Works Int'l. Interpretation of sewing patterns and prediction of their drape is an important technology for developing a virtual fitting room [**?**; **?**; **?**; **?**], ideally one that accommodates virtual people in all shapes, sizes, and poses [**?**]. Igarashi and Hughes [**?**] presented an interactive tool for placing garment panels on a mannequin. In this work, we extend Sensitive Couture [Umetani et al. 2011] to assemble parsed patterns and automatically drape them on a mannequin.

3 Sewing Patterns

We purchased 50 sewing patterns from burdastyle.de. While many pattern collections are available online, we chose Burdastyle because its collection of over 8000 patterns is large, diverse, and inexpensive ($3 to $15 per pattern). We have also examined patterns from a number of sites and found that they all use standardized diagrammatic elements to help tailors understand the steps required to stitch and assemble the garment.

3.1 Diagrammatic Elements of Sewing Patterns

Figure 2 shows a sewing pattern for a dress that is composed of nine closed polygonal regions called **panels**. Some edges of the panels include multiple contour lines that indicate how the shape of the panel must change for a small range of standard clothing sizes. Each such contour line is drawn using a different line style (e.g. dotted, crosses etc.), with the largest size drawn as a solid line.

Each panel is labeled with a generic ***panel name*** (e.g. Back Skirt Panel, Sleeve, Front, etc.) that roughly describes the position of the panel on the body. Some panels may also include ***placement labels*** (e.g. Center Front, Center Back, Waist, etc.) on interior placement lines or on the exterior contour of the panel. These labels further specify the position of the panel with respect to the body and other panels. We have found that panel names and placement labels are consistent across the pattern collections we have examined.

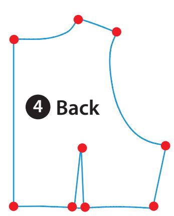
Patterns may also contain **styling elements**:

- ***Darts*** are triangular or diamond-shaped folds sewn into the fabric to fit the garment to the body. The two sides of the triangle or the diamond meet at an apex and must be stitched together to hold the fold in place. The width of a dart is the maximum distance between its two sides. We differentiate a *mid-panel-dart* which occurs within a panel from a *side-dart* which occurs along the panel contour. In Figure 2, panels 1, 4 and 6 contain side-darts.
- ***Pleats*** are formed by doubling fabric back on itself and securing it in place with a stitch. They are marked in the pattern by two nearly parallel lines, an arrow indicating the direction of the fold, and a "Pleat" label between the two lines. As with darts we differentiate *mid-panel-pleats* from *side-pleats* based the distance to the panel contour. In Figure 2 panels 3 and 7 contain side-pleats.
- ***Hemlines*** usually occur near the bottom contour of a panel and indicate that the fabric must be folded and sewn to the interior of the garment. In Figure 2 panels 3, 6 and 7 contain hemlines.

3.2 Assembling a Garment from a Sewing Pattern

The first step in tailoring a garment is to cut each panel from a piece of cloth. Sewing patterns exploit left-right symmetry and only include panels for the left side of the garment. Therefore tailors must duplicate and reflect each panel in the pattern to obtain the complete set of panels for the garment.

To assemble the garment, tailors sew the panels together along ***stitching edges***. These edges usually lie on the contour of the panel and exhibit low variation in curvature. For example, the Back panel of Figure 2 has six stitching edges along its contour and two more edges within a side-dart (see inset). Note that the bottom edge of the panel is counted as one stitching edge that is split by a side-dart. Each stitching edge either remains "free" or is stitched together with one or multiple other stitching edges usually belonging to different panels. We say that these attached stitching edges are in ***correspondence***.



Patterns usually include a sparse set of annotations we call ***corner-match circles*** that are designed to help tailors infer the correspondences between stitching edges. These numbered circles always appear at the intersection of two stitching edges and tailors must match the number across different panels to identify a correspondence between panel corners (Figure 2). While corner-match circles aid tailors in understanding how to sew together the garment, they only specify how panels join together at a few corners. Tailors must usually consider multiple corner-match circles as well the panel names and placement labels to determine the complete set of correspondences along stitching edges.

While most of the panels in a pattern form the main-body of the garment, many patterns also include a few decorative panels such as pockets, ruffles and belts. Since panel names are consistent across patterns, tailors can easily distinguish between ***main-body*** panels and ***decorative*** panels based on their names. Tailors often consider the main-body panels first and once they have understood how these are assembled they add in the decorative panels.

4 Overview

Converting a sewing pattern into a 3D garment model involves two components; a sewing pattern *parser* (Section 5) and a garment *simulator* (Section 6). Our work primarily focuses on the parser. It takes a sewing pattern in PDF format as input and starts by extracting the set of panels and styling elements contained in the pattern. It then identifies the stitching edges for each panel. Finally it applies a combination of machine learning and integer programming to infer the most likely correspondences between the stitching edges. Our garment simulator then uses these correspondences to generate and drape a 3D model of the garment. It extends the Sensitive Couture system of Umetani et al. [2011] with a small set of features that allow it to support a larger variety of real-world input patterns.

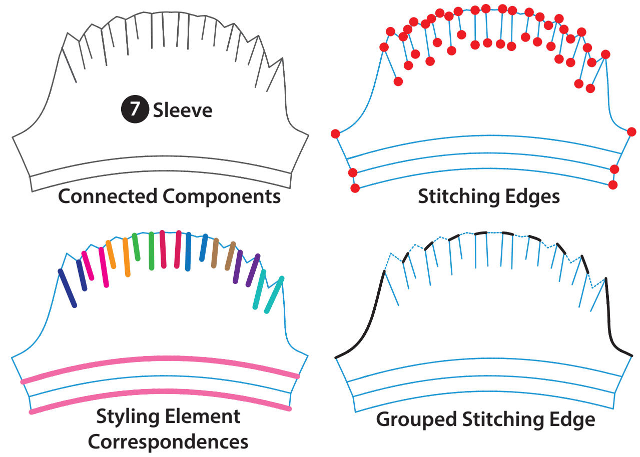


Figure 3: *Extracting the Sleeve panel. The connected component (top-left) and stitching edges (top-right) of the panel. The pleats and hemline stitching edge correspondences (bottom-left). Grouping the stitching edges adjacent to side-pleats.(bottom-right).*

5 Parser

Our parser includes two stages, a panel extractor and a correspondence identifier. It outputs a list of panels, styling elements and stitching edge correspondences that together describe how the panels must be stitched together to form the garment.

5.1 Extractor

Our input sewing patterns are vector graphics files in PDF format and are composed of two basic types of elements: line segments and text. The panel extractor is responsible for analyzing this collection of segments and text to identify the panels and styling elements along with their associated labels, placement labels, stitching edges and corner-match circles.

5.1.1 Extracting Panels

To identify the panels the extractor considers all solid line segments and groups them into connected components. In most cases each resulting component represents a single panel. For example Figure 3 (top-left) shows the Sleeve connected component for the pattern in Figure 2. In some instances, however, a component may represent a mid-panel styling element rather than a complete panel. Thus, for each component the extractor checks if it is fully enclosed by any other component and if so it groups them together.

The extractor then traces out the external contour of each panel at the largest clothing size. It starts with the line segment that is furthest away from the center of the panel bounding box and then steps along connected segments, while always choosing the outermost segment if the path branches. This tracing procedure ends when the extractor returns to the initial segment. The result is a closed contour loop.

The extractor next considers the set of line segments and text that lie within the contour of each panel. It identifies the largest text element within the panel as the panel name. The extractor differentiates between main-body panels and decorative panels based on the panel name. As a one-time pre-process we manually built a lookup table mapping the panel name to each of these two categories.

The extractor also collects all of the interior line segments and chains the connected segments into longer continuous lines. It then associates any remaining text elements with nearby interior lines if the distance between the text and line is within a small threshold. These labeled lines represent either interior placement lines, fold lines or hemlines. To identify the corner-match circles, the extractor finds interior line segments that form relatively small circles. It then treats nearby numerical text element as the corner-match label.

The extractor splits the contour and interior lines into stitching edges. As we have noted stitching edges are relatively smooth and do not contain sharp corners. Therefore the extractor breaks these lines into stitching edges whenever the angle between consecutive segments is larger than a threshold (we use 25°). In Figure 3 (top-right), red circles indicate endpoints of stitching edges.

5.1.2 Extracting Styling Elements

To identify pleats the extractor looks for a pair of nearly parallel interior lines that contain the text label “Pleat” between them. It differentiates mid-panel-pleats from side-pleats based on whether or not the pleat lines touch the panel contour. To identify darts the extractor looks for loops in the set of lines comprising the panel. It marks loops that only include interior lines as mid-panel-darts and loops that include part of the panel contour as side-darts. For example, the dart in the Front panel of Figure 2 contains part of the panel contour and is therefore classified as a side-dart.

Styling elements such as darts, pleats and hemlines directly encode their corresponding stitching edges and the extractor immediately marks these correspondences. In a dart, for example, the two interior lines that meet at the apex form two corresponding stitching edges that must be sewn together. In a pleat the two parallel stitching edges correspond to one another. For a hemline we first form an additional stitching edge located at twice the distance δ between the hemline and contour. This additional stitching edge corresponds to the parallel contour stitching edge. We color-code pleat and hemline correspondences for the Sleeve panel in Figure 3 (bottom-left).

Note that when side-darts or side-pleats are sewn into a garment they eliminate a portion of the contour stitching edge. Therefore we group the adjacent contour stitching edges on either side of the dart or pleat and treat them as a single stitching edge. In Figure 3 (bottom-right), the top of the Sleeve includes 9 side-pleats and we group the adjacent edges (black) into a single stitching edge.

5.2 Correspondence Identifier

The correspondence identifier is responsible for determining how the panels should be stitched together to form the garment. It infers the most likely correspondences between all of the remaining stitching edges in the garment using a combination of machine learning and integer programming.

To form a complete garment it is essential to stitch together the main-body panels. The decorative panels serve to further embellish the garment but are less important. Therefore, our approach is to first process the main-body panels (step 1) and then handle the decorative panels (step 2).

In each step we build a table of probabilities $\mathbf{P}$ where each entry $P_{i,j}$ captures the likelihood of a correspondence between a pair of stitching edges (e_i, e_j) (Figure 4). In the first step we consider all pairs of stitching edges (e_i, e_j) where both e_i and e_j belong to main-body panels. In the second step we focus on stitching edge pairs where the first edge e_i belongs to a decorative panel and the second edge e_j belongs to a main-body panel. We augment both of these tables with an extra column to capture the probability $P_{i,n+1}$ that stitching edge e_i remains “free” and is not attached to any other edge. We describe how we compute these probabilities in Section 5.2.2.

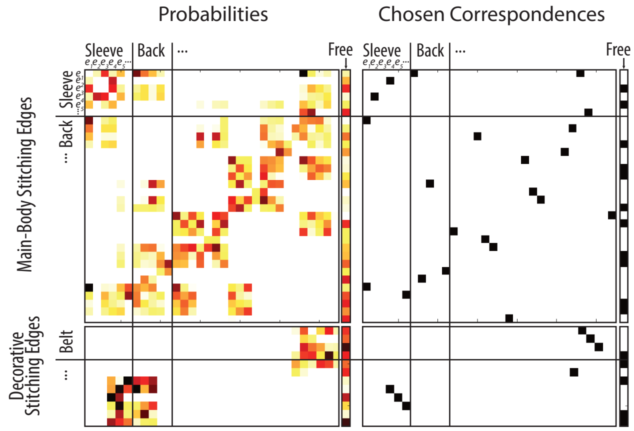


Figure 4: *Probability tables for the main-body panel stitching edges (top-left) and decorative panel stitching edges (bottom-left). Red indicates higher probability of correspondence. Optimal stitching edge correspondences chosen by our integer program (right).*

This probability table allows us to find the most likely stitching correspondences. Let $\omega : [1 \ldots n] \rightarrow [1 \ldots n+1]$ map the index i of each edge e_i to a corresponding edge index $\omega(i)$, with $\omega(i) = n+1$ indicating a free (unmatched) edge. We further restrict ω, in a way we will soon make precise, to respect a few important properties of garments. We seek the map ω that maximizes the joint probability

$$\prod_{i=1}^{n} P_{i,\omega(i)} .$$

Using the monotonicity of the logarithm, we equivalently seek to maximize $\sum_{i=1}^{n} \log P_{i,\omega(i)}$. We encode ω by the $n \times n+1$ indicator matrix $\mathbf{X}$, with unit entries encoding correspondences ($\forall i,\ x_{i,\omega(i)} = 1$), and remaining entries zero. This gives rise to the integer programming problem

$$\underset{\mathbf{x}}{\operatorname{argmax}} \sum_{i=1}^{n} \sum_{j=1}^{n+1} x_{i,j} \log P_{i,j} \quad (1)$$

subject to

$$\textstyle\sum_{j=1}^{n+1} x_{i,j} = 1 \quad \text{one corresp. per row } i \quad (2)$$

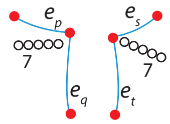
$$\textstyle\sum_{i=1}^{n} x_{ii} = 0 \quad \text{no self corresp.} \quad (3)$$

$$\forall i, j,\ x_{ij} - x_{ji} = 0 \quad \text{mutual corresp. } (\omega \text{ is symmetric}) \quad (4)$$

$$x_{ps} + x_{pt} + x_{qs} + x_{qt} = 1 \quad \text{one corresp. per circle} \quad (5)$$

Typically each stitching edge either remains free or corresponds to exactly one other stitching edge. Constraint (2) enforces this requirement; we will soon modify this constraint to handle multiple stitching edge correspondences. Constraint (4) prevents stitching edges from corresponding with themselves. Constraint (6) forces mutual correspondence between stitching edges.

Constraint (5) considers stitching edges that are adjacent to corner-match circles. These circles indicate matching corners on two different panels and limit the potential correspondences to four stitching edge pairs. As shown in the inset, edges e_p and e_q are adjacent to the first circle and edges e_s and e_t are adjacent to the second circle. Constraint (5) ensures that exactly one of the four potential correspondences between these edges is active.



We solve for $\mathbf{X}$ using a built-in MATLAB integer programming routine[1], which implements a linear program solver using branch-and-bound [Wolsey 2000]. Figure 4 shows the optimal set of stitching edge correspondences for our example.

5.2.1 Handling Multiple Correspondences

Some panels contain stitching edges that correspond to more than one other stitching edge. For example, one edge of a Sleeve often attaches to both a Front panel and a Back panel (Figure 2). We have manually examined our collection of patterns to identify a small set of panels that contain such multi-correspondence stitching edges. In the panel extractor we mark edges that belong to these panels as potential multi-correspondence edges.

Another example occurs when the garment contains multiple layers stitched at the same edge. In this case a corner-match circle with the same number appears on more than two panels. We mark edges adjacent to these corner-match circles as potential multi-correspondence edges.

Finally, for each potential multi-correspondence edge e_k we replace Constraint (2) with

$$1 \le \sum_{j=1}^{n+1} x_{kj} \le m \quad \text{up to } m \text{ corresp. in row } k\ .$$

It may seem at first sight that this constraint allows an edge to be simultaneously marked as free (col. $n+1$) and stitched (another column). It can be shown that such a simultaneous marking is never optimal for the objective (**??**).

5.2.2 Correspondence Probabilities

To compute the probability that two stitching edges correspond we consider geometric information about the edges (e.g. edge length, curvature, etc.) as well as panel-level information (e.g. the name of the panel the stitching edge belongs to, nearby placement labels, etc.). We have found that the panel-level information is crucial and geometric information alone is usually not enough.

For example in Figure 2, the long edge of panel 5 is geometrically similar to the top and bottom edges of panels 2, 3 and 6, as well as the bottom edge of panel 4. However, we can eliminate some of these possibilities based on the panel names. Analyzing a set of assembled patterns we find that a Back Band never attaches to a Front or a Front Skirt. Thus, we can set the correspondence probability between any Back Band edge and Front or Front Skirt edge to zero. We extend this idea to multiple features of the stitching edges.

Our approach is to analyze a training set of assembled patterns, for which correspondences are given, summarizing the analysis in a symmetric *panel-panel table* (Figure 5). For each pair of panel names (allowing for self-pairing, as in a sleeve), we (a) identify all stitching edge correspondences that attach the two panels; (b) if there are no correspondences, we mark the cell as empty, otherwise, we store feature histograms in this cell. We consider two types of features: *edge features* capture properties of each edge (e.g. length, curvature, placement labels, etc.), while *match features* capture how well the corresponding edges match with one another (e.g. length difference, curvature difference). A complete list of features is detailed below.

To compute the correspondence probability $P_{i,j}$ for a test pair of stitching edges e_i and e_j we first look up the names of the panels they belong to. If the corresponding cell in the panel-panel table is empty we set $P_{i,j}$ to zero. Otherwise $P_{i,j} = P(C_{i,j} = 1|\mathbf{F}_{i,j})$ where $C_{i,j}$ is a binary variable that equals 1 when the two edges correspond and $\mathbf{F}_{i,j} = [F^1_{i,j}, ..., F^n_{i,j}]$ is a vector of all features in the panel-panel cell for the edge pair. We use naive Bayes to compute

[1] bintprog(): http://www.mathworks.com/help/optim/ug/binary-integer-programming-algorithms.html

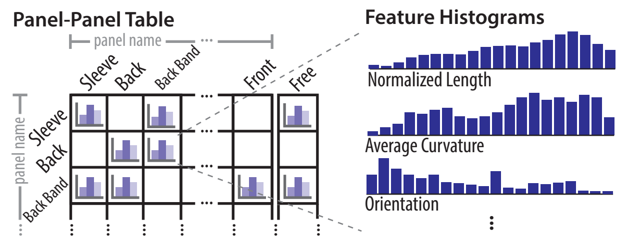


Figure 5: *Panel-Panel Table. For each pair of panel names, we identify all stitching edge correspondences that attach these two panels and compute a set of feature histograms for them. Empty cells indicate that the panels never attach to one another.*

$$P(C_{i,j} = 1|\mathbf{F}_{i,j}) = \frac{P(\mathbf{F}_{i,j}|C_{i,j}=1)P(C_{i,j}=1)}{P(\mathbf{F}_{i,j}|C_{i,j}=1)+P(\mathbf{F}_{i,j}|C_{i,j}=0)} \quad (6)$$

$$\text{where} \quad P(\mathbf{F}_{i,j}|C_{i,j}) = \prod_k P(F^k_{i,j}|C_{i,j}). \quad (7)$$

Each $P(F^k_{i,j}|C_{i,j} = 1)$ is directly given by the histogram in the panel-panel cell and we multiply them together to form the likelihood $P(\mathbf{F}_{i,j}|C_{i,j} = 1)$. We compute $P(F^k_{i,j}|C_{i,j} = 0)$ by first aggregating the histograms across all other cells in the panel-panel table and then form $P(\mathbf{F}_{i,j}|C_{i,j} = 0)$ as the product of these feature probabilities. We treat the prior probability $P(C_{i,j} = 1)$ as a constant for all pairs of stitching edges and can thus neglect the term in Equation 7.

Computing Edge and Match Features. We compute a number of features for each pair of stitching edges. Several of these features rely on panel-level geometric information. So as a pre-process, we compute the axis-aligned bounding box and the up orientation of each panel. We use styling elements and placement labels to compute the up orientation as follows. Since hemlines occur near the bottom contour of a panel (Section 3) if a panel contains a hemline we set the up vector perpendicular to it. Similarly the Top placement label occurs along the top contour of a panel and we set the up vector perpendicular to it. In the absence of such labels we set the up vector to the longest edge of the bounding box. We then compute the following edge features.

- ***Normalized length.*** We compute the length of the stitching edge normalized by the length of the diagonal of the panel bounding box. This feature captures the length of the stitching edge compared to the size of its panel.
- ***Average curvature.*** Each stitching edge is a polyline. We first fit a degree 5 polynomial curve to the polyline and then compute it's curvature at constant intervals. We average together these curvatures to form the final feature.
- ***Orientation.*** To capture the orientation of each stitching edge we compute the dot product of the vector connecting the endpoints of the stitching edge with the up vector.
- ***Placement label*** We determine if the stitching edge has a placement label associated with it. We set the feature to Empty if the stitching edge is not labeled and to the label name otherwise.
- ***Styling element.*** If the stitching edge is adjacent to a side-dart, side-pleat or near a hemline we set the feature to the name of the styling element. Otherwise we set the feature to Empty.

To obtain the match features for a pair of stitching edges we simply compute the L_2 distance between their normalized length, average curvature and orientation features. For the placement label and styling element features we build a binary feature that is set to 1 if the corresponding edge features have the same value.

6 Simulator

Once we have parsed a sewing pattern we use a garment simulator to generate the corresponding 3D model and drape it on a virtual mannequin. Work on dynamic, physical simulation of cloth spans over two decades [?; ?; ?; ?; ?; ?; ?]. We build on the Sensitive Couture interactive garment modeler [Umetani et al. 2011]. Sensitive Couture provides synchronized, interactive, bidirectional creation and editing of 2D clothing patterns and their corresponding physically simulated 3D garment. This interface is a natural front-end for our parser since it not only provides a 3D garment model but also enables users to immediately customize the parsed patterns via manipulation and editing with direct feedback on changes to the corresponding 3D physical drape.

To support a large variety of real-world input sewing patterns we have extended Sensitive Couture with a small set of additional features:

Positioning panels. To successfully drape a garment in Sensitive Couture, it is critical to provide good initial seed positions in 3D for all garment panels. In the original Sensitive Couture users had to manually specify the seed position. However, in our pattern collection, each panel name roughly describes the position of the panel with respect to the body (e.g. all Sleeve panels must be draped around arms). As a one-time pre-process we manually built a lookup table mapping main-body panel names to a rough 3D position on the virtual mannequin. We have modified Sensitive Couture to use this table to assign the seed position for main-body panels. We also rotate the main-body panels so that they point outwards with respect to the mannequin. Finally, we position decorative panels at a small offset from the main-body panels they attach to.

Interior stitching edges. The original Sensitive Couture only allowed stitching edges to occur along panel contours. However, styling elements like darts, pleats, and hemlines require interior stitching edges. To properly handle such interior edges we have modified Sensitive Couture to locally remesh the underlying simulation mesh around them [?]. This modification is crucial to obtain proper folding around the styling elements.

Sequential draping. Most garments are comprised of multiple layers (e.g. a Ruffle panel attaches on top of a Front main-body panel). Tailors generally drape the most interior layers first. We modified Sensitive Couture to allow draping in layers so that the most interior layers are draped first and then frozen in place before adding the next layer. This feature improves convergence speeds of the simulation and reduces the overhead of collision-processing.

7 Results

Figure 6 shows the parsing results as well as the simulated garments for seven example patterns from the set of 50 patterns we purchased from burdastyle.de. Pattern 9306 is the garment we generate for the pattern shown in Figure 2. Most of these patterns (6022, 6023, 6045, 6060, 9306) use styling elements. For example pattern 6022 includes darts and mid-panel pleats to fit the garment close to the body. Five of the patterns use decorative panels such as belts (6022, 6045), pockets (6028), multiple dress layers (6023) or additional pleat panels that are inserted between the main dress panels (6008). Figure 1 shows a variety of additional garments models generated by our system.

To evaluate our system, we processed all 50 patterns in our dataset. This collection is largely comprised of women's dresses but also includes some tops (e.g. blouses, sweatshirts, etc.) and trousers. Each pattern contains 10 to 30 panels that are attached to one another by an average of 33 (std: 17) stitching edge correspondence. Parsing the patterns is relatively fast: our MATLAB implementation requires 5-10 seconds for pattern extraction and a few seconds for correspondence identification. Our simulator requires about 15 seconds to drape the 3D garment model and fully converge.

To test the parser we first built ground-truth data by manually annotating all of the stitching edge correspondences for all 50 patterns. Using a leave-one-out cross validation we found that for 68% of the patterns (34 out of the 50) our parser correctly identified all stitching edge correspondences. For the other 32%, our parser incorrectly marked an average of 4.3 (std: 2.7) correspondences. However, for two outlier patterns (a flamenco dress and summer dress with many differently shaped layers) our parser incorrectly identified most of the stitching edge correspondences. Note that all results shown in Figures 1 and 6 were parsed correctly, and did not require any manual correction.

Aggregating across all patterns our parser correctly identified 87% of the stitching edge correspondences. Corner-match circles are the most informative indicators of a correspondence. For each pair of corner-match circles with the same label, exactly one pair of adjacent edges correspond. To determine how much corner-match circles contribute to the learning, we manually annotated correspondences indicated by corner-match circles in our ground truth data. We then found that our parser correctly marked 97% of these correspondences. These numbers confirm that corner-match circles are very helpful in determining how the panels attach to one another. However, patterns are only sparsely annotated with such circles. Only 27% of all correspondences in our dataset were indicated with corner-match circles.

While not perfect, the correspondences identified by our parser provide the user with a good starting point. In most cases, the user can simply load the pattern into our simulator and drape the garment to produce a 3D model. If the parser generated incorrect correspondences, they can use the interactive tools in Sensitive Couture to quickly correct the problem. For example, the user can select two stitching edges to add or remove a correspondence.

8 Applications

Users can take advantage of our collection of parsed sewing patterns to explore the space of garment designs. We have developed two applications that support such exploration. Our garment hybrids application lets users smoothly interpolate multiple garments in the 2D space of patterns. Our sketch-based search application allows users to navigate the pattern collection by drawing panels.

8.1 Garment Hybrids

We have developed an algorithm that allows users to smoothly interpolate multiple sewing patterns and thereby create garment hybrids. Users must choose two input sewing patterns and can either interpolate all the panels or select individual panels to interpolate. Our interpolation algorithm works in the space of the 2D patterns to ensure that each intermediate result maintains a valid garment design. Interpolating the 3D shape of each garment would require users to manually specify correspondence information and would likely violate constraints imposed by the panels, styling elements and stitching correspondences.

To interpolate between two garments we identify all of the panels that they have in common based on the panel names. We then interpolate the contour stitching edges between each pair of panels and

Figure 6: *Parsing and simulation results for 7 sewing patterns. For each pattern we color-code the stitching edge correspondences identified by our parser. We show three views of the 3D garment model draped on a mannequin using our simulator. For comparison we include a photograph of the garment from burdastyle.de. Note that our parser correctly identified all of the stitching edge correspondences for the garments shown in this Figure.*

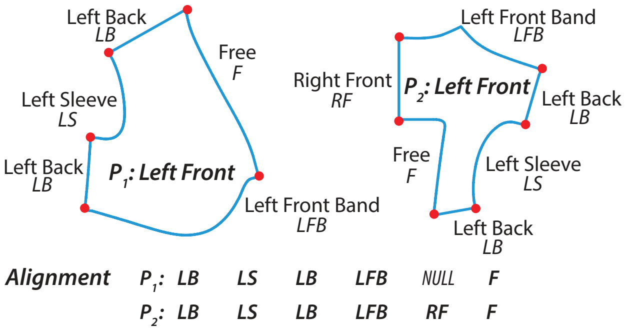


Figure 7: *Aligning two panels. We first identify the attachment labels for the two panels and then compute the best alignment between these labels using string edit distance.*

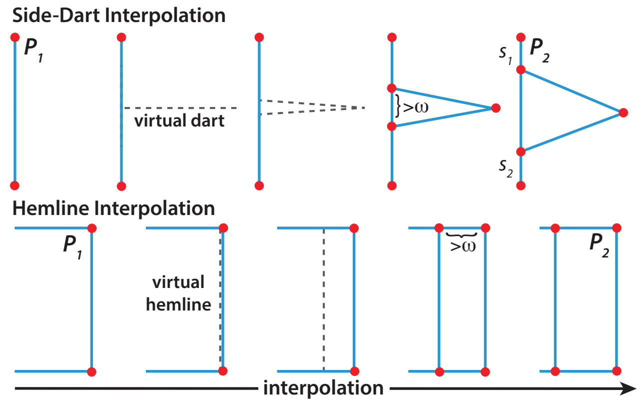


Figure 8: *Interpolation of side-dart (top) and hemline (bottom).*

then interpolate the styling elements within the panels.

8.1.1 Interpolating Panel Contour Stitching Edges

Stitching edges on panel contours serve to attach the panel to surrounding panels. When interpolating between two garments it is essential to maintain these stitching edge correspondences as much as possible to ensure proper connectivity of the interpolated result.

Given two panels P_1 and P_2 with the same name, but from different patterns, we start by computing an alignment between their contour stitching edges. This alignment ensures that we interpolate between stitching edges that connect to surrounding panels in similar ways. To create the alignment, we label each stitching edge of both panels with the name of the panel it is attached to (Figure 7). If the edge does not participate in a correspondence we label it as Free. Note that while many of attachment labels for P_1 and P_2 are identical the set of labels is not exactly the same; P_2 has a Right Front label that is not present in P_1. Such similarity in the set of adjacent panels is typical for panels that have the same name.

The contour attachment labels form a circular list, where the ordering is determined by connectivity of the edges. In Figure 7, the list for P_1 is Left Back, Left Sleeve, Left Back and Left Front Band, Free and the list for P_2 is Left Front Band, Right Front, Free, Left Back, Left Sleeve and Left Back. We use string edit distance to compute the the optimal alignment between these lists. Since the lists are circular, we compute the alignment for each circular offset of the second list and choose the best one.

Unlike the standard string edit distance algorithm, we do not allow for substitutions because such modifications would change the connectivity of the interpolated panel and could generate an invalid garment. We do allow insertion of Null labels into either string for a small constant cost. Null insertion corresponds to adding a zero length stitching edge to the panel. In Figure 7 we find that the best alignment for P_1 and P_2 occurs at offset 4 and adds a null label to the stitching edge label list for P_1.

After aligning the stitching edge labels we translate the panels so that their centers of mass are aligned. We also rotate the panels so that stitching edges with the same attachment label are as close to one another as possible. Finally, we parameterize the aligned stitching edges by arc length and linearly interpolate them.

8.1.2 Interpolating Styling Elements

Our interpolation algorithm supports adding or removing styling elements such as darts, pleats and hemlines. These elements are often essential to the look and fit of the garment and are therefore important to handle during interpolation.

Consider the case of interpolating between a panel P_1 that does not contain a dart and P_2 that contains a side-dart. After aligning the contour stitching edges of the panels we compute a proportional offset distance in P_2 between one endpoint of the stitching edge and the dart. For the example in Figure 8 (top) this distance corresponds to $s_1/(s_1+s_2)$, where s_1 and s_2 are the segments of the stitching edge adjacent to the dart. We then insert a corresponding virtual side-dart at the same proportional distance in P_2. Once we have aligned the two stitching edges and their respective side-darts in this manner we linearly interpolate between them. However, darts that are smaller than a minimum width, are unlikely to occur in sewing patterns. Therefore, during the interpolation we convert the virtual side-dart into a true side-dart only when its width is greater than a minimum width threshold ω. We experimentally set ω by examining our pattern collection and finding the minimum width dart within it. Removing a side-dart is equivalent to flipping the panel that initially contains the side-dart in this procedure.

If both panels contain a dart on the same aligned stitching edge we consider the proportional offset distance to each one. If these distance are similar then we interpolate the respective stitching edges of the darts. If the offset distance differ significantly we remove the first dart and insert the second. We use analogous procedures to add/remove/interpolate side-pleats.

For mid-panel-darts and pleats we use a similar procedure, but instead of computing the offset distance proportional to a single stitching edge endpoint, we compute the offset distances to the three closest stitching edge endpoints that share the same attachment labels. We then linearly interpolate these offset distances to generate the intermediate position of the dart or pleat. In the cases we have tried the intermediate dart or pleat always stayed within the panel contour. However our algorithm does not guarantee this property.

Finally, consider the case of adding a hemline (Figure 8 (bottom)). We first identify the stitching edge of P_2 that runs parallel to the hemline and find the corresponding stitching edge in P_1. We then insert a virtual hemline into P_1 that is initially co-located with this stitching edge. We then interpolate the relative distance between the virtual hemline in P_1 and the hemline in P_2. As with darts, we consider a minimum hemline width threshold before adding it to the intermediate pattern.

8.2 Hybrid Results

Figure **??** shows several garment hybrids created with our application. In the top row the user interpolated complete garments (all panels). In the transition between 9306 and 7951 note how the neckline smoothly changes while the pleats in the Sleeve and Skirt panels gradually disappear. Interpolating from 7951 to 6047, introduce new side-darts to the Front and Skirt panels. These darts fit the garment closer to the body without using pleats as were originally used

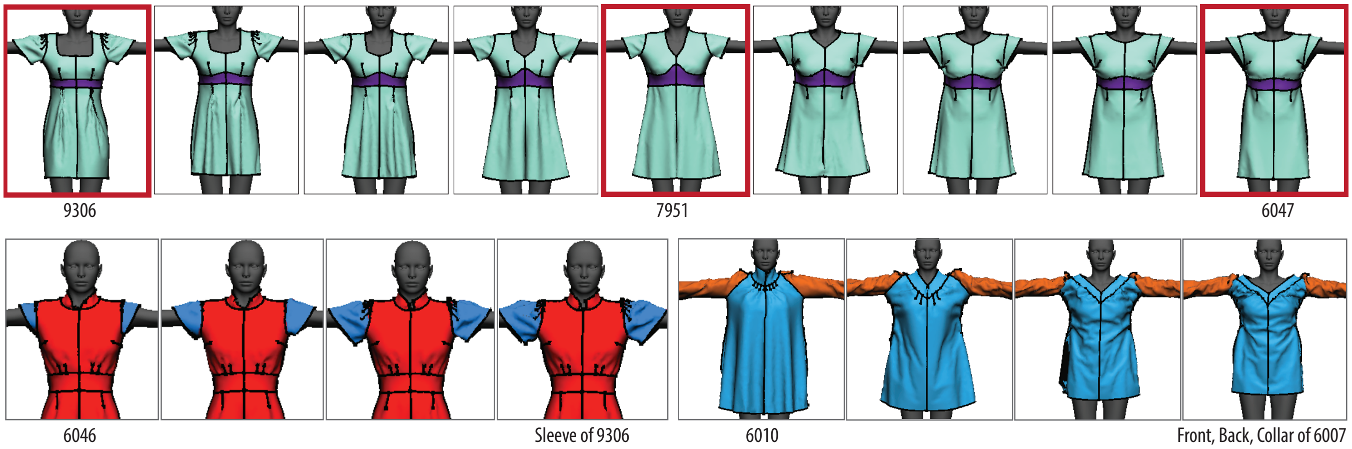


Figure 9: *Hybrid results. (top) A user interpolates between two pairs of dresses, first going from 9306 to 7951 and then from 7951 to 6047. (bottom) A user only interpolates the Sleeve panel of 6046 and the Collar, Front and Back panels of 6010.*

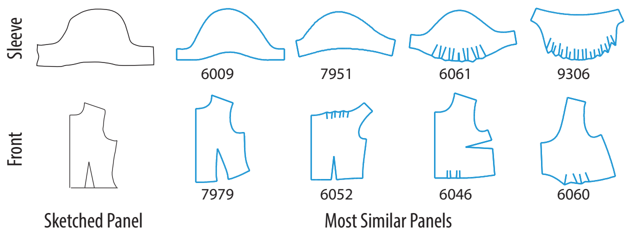


Figure 10: *(left) A user sketches a Sleeve (top) and a Front panel (bottom). (right) After specifying target panel name, our system retrieves other panels that match the target name ranks them by similarity to the sketch. The number below each retrieved panel indicated the pattern it belongs to.*

in 9306. The neckline and sleeves also change significantly in shape in the intermediate designs.

In the bottom row, the user specifies subsets of panels to interpolate. On the left side, the user interpolates the sleeves marked in blue. Note how the number of pleats in the sleeves increase. On the right, the user interpolates the Front, Back and Collar panels. The intermediate designs introduce many new pleats at the side of the garment and reduce the size of the Front and Back panels to make the garment tighter and shorter. In addition the Collar opens up significantly.

8.3 Search and Navigation of Pattern Collections

The look and style of a 3D garment is largely dependent on the shape of its panels and styling elements. Yet, online sewing pattern collections today force users to navigate the collection by scrolling through lists of pattern names. To help users better navigate the space of garment designs we have developed a sketch-based search application.

With our application users sketch the shape of a panel, and specify a target panel name. Figure **??** (left) shows two examples of panels sketched by a user. Our search algorithm first retrieves all the panels that match the target panel name and then uses the shape context algorithm of Belongie et al. [**?**] to rank each panel by similarity to the sketch. Each retrieved panel also provides a link back to the pattern it belongs to in the form of a pattern number. As shown in Figure **??** (right) users can easily find all the patterns that contain a particular style of Sleeve or Front panel. Inspecting the corresponding patterns and 3D drapes allows users to see how similarly shaped panels can affect the look of different garments.

9 Conclusion and Future Work

Sewing patterns designed for human tailors are readily available online. We have presented techniques for automatically parsing such patterns into 3D garment models. Our approach significantly reduces the time, effort and expertise required to create detailed clothing models for virtual characters. We also demonstrate two applications that use our collection of parsed patterns to help users explore the space of garment designs. We believe that automated parsing of sewing patterns can enable many more such data-driven applications.

Data-driven resizing. While sewing patterns do contain panel contours for a range of garment sizes our parser only extracts the panels for the largest size. Extending the parser to extract the entire size range would enable accurate *grading* of virtual garments and could serve as ground-truth data for recent algorithms for automatically resizing 3D garment models [**?**].

Suggestive garment design tools. Computer-aided tools for designing sewing patterns (e.g. Optitex PDS, Marvelous Designer, etc.) assume that users have enough garment design knowledge to know how to shape panels and where to place styling elements. Using a database of parsed patterns it should be possible for the system itself to suggest where styling elements might be placed even as the users starts creating panels. Similarly the system could suggest the most likely stitching edge correspondences for each panel when designing a new garment. Such automatic suggestions could further enable inexperienced designers to create garment designs.

Precomputed drape. Consistent reuse of common panel types over large collections of garments makes pre-computation and learning of corresponding 3D physical drapes another interesting avenue of future exploration. With a large database of example drapes, precomputed drape shapes will allow beginning users to quickly mix and match constituent pattern parts with instant feedback.