A geometric approach to robotic laundry folding

Stephen Miller, Jur van den Berg, Mario Fritz, Trevor Darrell, Ken Goldberg and Pieter Abbeel

The International Journal of Robotics Research published online 20 December 2011
DOI: 10.1177/0278364911430417

The International Journal of Robotics Research
0(0) 1–19

DOI: 10.1177/0278364911430417
ijr.sagepub.com

A geometric approach to robotic laundry folding

Stephen Miller[1], Jur van den Berg[2], Mario Fritz[3], Trevor Darrell[1], Ken Goldberg[1] and Pieter Abbeel[1]

Abstract

We consider the problem of autonomous robotic laundry folding, and propose a solution to the perception and manipulation challenges inherent to the task. At the core of our approach is a quasi-static cloth model which allows us to neglect the complex dynamics of cloth under significant parts of the state space, allowing us to reason instead in terms of simple geometry. We present an algorithm which, given a 2D cloth polygon and a desired sequence of folds, outputs a motion plan for executing the corresponding manipulations, deemed g-folds*, on a minimal number of robot grippers. We define parametrized fold sequences for four clothing categories: towels, pants, short-sleeved shirts, and long-sleeved shirts, each represented as polygons. We then devise a model-based optimization approach for visually inferring the class and pose of a spread-out or folded clothing article from a single image, such that the resulting polygon provides a parse suitable for these folding primitives. We test the manipulation and perception tasks individually, and combine them to implement an autonomous folding system on the Willow Garage PR2. This enables the PR2 to identify a clothing article spread out on a table, execute the computed folding sequence, and visually track its progress over successive folds.*

Keywords

Personal robotics, cloth, manipulation, perception

1. Introduction

With the 20th century advent of personal computers, the dream of the future was one of convenience and autonomy: of intelligent machines automating the monotonous rotes of daily life.

Few tedious tasks are as universal to the human experience as household chores. No utopian future would be complete, then, without household robots relieving humans of these tasks: doing the dishes, sweeping the floors, setting the table, and doing the laundry. In this paper, we explore the latter challenge. Washing machines and dryers have automated much of the process, but one clear bottleneck remains: autonomous laundry folding.

While advances continue to be made in the field of household robotics, this vision has proven particularly difficult to realize. This is largely due to the vast state space in which such a robot is expected to operate. In addition to the well-established problems of perception and manipulation in unconstrained environments, the task of laundry folding poses a unique challenge as not only the environment, but the very object which must be manipulated, is itself highly complex. Cloth is non-rigid, flexible, and deformable, with an infinite-dimensional configuration space. It may be found in an innumerable variety of poses, rendering the perceptual tasks of classification and pose-estimation extremely difficult. Furthermore, the dynamics of cloth are difficult to capture in even the most sophisticated simulators, posing great challenges to the manipulation planning task.

We do not, in this work, intend to provide a brute-force mechanism for planning under such complexity. Rather, we aim to simplify the problem by carving out a particular subset of cloth configurations which, under a number of governing assumptions, may be represented by few parameters while retaining predictable behavior during robotic interaction. In so doing, we build on the results of fellow researchers, such as Balkcom and Mason (2008) and Bell (2010). In particular, we exploit the use of gravity by

[1]Electrical Engineering and Computer Science, College of Engineering, University of California, Berkeley, CA, USA
[2]School of Computing, University of Utah, Salt Lake City, UT, USA
[3]Max-Planck Institute for Informatics, Saarbrücken, Germany

Corresponding author:
Stephen Miller, Electrical Engineering and Computer Science, College of Engineering, University of California, Berkeley, CA 94720, USA.
Email: sdmiller@eecs.berkeley.edu

considering clothing articles which may be separated into at most two parts: one part which lies flat on a horizontal surface and one (possibly empty) which hangs vertically from the robot's grippers parallel to the gravity vector, separated by a single line which we deem the *baseline*. In so doing, we replace an infinitely dense mesh with a finite-sided polygon, and reduce the recognition and planning tasks to shape-fitting and two-dimensional geometry, respectively.

The work that follows considers three questions:

1. What is this simplified model, and under what assumptions is it a valid approximation for cloth behavior?
2. Given a cloth polygon, how can a robot be made to execute folds on it?
3. Given a single image of a clothing article, how can such a polygonal representation be extracted?

The remainder of this paper is organized as follows. In Section 2 we discuss work related to both the manipulation and detection of deformable objects. In Section 3 we define the folding problem, and present a subset of cloth configurations under which the folding task becomes purely geometric. We additionally discuss the assumptions under which our predictions may reasonably hold. In Section 4 we describe an algorithm to compute the manipulation necessary to allow a robot to execute a folding sequence, using a set of primitives deemed *g-folds*. In Section 5 we devise a perceptual scheme for extracting, from a single image of a clothing article, a category-level polygonal representation on which the above algorithm may act; one which both classifies the presented article, and projects its structure a simpler polygon which may be more easily folded. In Section 6 we show the experimental results of our approach, both in folding clothing articles when the category and cloth polygon are known, and in inferring this knowledge visually when unknown. We combine these tools to implement a complete folding system on a Willow Garage PR2 robot. We conclude in Section 7.

This paper brings together two previously separate bodies of work. The g-fold formalism and subsequent motion planning strategies was first presented by van den Berg et al. (2010). The task of classifying and recognizing the pose of clothing was done by Miller et al. (2011). We now present, for the first time, our complete framework for robotic laundry folding.

2. Related work

2.1. Manipulation

In the work of Bell and Balkcom (2010), grasp points necessary to immobilize a polygonal non-stretchable piece of cloth are computed. Gravity is used by Bell (2010) to reduce the number of grasp points required to hold cloth in a predictable configuration, potentially with a single fold, to two grippers. We extend this work and include a folding surface. We assume that points that are lying on a table are fixed by friction and gravity, and need not be grasped. Bell's work also demonstrates how to fold a T-shirt using the *Japanese method*;[1] this fold can be achieved by grasping the cloth at three points without regrasping.

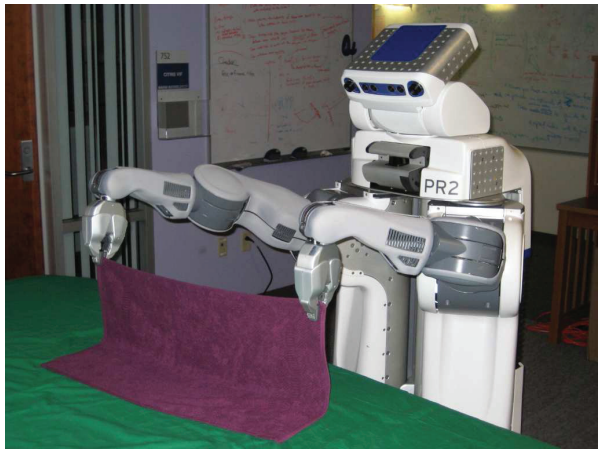

Fig. 1. The PR2 robotic platform (developed by Willow Garage) performing a g-fold on a towel.

Fahantidis et al. (1997) discuss robotic handling of cloth material with application to a number of specific folds. The work of Osawa et al. (2006) also discusses a specific folding manipulation. The work of Maitin-Shepard et al. (2010) deals specifically with folding towels. This work focuses on visual detection of the vertices of the towel, and uses a scripted motion to achieve folds using a PR-2 robot. We build on the results of this work in our experiments.

Some prior work also describes robots using tools and the design of special purpose end-effectors as a step towards laundry folding. For example, Osawa et al. (2006) developed a robot capable of using a 'flip-fold' for folding and a plate for straightening out wrinkles. Salleh et al. (2007) present an inchworm gripper for tracing the edge of a piece of clothing. A range of gripper designs is presented by Monkman (1995).

There is also quite a large body of work on *cloth simulation*, which simulates the behavior of cloth under manipulation forces using the laws of physics, including those of Baraff and Witkin (1998), Bridson et al. (2002), and Choi and Ko (2002). In our work, we manipulate cloth such that it behaves quasi-statically, allowing us to reason about the geometry of the cloth, while avoiding complex physics or dynamics.

Folding has been extensively studied in the context of *origami*. Balkcom and Mason (2008) consider a model of paper where unfolded regions are considered to be rigid facets connected by creases which form 'hinges', and detail a folding procedure which respects the assumptions of this model. Our approach is similar, in that we also consider a subset of the configuration space where the dynamics are simpler. However, unlike the hinge model, our cloth model assumes full flexibility and requires no bending energy. While this assumption yields a notably different fold

manipulation, the formalism of the fold lines themselves is similar, and we draw from results in paper folding in our own work. Applications of paper folding outside origami include box folding as presented by Liu and Dai (2003) and metal bending as presented by Gupta et al. (1998), where the material model is essentially the same as that of paper.

2.2. Perception

Estimating the configuration of a clothing article can be seen as an instance of an articulated pose estimation task. Classic articulated pose estimation methods iteratively fit or track an articulated model, updating the pose of individual part segments subject to the overall body constraints. Early methods such as those of Bregler and Malik (1998) were based on optic flow and linearized exponential map-based constraints; subsequent approaches include the efficient sampling methods of Sidenbladh et al. (2002), exemplar methods of Demirdjian et al. (2003), regression strategies of Urtasun and Darrell (2008), and subspace optimization methods of Salzmann and Urtasun (2010). Borgefors (1988) used an energy-optimization strategy to match edge points between two images.

Related models for fully non-rigid shape modeling and estimation are typically based on a learned manifold, e.g. active appearance models as proposed by Cootes and Taylor (2001). Few methods investigate clothing explicitly. Notable exceptions to this are the recent work of Guan et al. (2010), which expands the SCAPE manifold-based model of Anguelov et al. (2005) to include a model of three-dimensional clothing forms; the work in person tracking systems by Rosenhahn et al. (2007), which attempt to account for clothing variation; and methods for estimating folds in deformable surfaces proposed by Salzmann and Fua (2009). In this work we adopt a much simpler model, and propose schemes for direct optimization of specific shape forms that can be directly related to ensuing manipulation.

Fahantidis et al. (1997) describe the isolated executions of grasping a spread-out material, folding a spread-out material, laying out a piece of material that was already being held, and flattening wrinkles. Their perception system relies on a library of exact, polygonal shape models of the instances considered and then matches the sequence of extracted edges.

There is a body of work on recognizing categories of clothing. For example, Osawa et al. (2007) and Hamajima and Kakikura (2000) present approaches to spreading out a piece of clothing using two robot arms and then classifying its category.

Yamakazi and Inaba (2009) present an algorithm that recognizes wrinkles in images, which in turn enables them to detect clothes in a scene. Kobori et al. (2010) have extended this work towards flattening and spreading clothing. Kita et al. (2004) fit the geometry of the silhouette of a hanging piece of clothing to the geometry of a mass spring model of the same piece of clothing and are able to infer some three-dimensional information about the piece of clothing merely from its silhouette.

3. Problem description

Let us begin with a description of the folding task. We assume gravity is acting in the downward vertical ($-z$) direction and a sufficiently large planar table in the horizontal (xy) plane. We assume the article of clothing can be fully described by a simple *polygon* (convex or non-convex) initially lying on the horizontal surface. We are given the initial n vertices of the polygonal cloth in counterclockwise order.

We make the following assumptions on the cloth material:

1. The cloth has *infinite flexibility*. There is no energy contribution from bending.
2. The cloth is *non-stretchable*. No geodesic path lengths can be increased.
3. The cloth has *no slip* between either the surface on which it lies or itself.
4. The cloth has *zero thickness*.
5. The cloth is *subject to gravity*.
6. The behavior of the cloth is *quasi-static*: the effects of inertia are negligible.

At the core of our approach is the following additional assumption, which we call the *downward tendency assumption:*

7. If the cloth is 'released' from any gripped state, no point of the cloth will ever move upwards as a result of only gravity and internal forces within the cloth.[2]

The above assumptions do not directly follow from physics, rather they are an approximation which seems to match the behavior of reasonably shaped cloth, such as everyday clothing articles, surprisingly well, allowing us to reason purely about the geometry of the cloth: the state space consists of just configurations, and cloth motion is readily determined from hand motion.

The downward-tendency assumption allows the cloth to be held by the grippers such that one section lies horizontally on the surface and another section hangs vertically. The line that separates the horizontal and the vertical parts is called the *baseline*. To ensure deterministic behavior of the cloth, the grippers must be arranged such that the vertical section does not deform, i.e. such that it does not change its shape with respect to the original (potentially stacked) geometry. The points that are lying on the surface (including those on the baseline) are immobilized, as they cannot move in the plane due to friction and will not move upward per the downward-tendency assumption, so they need not be grasped. Figure 2 shows an example, where points of the cloth are held by grippers.

To ensure that the vertical part of the cloth does not deform, we employ the following theorem:

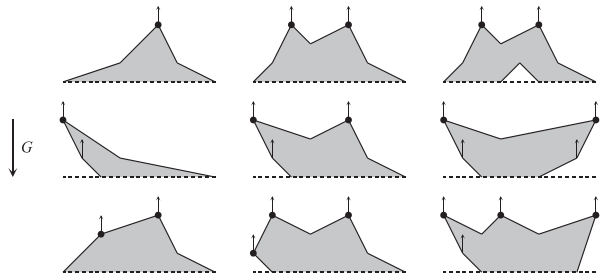

Fig. 2. Examples of vertical parts of cloths in various configurations. In order for the cloth not to deform, all convex vertices not at the baseline at which the negative gravity vector (small arrows) does not point into the cloth must be grasped. These vertices are indicated by the dots.

Theorem 1. *A vertically hanging cloth polygon is immobilized when every* convex *vertex of the cloth at which the negative gravity vector does not point into the cloth polygon is fixed (i.e. be held by a gripper or be part of the baseline).*

Proof. See Appendix A. □

A *g-fold* (*g* refers to gravity) is specified by a directed line segment in the plane that partitions the polygon into two parts, one to be folded over another. A g-fold is successfully achieved when the part of the polygon to the *left* of the directed line segment is folded across the line segment and placed horizontally on top of the other part, while maintaining the following property:

- At all times during a folding procedure, every part of the cloth is either horizontal or vertical, and the grippers hold points on the vertical part such that it does not deform (see Figure 3).[3]

This ensures that the cloth is in a fully predictable configuration according to our material model at all times during the folding procedure.

A *g-fold sequence* is a sequence of g-folds as illustrated in Figure 4. After the initial g-fold, the *stacked* geometry of cloth allows us to specify two types of g-fold: a 'red' g-fold and a 'blue' g-fold. A blue g-fold is specified by a line segment partitioning the polygon formed by the *silhouette* of the stacked geometry into two parts, and is successfully achieved by folding the (entire) geometry left of the line segment. A red g-fold is similarly specified, but only applies to the geometry that was folded in the previous g-fold (see Figure 5).

We are given a robot with k point grippers that can grasp the cloth at any point on the *boundary* of the polygon formed by the silhouette of the stacked geometry. At each such point, the gripper will grasp all layers of the stack at that point (i.e. it is not capable of distinguishing between layers). Each of the grippers is able to move independently above the *xy*-plane and we assume that gripper motion is exact.

The problem we discuss in this paper is then defined as follows. Given a specification of a sequence of g-folds, determine whether each of the folds are feasible given the constraints and, if so, compute the number of grippers needed and the manipulation motion for each of the grippers to achieve the g-folds.

4. Fold execution

In this section, we describe the algorithm that addresses the problem as formulated above. We first discuss single g-folds on unstacked geometry (Section 4.1) and then sequences of g-folds and stacked geometry (Section 4.2).

4.1. Single g-folds on unstacked geometry

Here we discuss the case of performing a single g-fold of the original (unstacked) polygon. During the manipulation, the cloth must be separated in a vertical part and a horizontal part at all times. The line separating the vertical part and the horizontal part is called the *baseline*.

Given a polygonal cloth and a specification of a g-fold by a directed line segment (e.g. the first g-fold of Figure 5(a)), we plan the manipulation as follows. The manipulation consists of two phases. In the first phase, the part of the cloth that needs to be folded is brought vertical above the line segment specifying the g-fold (see Figure 3(1)–(3)). In the second phase, the g-fold is completed by manipulating the cloth such that the vertical part is laid down on the surface with its original normal reversed (Figure 3(3)–(5)).

Let us look at the configuration the cloth is in when the part of the polygon left of the line segment is fully vertical above the line segment (see Figure 3(3)). Each convex vertex at which the negative gravity vector does not point into the cloth must be grasped by a gripper. This set of vertices can be determined in $O(n)$ time, if n is the number of vertices of the cloth. We show here that the entire g-fold can be performed by grasping only vertices in this set.

The first phase of the g-fold is bringing the part of polygon that is folded vertically above the line segment specifying the g-fold. We do this as shown in Figure 3(1)–(3), manipulating the cloth such that the baseline of the vertical part is parallel to the line segment at all times. Initially, the 'baseline' is outside the cloth polygon (meaning that there is no vertical part) and is moved linearly towards the line segment specifying the g-fold.

In the second phase, the g-fold is completed by laying down the vertical part of the cloth using a mirrored manipulation in which the baseline is again parallel to the line segment at all times. Initially the baseline is at the line segment specifying the g-fold and is moved linearly outward until the baseline is outside the folded part of the polygon (see Figure 3(3)–(5)).

The corresponding motions of the grippers holding the vertices can be computed as follows. Let us assume without loss of generality that the line segment specifying the

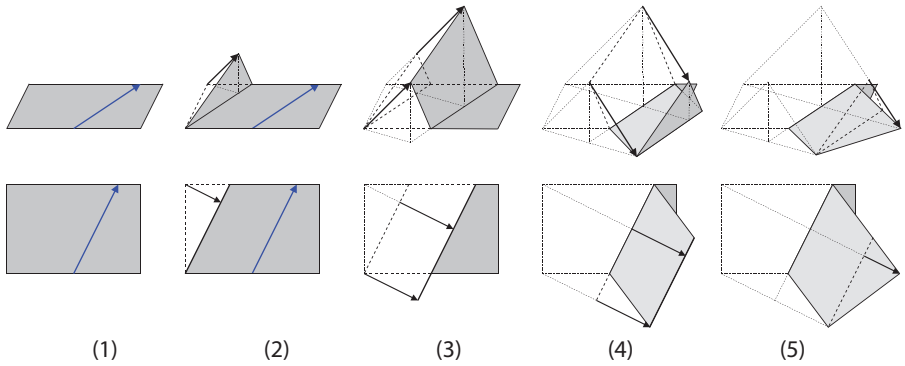

Fig. 3. The motion of two grippers (arrows) successfully performing the first g-fold specified in Figure 5(a) shown both in a three-dimensional view and top view. At all times, all parts of the cloth are either vertical or horizontal and the cloth does not deform during the manipulation. The boundary between the vertical part and the horizontal part of the cloth is called the *baseline*.

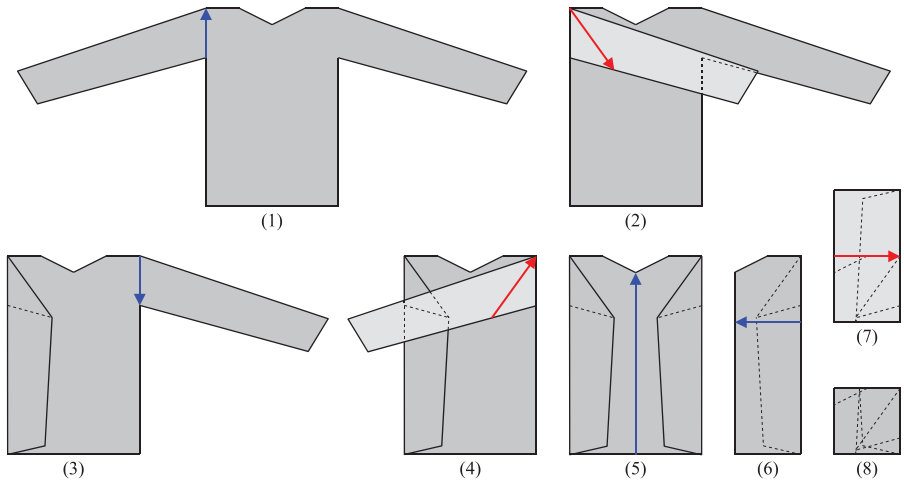

Fig. 4. Folding a long sleeve into a square using a sequence of seven g-folds. 'Red' g-folds apply to the geometry that was folded in the preceding g-fold. 'Blue' g-folds apply to the entire stacked geometry.

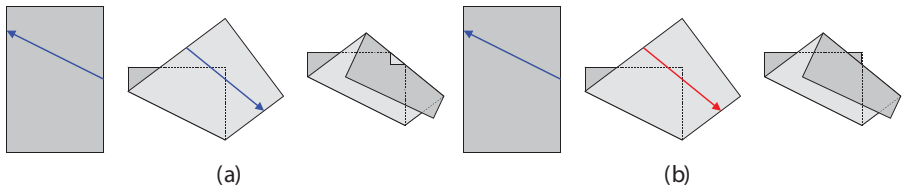

Fig. 5. (a) A g-fold is specified by a directed line segment partitioning the (stacked) geometry into two parts. The g-fold is successfully achieved when the part of the geometry left of the line segment is folded around the line segment. A sequence of two g-folds is shown here. (b) A g-fold sequence similar to (a), but the second g-fold (a red g-fold) is specified such that it only applies to the part of the cloth that was folded in the previous g-fold.

g-fold coincides with the x-axis and points in the positive x-direction. Hence, the part of the polygon above the x-axis needs to be folded. Each convex vertex of this part in which the positive y-vector points outside of the cloth in its initial configuration needs to be held by a gripper at some point during the manipulation. We denote this set of vertices by V. Let y^* be the maximum of the y-coordinates of the vertices in V. Now, we let the baseline, which is parallel to the x-axis at all times, move 'down' with speed 1, starting at $y_b = y^*$, where y_b denotes the y-coordinate of the baseline. Let the initial planar coordinates of a vertex $v \in V$ be (x_v, y_v). As soon as the baseline passes y_v, vertex v starts to be manipulated. When the baseline passes $-y_v$, vertex v stops being manipulated. During the manipulation, the vertex is held precisely above the baseline. In general, the three-dimensional coordinate $(x(y_b), y(y_b), z(y_b))$ of the

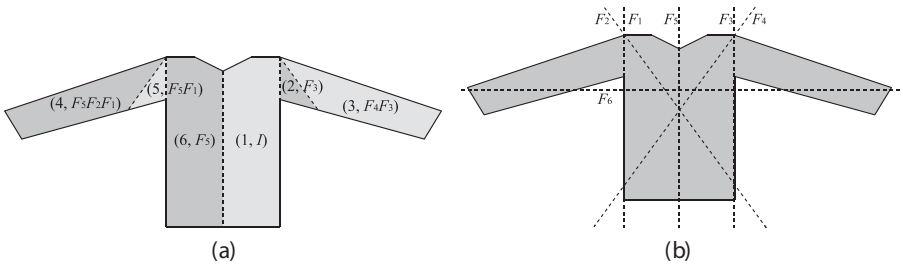

Fig. 6. (a) The representation of folded stacked geometry. The example shown here is the long-sleeved shirt of Figure 4 after five g-folds. With each facet, the stack height (integer) and a transformation matrix is stored. (b) Each transformation matrix F_i corresponds to mirroring the geometry in the line segment specifying the ith g-fold.

gripper holding vertex v as a function of the y-coordinate of the baseline is given by

$$x(y_b) = x_v \quad (1)$$

$$y(y_b) = y_b \quad (2)$$

$$z(y_b) = y_v - |y_b| \quad (3)$$

for $y_b \in [y_v, -y_v]$. Outside of this interval, the vertex is part of the horizontal part of the cloth and does not need to be grasped by a gripper. This reasoning applies to all vertices $v \in V$. When the baseline has reached $-y^*$, all vertices have been laid down and the g-fold is completed.

As a result, we do not need to grasp any vertex outside of V at any point during the manipulation, where V is the set of vertices that need to be grasped in the configuration where the part of the cloth that is folded is vertical above the line segment specifying the g-fold (in this case the x-axis). At all other points in time the vertical part is a subset that has exactly the same orientation with respect to gravity, so the same amount of vertices, or fewer, needs to be grasped. Hence, the set of vertices that need to be grasped and the motions of them can be computed in $O(n)$ time.

4.2. Sequences of g-folds and stacked geometry

Here we discuss the case of folding already folded geometry. First, we discuss how to represent folded, stacked geometry. Let us look at the example of the long-sleeve T-shirt of Figure 4, and in particular at the geometry of the cloth after five g-folds. The creases of the folds have subdivided the original polygon into facets (see Figure 6(a)). With each such facet, we maintain two values: an integer indicating the height of the facet in the stacked geometry (1 is the lowest) and a transformation matrix indicating how the facet is transformed from the original geometry to the folded geometry. Each transformation matrix is a product of a subset of the matrices F_i that each correspond to the mirroring in the line segment specifying the ith g-fold. In Figure 6(b), we show the lines of each of the g-folds with the associated matrix F_i.

Given the representation of the current stacked geometry and a line segment specifying a new g-fold, we show how we manipulate the cloth to successfully perform the g-fold or report that the g-fold is infeasible. We assume that the line segment specifying the g-fold partitions the silhouette of the stacked geometry into two parts (i.e. a blue g-fold). Let us look at the sixth specified g-fold in the long-sleeve T-shirt example, which folds the geometry of Figure 6.

Each facet of the geometry (in its folded configuration) is either fully to the left of the line segment, fully to the right, or intersected by the line segment specifying the g-fold. The facets intersected by the line segment are subdivided into two new facets, both initially borrowing the data (the stack height and the transformation matrix) of the original facet. Now, each facet will either be folded, or will not be folded. Figure 7 shows the new geometry in the long-sleeve T-shirt example after subdividing the facets by the line segment specifying the g-fold. The gray facets need to be folded.

As in the case of folding planar geometry, for each facet each convex vertex at which the gravity vector points outside of the facet at the time it is above the line segment specifying the g-fold should be held by a gripper, and each non-convex vertex or convex vertices where the negative gravity vector points inside the facet need not be held by a gripper. If a vertex is part of multiple facets, and according to at least one facet it needs not be held by a gripper, it does not need to be held by a gripper.

For the T-shirt example, the vertices that need to be grasped are shown using dots in Figure 7 and labeled $v_1, \ldots, v_7$. Applying the transformation matrices stored with the incident facet to each of the vertices shows that v_1, v_3, v_5, and v_7 will coincide in the plane. As a gripper will grasp all layers the geometry, only one gripper is necessary to hold these vertices. Vertex v_4 also needs to be held by gripper. Vertices v_2 and v_6 remain, but they need *not* be grasped. This is for the following reason. As can be seen in Figure 4, these vertices are fully *covered*. That is, the vertex is 'hidden' behind other facets of the cloth both below and above it in the stacked geometry. As we assume that the friction between two pieces of the cloth is infinite, this vertex will not be able to deform as a result of gravity, and need not be grasped. Using the heights stored at each facet, we can compute for each vertex whether it is covered or not.

This defines fully what vertices need to be grasped to achieve a g-fold of stacked geometry. If any such vertex is not on the boundary of the silhouette of the stacked

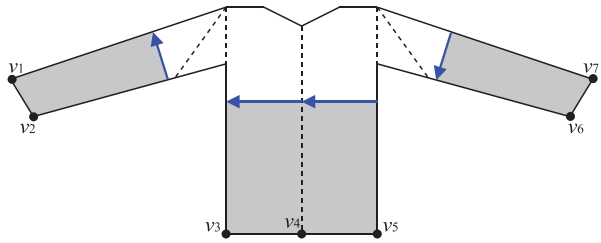

Fig. 7. The geometry in the long-sleeve T-shirt example after subdividing the facets by the line segment specifying the sixth g-fold. The gray facets need to be folded. The convex vertices for which the negative gravity vector points outside of the facet are shown using dots.

geometry, the g-fold is infeasible (for example, the second g-fold of Figure 5 (a) is infeasible for this reason). The 3-D motion of the grippers can be computed in the same way as for planar geometry, as discussed in Section 4.1. The running time for computing the vertices that need to be grasped is in principle exponential in the number of g-folds that preceeded, as in the worst case i g-folds create 2^i facets. If we consider the number of g-folds a constant, the set of vertices that need to be grasped can be identified in $O(n)$ time.

After the g-fold is executed, we need to update the data fields of the facets that were folded in the geometry: each of their associated transformation matrices is *pre-multiplied* by the matrix F_i corresponding to a mirroring in the line segment specifying the g-fold (F_6 in Figure 6(b) for the T-shirt example). The stack height of these facets is updated as follows: the order of the heights of all facets that are folded is *reversed*, and these facets are put on top of the stack. In the example of Figure 7, the facets that are folded have heights 4, 6, 1, and 3 before the g-fold, and heights 8, 7, 10, and 9 after the g-fold, respectively.

The above procedure can be executed in series for a sequence of g-folds. Initially, the geometry has one facet (the original polygon) with height 1 and transformation matrix I (the identity matrix). If a g-fold is specified to only apply to the folded part of the geometry of the last g-fold (a 'red' g-fold), the procedure is the same, but only applies to those facets that were folded in the last g-fold. We allow these kinds of g-folds as a special primitive if they need the same set of vertices to be grasped as the previous g-fold. Even if the vertices that are grasped are not on the boundary of the silhouette of the geometry, the g-fold can be achieved by not releasing the vertices after the previous g-fold. This enriches the set of feasible fold primitives.

5. Determining the cloth polygon

The above sections establish a robust framework for manipulating a clothing article given an approximate polygonal representation of its configuration. In this section, we examine the problem of visually inferring this representation, by both classifying which type of clothing article (e.g. towel, pants, or shirt) is present in a single image, and identifying an annotated polygon by which it can be approximated when folding. We additionally consider the problem of visually tracking folds, to gauge progress and accuracy throughout the folding procedure.

Spread crudely on a table, real-world clothing items do not perfectly resemble simple polygons. They contain curves rather than straight lines, corners which are rounded rather than sharp, and small intricacies which no two articles of a given class will necessarily share. Rather than reason explicitly about these complex shapes, we wish to transpose them as best we can into a shape which we know how to fold: in our experiments, this will be one of the four polygons detailed in Figure 12, either spread out or partially folded.

To do so, we employ a top-down approach to pose estimation: if a particular class of polygon is desired, let the article's shape be approximated by the best-fitting instance of that class, governed by some choice of distance metric. This draws upon the template-matching approach of Borgefors (1988), in which a polygonal template is iteratively fit to an observed contour. As will be explored in Section 6.2, however, a priori knowledge of clothing structure (for instance, the symmetry between left and right sleeves) may be exploited to greatly improve the resulting fit. We therefore consider an augmented representation, deemed the parametrized shape model, which preserves this constrained internal structure, as well as a scheme for optimizing fit while incrementally relaxing these constraints.

In Section 5.1 we formalize the notion of a parametrized shape model. For every clothing category, we attempt to find a minimal set of parameters which can describe the range of shapes it may take on. Every legal setting of these parameters defines a polygon. We call the vertices of this polygon the landmark points, which may be used as input to the folding algorithm in Section 4. Figure 8(a) shows a parametrized shape model for T-shirts as well as the polygon associated with this particular model instantiation. We further augment this representation to include folded versions of these articles.

In Sections 5.2 and 5.3 we propose an optimization strategy for determining the parameters which best fit an observed contour. In Section 5.4, we demonstrate how this fit may also be used to classify the clothing article. What results is a class-level description of the observed clothing article, and a polygon to represent it.

5.1. Parametrized shape models

We define a *model* M by the following components:

A *landmark generator*

$$M_{LG} : \{P \in \mathbb{R}^p\} \rightarrow \{L \in \mathbb{R}^{2\times \ell}\}$$

which takes a parameter vector P as input, and returns the associated collection of landmark points, L.[4]

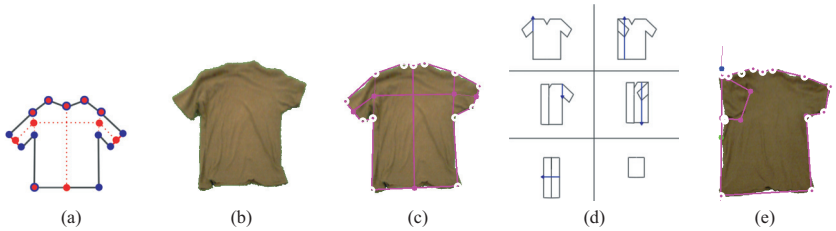

Fig. 8. (a) A parametrized shape model for a T-shirt. Red indicates a skeletal parameter, blue indicates a landmark point. (b) An example instance. (c) The result from running our approach: our model (pink) is overlayed onto the image of the clothing it is trying to fit. The determined landmark points are shown as white dots. (d) An example of a parametrized specification of a folding sequence. All fold lines shown are defined relative to the parametrized shape model. (e) Result from running our approach when a fold line is present.

A *contour generator*

$$M_{CG} : \{P \in \mathbb{R}^p\} \rightarrow \{C \in \mathbb{R}^{2\times c}\}$$

which takes a set of scalar parameters P as input, and returns the contour of the polygon which would arise from the given parameters, with a fixed number of samples per side.

A *legal input set*

$$M_{\mathcal{L}} \subseteq \mathbb{R}^p$$

which define the set of parameters in which M may reasonably be found.

A *transformation operator*

$$M_T : \{P \in \mathbb{R}^p, T \in \mathbb{R}^2, \theta \in \mathbb{R}, s \in \mathbb{R}\} \rightarrow \{P' \in \mathbb{R}^p\}$$

which transforms a set of parameters in such a way that the resultant contour M_{CG} will be translated, rotated, and scaled by the given values of T, θ, and s.

5.1.1. Skeletal models To capture the structure of the clothing, we parametrize a model about a set of interior (or *skeletal*) points, as well as features which detail the distance from the interior points to the contour. These may include landmark vertices, displacements between a skeletal vertex and its nearest edge, or scalars such as height and width.

Figure 8(a) shows an example skeletal model for a T-shirt; a more detailed list of the parameters, as well as all other skeletal models, may be found in Appendix B. The parameters are highlighted in red, and the landmark points are highlighted in blue. A red point with a blue outline indicates a landmark point which is itself a parameter. The generated contour is outlined in black. The legal input set is detailed in Appendix B.3.

5.1.2. Folded models Once the pose of the spread-out article has been determined, we wish to visually track the progress and accuracy of our folding procedure. To any model M^0, we may wish to add a single fold line. We thus define a *folded model*, such that

$$P^{\text{folded}} = \left[\Theta \mid P^0\right]$$
$$M_{\mathcal{L}}^{\text{folded}} \equiv \mathbb{R}^4 \times M_{\mathcal{L}}^0,$$

where all parameters of the original model P^0 are allowed to vary and, in addition, the parameters Θ specify a directed line segment about which the model is to be folded. The resulting landmark points are computed by folding the polygon specified by $M_{LG}^0(P^0)$ about this line. Note that there is no restriction on what sort of model M^0 is. This allows us to specify folds recursively. If M^0 is unfolded, M^{folded} will contain a single fold. If M^0 contains a single fold, M^{folded} will contain two, and so on.[5]

If we are certain the clothing article did not move during a folding operation, we may reduce this task to finding a single fold line on a known polygon, rather than determining both simultaneously. We therefore define a *static folded model*, such that

$$P^{\text{folded}} = \left[\Theta\right]$$
$$M_{\mathcal{L}}^{\text{folded}} \equiv \mathbb{R}^4.$$

5.2. Energy function

We now aim to find the parameters which optimally fit a given image. Our approach extracts the contour of the clothing article in the image and uses an energy function which favors contour fit. We define the energy E as follows:

$$E(P) = (\alpha) \times \bar{d}(M_{CG}(P) \rightarrow C) + (1-\alpha) \times \bar{d}(C \rightarrow M_{CG}(P)),$$

where $\bar{d}(A \rightarrow B)$ is the average nearest-neighbor distance[6] from A to B:

$$\bar{d}(A \rightarrow B) \equiv \frac{1}{|A|} \sum_{a \in A} \underset{b \in B}{\text{argmin}} \|b - a\|.$$

The parameter α is used to adjust the way in which the model fits to the contour. If α is too low, the model will attempt to fit every point of the contour, often overfitting to deviations such as wrinkles. If α is too high, the model may cease to cover the contour at all, fixating instead on a single portion. We have found that setting $\alpha = 0.5$ is sufficient to counter both negative tendencies.

5.3. Energy optimization

Our energy optimization follows a coarse-to-fine strategy, in which the parameter space begins small and increases

as the procedure continues. It first only considers translation, rotation and scale, then considers all parameters but enforces certain symmetry constraints amongst them, and finally optimizes over all parameters without the symmetry constraints.

5.3.1. Initialization

PCA approach To infer the necessary translation, rotation, and scale, we rely on a principal component analysis (PCA) of the observed contour, and contour defined by the model.

We first compute the initial model contour as

$$M_c = M_{CG}(P_0).$$

We then calculate the centers of mass of the observed contour and the model contour: c_o and c_m, respectively. We then compute the relative translation between the two contours,

$$T = c_o - c_m.$$

We then perform PCA to estimate the principal axes of each contour, denoted by a_o and a_m. We compute the relative angle between the two axes

$$\theta = \arccos(a_o \cdot a_m).$$

Finally, for each contour we find the point of intersection between the top of the contour and its principal axis, denoted t_o and t_m. We compute the relative scale between the two contours as

$$s = \frac{\|t_o - c_o\|}{\|t_m - c_m\|},$$

which is approximately the ratio of the heights of the two contours. The resultant contour $M_c(P)$ will be centered about c_o, and scaled and rotated such that $t_o = t_m$.[7]

Having computed these three values, we then update our model estimate such that

$$P' \leftarrow M_T(P, T, \theta, s).$$

Multi-angle approach We additionally consider a second approach, in which the optimization is run with multiple initializations, attempting all possible rotations within a granularity of $\delta\theta$. Upon completion, the fitted model which yields the lowest energy function is chosen, and all others are discarded. The method for choosing translation and scale is the same as in the PCA approach.

5.3.2. Optimization To ensure the best possible fit, our standard approach performs the optimization in three phases: orientation, symmetric, and asymmetric.

In the *orientation phase*, all parameters are held relatively fixed, with only one external degree of freedom: θ, which defines the net rotation of the contour points about the center of gravity of the model. This phase is only run when using the PCA-based initialization, and it tends to improve the orientation estimate as it considers the entire contour, rather than just its principal component. When using the multi-angle initialization we found it better to skip the orientation phase as it reduced the variety of orientations explored.

In the *symmetric phase*, the model is free to translate, rotate, scale, or deform within the limits determined by its legal input set, as long as left–right symmetry is maintained. In terms of implementation, this is done by optimizing over a subset of the model parameters, those which describe the left and center portions of the model, and computing the implied values for the remaining right parameters such that symmetry is enforced.

In the *asymmetric phase*, all parameters are optimized over, and the model is free to translate, rotate, scale, or deform within the limits determined by its legal input set.

For the numerical optimization, we use coordinate-wise descent over the parameters: evaluating the gradients numerically (rather than analytically) and maintaining an adaptive step size for each parameter. This algorithm is presented in detail in Appendix C.

To enforce legality constraints on the parameters, we augment the energy function with a penalty for constraint violation. We first normalize the fit such that

$$\forall P : 0 \leq E_{\text{norm}}(P) < 1.$$

To do so, we set

$$E_{\text{norm}} = \frac{E}{E_{\text{max}}}.$$

As a simple upper bound, E_{max} is set to $\sqrt{h^2 + w^2}$, where h and w denote the height and width of the image, respectively. This corresponds to the case in which the two contours are maximally distant given the size of the image.

We then define the structural penalty S as

$$S(P) = \begin{cases} 0 & \text{if } P \in M_{\mathcal{L}}, \\ 1 & \text{otherwise.} \end{cases}$$

The resulting energy function is then given by

$$\mathcal{C}(P) = E_{\text{norm}}(P) + S(P).$$

As the normalized energy E_{norm} lies between zero and one, the optimum of the cost function will never violate a constraint if a legal alternative exists.

5.4. Classification

For any image and specified model, the above procedure is able to return a set of fit parameters and an associated energy. By considering the value of the energy function as a measure of overall model fit, this provides a convenient means of category classification. When presented with an image and a set of possible categories, we run the above procedure multiple times, with one model associated with each category. The fitted model which results in the lowest final energy is selected, and the image is classified accordingly.

Fig. 9. Three of each clothing category were used in conducting our experiments.

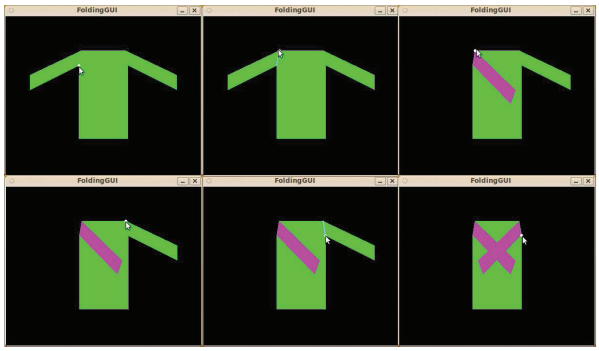

Fig. 10. An example sequence of user-specified folds. The user first clicks on the left arm pit, then on the left shoulder to specify the first fold. The program then verifies that this is a valid g-fold for the chosen number of grippers. In this case it is, and it then shows the result after executing the g-fold (third image in the top row). Then the user specifies the next fold by two clicks, the program verifies whether it is a valid g-fold, and then shows the result after executing the g-fold.

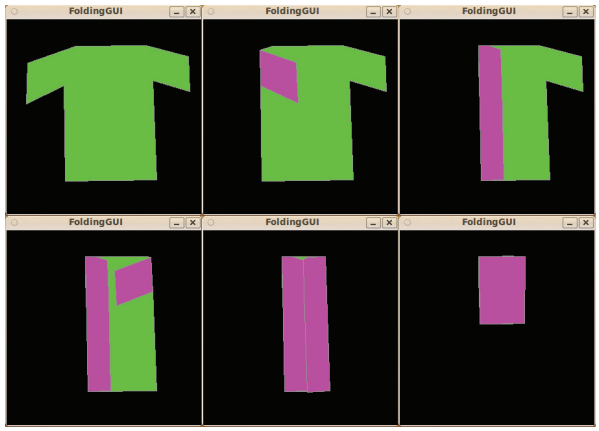

Fig. 11. An example folding primitive, automatically executed on a T-shirt polygon. Note the clean fold, despite the imperfect symmetry of the original polygon.

6. Experimental results

In this section we describe the experimental verification of our approach. We begin by validating the reliability of our g-fold mechanism for folding in Section 6.1, using human annotated cloth polygons as input. We then evaluate the success of our perceptual tools for inferring this polygon in Section 6.2, run on a hand-compiled dataset of 400 clothing images. Finally, we combine the two components into an end-to-end robotic system in Section 6.3.

6.1. Clothing manipulation

We validate the power of our g-fold framework by first implementing an open-loop laundry folding mechanism on a household robot. We first describe the setup, and then the corresponding results.

6.1.1. Experimental setup We used the Willow Garage PR2 robotic platform developed by Wyrobek et al. (2008). The PR2 has two articulated seven-axis arms with parallel jaw grippers. We used a soft working surface, so the relatively thick grippers can easily get underneath the cloth. Our approach completely specifies end-effector position trajectories. It also specifies the orientation of the parallel jaw grippers' planes. We used a combination of native IK tools and a simple linear controller to plan the joint trajectories.

We experimented with the clothing articles shown in Figure 9. Whenever presented with a new, spread-out clothing article, a human user clicks on the vertices of the article in an image. This specifies the location of the cloth polygon.

To allow for arbitrary fold sequences, we give a human user the option of manual fold specification. The user is presented with a graphical representation of the article, and the ability to draw arbitrary folds. Once a valid g-fold has been specified, the robot executes the fold, allowing the user to specify another. Figure 10 illustrates the fold sequence specification process through an example.

To autonomously execute folds on known clothing categories, the program is also seeded with a set of folding primitives. When presented with a particular article of clothing, the user is given the option of calling one of these primitives. Once called, a sequence of folds is computed, parametrized on a number of features such as scaling, rotation, and side lengths. Figure 11 shows an example primitive being executed on a user-defined polygon in the shape of a shirt. To ensure consistency across multiple trials, such primitives were used to execute the folds detailed in the following experimental results section.

While our approach assumes the cloth has zero resistance against bending, real cloth does indeed resist against bending. As a consequence, our approach outlined so far overestimates the number of grippers required to hold a piece of cloth in a predictable, spread-out configuration. Similarly, our robot grippers have non-zero size, also resulting in an overestimation of the number of grippers required. To account for both of these factors, our implementation offers the option to allocate a radius to each of our grippers, and we consider a point being gripped whenever it falls inside

this radius. To compute the grasp points, we first compute the grasp points required for point grippers and infinitely flexible cloth. We then cluster these points using a simple greedy approach. We begin by attempting to position a circle of fixed radius in the coordinate frame such that it covers the maximum number of grasp points, while subsequently minimizing the average distance from each covered point to its center. This process is iterated until no point remains uncovered. For the duration of the fold, our grippers now follow the trajectory of the center of each cluster, rather than individual grasp points.

6.1.2. Experimental results We tested our approach on four categories: towels, pants, short-sleeved shirts, and sweaters. Figure 12 shows the fold sequences used for each category. To verify the robustness of our approach, we tested on three instances of each category of clothing. These instances varied in size, proportion, thickness, and texture. At the beginning of each experimental trial, we provided the PR2 with the silhouette of the polygon through clicking on the vertices in two stereo images.

Table 1 shows success rates and timing on all clothing articles. Figure 13 shows the robot going through a sequence of folds.

As illustrated by the reported success rates, our method demonstrates a consistent level of reliability on real cloth, even when the manipulated fabric notably strays from the assumptions of our model. For instance, the g-fold method worked reasonably well on pants, despite the material's clear violation of the assumption of non-zero thickness, and a three-dimensional shape which was not quite polygonal. It was also able to fold a collared shirt quite neatly, despite that its rigid collar and buttons are not expressible in the language of our model. While these elements would likely be problematic if they intersected a g-fold, they can otherwise be ignored without issue.

Despite the simplifications inherent to our model, we have found it to match the behavior of real cloth quite closely in this setup. While human manipulation of cloth exploits a number of features which our model neglects, these features generally arise in states which our model considers unreachable. That is, handling true fabric often requires less caution than our g-fold model predicts, but rarely does it require more. Furthermore, even when unpredicted effects did arise, the final result was often not compromised.

Although factors such as thickness may cause the cloth to deviate slightly from its predicted trajectory, most often in the form of 'clumping' for thick fabrics, the resulting fold generally agrees with the model, particularly after smoothing. Much of our success can be attributed to a number of assumptions which were very closely met: namely, the lack of slip between the cloth and the table, and the lack of slip between the cloth and itself. The former allowed us to execute g-folds even when the modeled polygon did not perfectly match the silhouette of the cloth. As actual articles of clothing are not comprised solely of well-defined corners, this imprecision often resulted in a non-zero horizontal tension in the cloth during the folding procedure. However, as the friction between the cloth and the table far outweighs this tension, the cloth remained static. This allowed us to stabilize loose vertices by 'sandwiching' them between two gripped portions of cloth. This technique, in combination with the robust gripping approach detailed above, allowed us to execute a number of folds (such as the shirt folds in Figure 12) which more closely resembled their standard human counterpart. With the exception of long-sleeved shirts, all sequences could theoretically be executed by a pair of point grippers. However, some relied on the ability to create perfect 90° angles, or precisely align two edges which (in actuality) were not entirely straight. Exact precision was impossible in both of these cases; but where there was danger of gravity influencing a slightly unsupported vertex, the friction of the cloth, in conjunction with its stiffness, often kept it in a stable configuration.

The trials were not, however, without error. Most often, failure was due to the limitations of our physical control, rather than a flaw in our model. For instance, 2/2 short-sleeved failures and 3/4 long-sleeved failures occurred at steps where the robot was required to grasp a portion of previously folded sleeve (short-sleeve steps 2 and 4, long-sleeve steps 3 and 6 in Figure 12). In each of these cases, the failure could be easily predicted from the location of the initial grasp. Either the robot did not reach far enough and grasped nothing, or reached too far and heavily bunched the cloth. These failures suggest a clear issue with our original implementation: namely, the reliance on open-loop control. While the initial position of each vertex is given, the location of a folded vertex must be derived geometrically. For this location to be correct, we must make two assumptions: that the cloth at hand is perfectly represented by the given polygon, and that the trajectory, once computed, can be exactly followed. Clearly, both are idealizations: the former disregards the multi-layered nature of all but towels (which saw a 100% success rate) and the latter is hindered by the inherent imprecision of any robotic mechanism. These errors greatly entail the need for a perceptual component which can track folds over time, as detailed in Section 5.1.2 and implemented in Section 6.3.

While overall folds were often executed correctly, the resulting article often contained minor imperfections, such as wrinkles. The robot was able to remove many of these via an open-loop smoothing motion. However, in order to make the fold truly neat, more advanced manipulations, such as ironing or precisely targeted smoothing motions, would most likely be necessary.

6.2. Clothing detection

Using the methods detailed in Section 5, we designed a system able to infer the class and pose of a spread-out article of

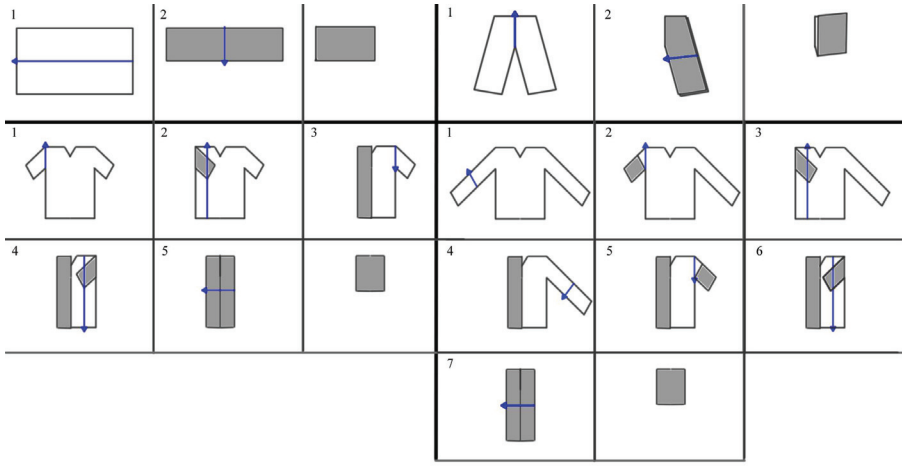

Fig. 12. The sequences of folds used in our experiments. Note that the long-sleeved fold is identical to the short-sleeved fold, with added folds for tucking in the sleeves. This may well be seen as a single primitive, parametrized about the sleeve length.

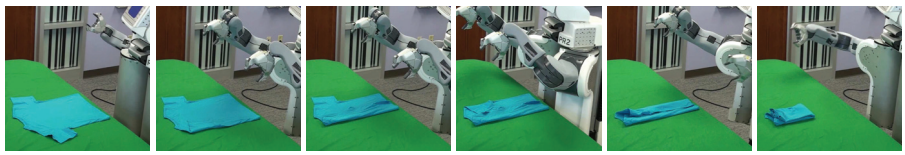

Fig. 13. The robot folding a T-shirt using our approach.

Fig. 14. The 40 articles of clothing in our dataset.

Table 1. Experimental results of autonomous laundry folding.

Category	Success rate	Average time (s)	Category	Success rate	Average time (s)
Towels	**9/9**	**200.0**	**Short-Sleeved Shirts**	**7/9**	**337.6**
Purple	3/3	215.6	Pink T-Shirt	2/3	332.8
Leopard	3/3	210.9	Blue T-Shirt	2/3	343.2
Yellow	3/3	173.5	White Collared	3/3	337.6
Pants	**7/9**	**186.6**	**Long-Sleeved Tops**	**5/9**	**439.0**
Long Khaki	3/3	184.9	Long-Sleeved Shirt	2/3	400.7
Brown	1/3	185.9	Gray Sweater	1/3	458.4
Short Khaki	3/3	189.1	Blue Sweater	2/3	457.8

clothing. To do so, we defined a set of parametrized shape models which corresponded to each clothing class (Section 6.2.1). We then collected a dataset of clothing images (Section 6.2.2) and verified our detection algorithm on this dataset (Section 6.2.4).

6.2.1. Models used For each of the four categories of clothing detailed above, we define an associated parametrized model: thus, to each article of clothing we attempt to fit a towel, pants, short-sleeved, and long-sleeved model. Each model defines a polygon which may be folded

using one of the primitives detailed in Figure 12. The parameters and constraints of these models are discussed in detail in Appendix B.

As a baseline for performance comparison, we also define a *polygonal model* which is parametrized about

$$P_{poly} = \left[l_1(x)\ l_1(y)\ \ldots\ l_\ell(x)\ l_\ell(y)\right]$$
$$\mathcal{L}_{poly} \equiv \mathbb{R}^{2\ell}.$$

This model has no interior structure, and no legality constraints beyond self-intersection. For every clothing category, we construct a polygonal model whose initial landmark points are identical to those of the skeletal model for that category. This model provides a useful baseline for the performance of pure contour fitting, beginning with the same initialization and optimization techniques, but without taking any prior knowledge about clothing into consideration.

6.2.2. Data collection To quantitatively gauge the accuracy of our approach, our shape-fitting code was run on a dataset of roughly 400 images, divided into four categories: towels, pants, short-sleeved shirts, and long-sleeved shirts. For each category, 10 representative articles of clothing were considered. These 40 articles varied greatly in size, proportion, and style (see Figure 14). Each article was then further placed in 10 or more poses, encompassing a variety of common spread-out configurations (see Figure 15).

Each object was initially photographed on a green table. To ensure rotational invariance, each image was transformed to a birdseye perspective, using OpenCV's checkerboard detector to locate the top-down frame. The background was then subtracted from each image. For most of these images, hue thresholding against the green background was sufficient: however, in cases where the complex texture of the clothing precluded hue thresholding, the Grabcut algorithm (Rother et al. 2004) was used to perform the subtraction, with foreground and background pixels manually selected by a user. Finally, the location of each landmark point was hand-annotated, to provide ground truth data for the model fitting task. The pipeline is illustrated in Figure 16.

6.2.3. Implementation details We ran our experiments on a Lenovo Thinkpad, running an Intel Core 2 Extreme Processor. A typical model fit took roughly 30 seconds; for more complex procedures such as the four-phase multi-model approach for T-shirts, convergence would occasionally take up to 2.5 minutes. To rapidly compute nearest-neighbor distances for the cost function, the Flann library (Muja and Lowe 2009) was used. The bulk of the image processing, including transformations, thresholding, and contour detection, was done with OpenCV (Bradski 2000).

6.2.4. Experimental results Each image was first fit to the proper model according to its known category. Table 2 shows the accuracy of our approach on the 400 image dataset using both the PCA and multi-angle initializations, and the performance of the associated polygon model on the same set. These results are represented pictorially in Figure 17.

Fig. 15. The article of clothing is put in various poses.

Fig. 16. The dataset pipeline. Top left: Initially, the clothing is spread out on a green table. Top right: A birdseye transformation is then performed. Bottom left: The image is cropped, and the background is segmented out. Bottom right: To provide ground truth for the fitting procedure, the resulting image is hand-annotated.

Our approach performs very well, obtaining typical accuracies of within 8 pixels per landmark point and significantly outperforming the polygonal approach, the shortcomings of which are detailed in Figure 19.

Moreover, the relative gain of the skeletal approach on each category is quite telling. As the towel model is effectively structureless, there is no distinction between the two models, and hence no improvement. In the case of pants, the proximity between the two legs frequently caused the polygonal approach to attract to poor local minima; whereas the skeletal approach, with its implicit knowledge of structure, performed quite well. Short-sleeved shirts, being fairly homogeneous in shape, proved extremely difficult for the polygonal approach to fit, as can be readily seen in Figure 17. Despite the subtlety of shoulder and collar point locations, the longer sleeves of sweaters tend to sketch out a very clear polygonal shape; thus the polygon model performed somewhat reasonably, with most errors centered about shoulders, collars, and sleeve edges.

The results of the multi-angle approach were extremely consistent with that of PCA initialization, suggesting that

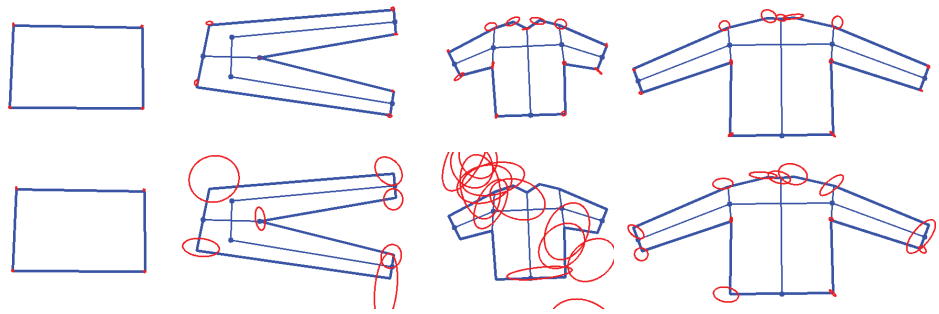

Fig. 17. Comparison of individual landmark point errors. The center of the ellipses denotes mean error, and the size and skew their covariance, projected onto a canonical version of the article. Top: Pointwise error for skeletal models using the PCA approach. Bottom: Pointwise error for polygon models.

Table 2. Results of fitting our skeletal models to the dataset. Model accuracy is measured as the average pixel distance from the predicted landmark point to the annotated landmark point.

Category	Polygon model		Skeletal model (PCA)		Skeletal model (multi-angle) $\delta\theta = 10^\circ$	
	(pixels)	(cm)	(pixels)	(cm)	(pixels)	(cm)
Towels	2.89 ± 1.78	0.75 ± 0.46	2.89 ± 1.78	0.75 ± 0.46	$\mathbf{2.86 \pm 1.75}$	$\mathbf{0.74 \pm 0.45}$
Pants	14.91 ± 35.97	3.88 ± 9.35	4.23 ± 1.64	1.10 ± 0.43	$\mathbf{4.13 \pm 1.54}$	$\mathbf{1.07 \pm 0.40}$
Short sleeved	89.63 ± 44.88	23.30 ± 11.67	6.58 ± 3.14	1.71 ± 0.82	$\mathbf{6.41 \pm 3.05}$	$\mathbf{1.67 \pm 0.79}$
Long sleeved	14.77 ± 8.27	3.84 ± 2.15	$\mathbf{7.09 \pm 3.68}$	$\mathbf{1.84 \pm 0.96}$	8.06 ± 4.52	2.09 ± 1.17

the latter approach is sufficient for most purposes. Indeed, given the inherent ambiguity in landmark location and small number of examples on which the two differed, any perceived performance advantage would best be attributed to noise.

We then examined the case of unknown clothing category. On 100% of test images, our method was able to accurately classify the clothing category. The classification scheme in Section 5.4 was used to distinguish shirts, pants, and towels. Thresholding the sleeve length at 35% the shirt width further distinguished all long-sleeved shirts from short-sleeved shirts. Therefore, the correct model is always chosen, and the performance is identical to the known, tabulated case.

Our approach, however, was not perfect. The location of collar points proved to be quite ambiguous, and were often incorrectly identified. Shoulders, while significantly localized by structural constraints, still proved a source of difficulty. Finally, the initialization was poor on a small number of instances, and in very rare cases could not be recovered from.

Fig. 18. Example results of our approach on the four categories of clothing.

6.3. A combined end-to-end system

We then combined the perception system introduced in Section 6.2 with the folding system of Section 6.1 to provide the Willow Garage PR2 with a closed-loop folding system. The system runs as follows:

- A towel, pair of pants, short-sleeved or long-sleeved shirt begins spread out on a table in front of the PR2.
- Using the approach detailed in Section 6.2, the PR2 fits a skeletal model to the contour of the observed article. To avoid grasping virtual points, the landmark points are then relocated to their nearest neighbor on the observed contour.
- The PR2 then computes the parametrized fold primitive corresponding to the newly fit polygon.

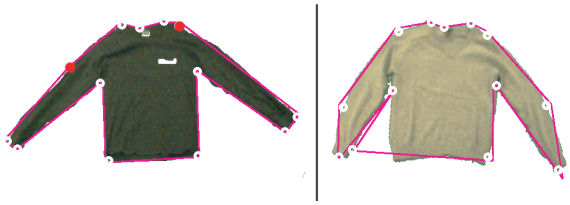

Fig. 19. Failures of the polygon model. Left: Without more detailed structural information, the model is unable to detect more subtly defined points, such as the shoulder (detected shoulder points in red). Right: The unconstrained polygon approach will generally be attracted to the nearest edge; a poor initialization can easily ruin it.

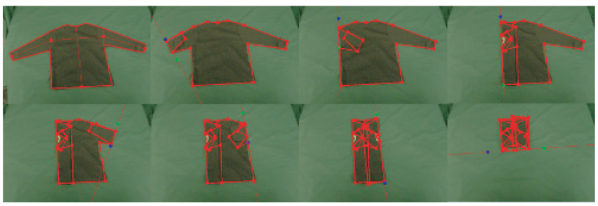

Fig. 20. The robot fits a model to the initial configuration, then tracks each fold in the procedure

- After each fold, the robot then re-examines the article. It then fits a static folded model to the newly observed contour, with initial model M^0 given by the previously determined landmark points, and parameters Θ seeded by the robot's intended fold.
- This is repeated until the article is folded.

Preliminary results have shown this approach to work consistently well, enabling fully automatic folding by the PR2. As suggested by the perception results in Section 6.2, the initial polygon detection phase has been able to eradicate human input with no notable deterioration in performance. Furthermore, the problem of failing to grasp previously folded portions of cloth, a frequent issue in the open-loop setup detailed in Section 6.1, is virtually eradicated; as errors are tracked the moment they occur, they are no longer compounded over time. (See Figure 20 for an example of the tracking process.)

A number of factors, however, continue to hinder complete robustness. Most notably, the fold-tracking system works best when the view of the camera is stationary. Therefore, the robot remains stationary during the procedure, limiting the size of folded articles to the arm span of the robot. In addition, while grasp imprecision no longer compounds over successive folds, it remains a substantial issue: particularly on smaller articles, where the error relative to the size is often fairly high.

Videos of representative successful runs and continued progress, as well as software implementations of all aforementioned algorithms are available at http://rll.berkeley.edu/IJRR2011

7. Conclusion and future work

We proposed a novel take on robotic laundry folding which averts the high dimensionality inherent to cloth by determining a set of necessary conditions under which its behavior is repeatable and known. In so doing, we greatly simplified the complexity of the system, and showed that even in this limited subspace, many folding procedures can be executed.

We further described the steps necessary to execute folds on polygonal cloth under these conditions, relying on intuitive geometric reasoning rather than computationally costly planning and simulation. Our experiments show that (i) this suffices to capture a number of interesting folds and (ii) real cloth behaves benignly, even when moderately violating our assumptions.

We also provide an approach that equips a robotic system with the necessary perception capabilities; namely, the ability to visually infer a reasonable polygonal representation of a clothing article present in an image. We show that, via a model-based optimization approach, the pose and corresponding polygon of many common clothing articles can be reliably detected.

We experimentally demonstrated that this framework is capable of enabling a household robot to fold laundry. We tested the manipulation task in an open-loop setting and found it to be quite reliable. We tested the visual components on a large dataset of images, and found it to be highly accurate both in its ability to classify and infer the configuration of spread out clothing. Finally, we combined the perception and manipulation tools on a robotic platform, and give a first look at an end-to-end system for robotic laundry folding.

Careful inspection of the gripper paths shows that a single very large parallel jaw gripper would suffice to execute a g-fold requiring an arbitrary number of point grippers. We plan to investigate a practical implementation of this idea for the PR2. However, a large gripper of this kind would reduce the collision-free workspace volume significantly.

In this work, the category-level folding primitives are specified by human users. Owing to the inherently aesthetic nature of the choice of primitive, such input may well be necessary. Yet it is interesting to consider the problem of automating this decision process, to allow for the folding of previously unseen article types in a reasonable way.

Our work assumes that, via some mechanism, an arbitrarily crumpled article of clothing may be spread out on a table. We are currently working on, and will continue to explore, a set of primitive actions which accomplish this task. We also look to expand our approach beyond folding to the entire laundry system.

Finally, while this work dealt explicitly with the task of laundry folding, we believe the tools put forth may generalize well beyond this to many deformable object manipulation challenges, such as ironing or bed-making. We hope that these results will provide another step forward on the

long road toward the utopian future, where humans are never again nagged to 'do their chores'.

Funding

This work was supported in part by the NSF (award number IIS-0904672), Willow Garage under the PR2 Beta Program, and a Feodor Lynen Fellowship granted by the Alexander von Humboldt Foundation.

Notes

1. As popularized by the video 'Japanese way of folding T-shirts!' at http://www.youtube.com/watch?v=b5AWq5aBjgE. Original footage 2006, uploaded to YouTube 2010.
2. While this assumption tends to hold for typical shapes, it is not always true. An example of where the assumption is not accurate is for an exotic family of shapes called pinwheels, as is proven by Bell (2010).
3. Not all folds can be achieved using a g-fold. In terms of origami, it is only possible to execute a *valley fold* on an upright portion of cloth, or a *mountain fold* on a flipped portion. While this renders many folds possible, more complex ones, such as *reverse folds*, cannot be done in this way. See Balkcom and Mason (2008).
4. In this work, the only information used is the resulting polygon, and hence all articles have landmark points which lie on the contour. In general, however, there is no reason that this must be the case: any point, whether virtual or on the contour, may be considered a landmark.
5. One may wonder why Θ has four dimensions rather than the expected two. This is because a fold line is not a line *per se*. Rather, it is a directed line segment, which may intersect one portion of cloth without intersecting another, colinear portion.
6. We additionally considered the use of dynamic time warping (Needleman and Wunsch 1970; Sakoe and Chiba 1978) in our distance metric. The results, however, showed little improvement, so for the sake of simplicity and computational efficiency, we restrict our approach to nearest-neighbor.
7. Thus described, the PCA approach leaves an ambiguity in terms of which direction is assumed to be 'up' on the principal axis. To resolve this, we attempt both upright and upside-down initializations, and choose the minimum-cost result after the optimization is complete.
8. In all of these models, the preferred representation of parameters was in Cartesian coordinates. We additionally explored optimizing directly over angles and lengths. In practice, however, the optimization worked best when all parameters were aperiodic and similarly scaled. Hence, whenever possible, a length/angle combination was represented by a two-dimensional point.
9. For the precise numerical constraints of all of our models, see the attached code at http://rll.berkeley.edu/IJRR2011.

References

Anguelov D, Srinivasan P, Koller D, Thrun S, Rodgers J and Davis J (2005) SCAPE: shape completion and animation of people. *ACM Trans Graph* 24(3): 408–416.

Balkcom D and Mason M (2008) Robotic origami folding. *Int J Robotics Res* 27: 613–627.

Baraff D and Witkin A (1998) Large steps in cloth simulation. In *Proceedings of SIGGRAPH 1998*.

Bell M (2010) *Flexible Object Manipulation*. PhD Thesis, Dartmouth College, 2010.

Bell M and Balkcom D (2010) Grasping non-stretchable cloth polygons. *Int J Robotics Res* 29: 775–784.

Borgefors G (1988) Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans Pattern Anal Machine Intell* 10: 849–865.

Bradski G (2000) The OpenCV Library. *Dr. Dobb's J Software Tools* 25: 120–123.

Bregler C and Malik J (1998) Tracking people with twists and exponential maps. In *Proceedings 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8–15.

Bridson R, Fedkiw R and Anderson J (2002) Robust treatment of collisions, contact, and friction for cloth animation. In *Proceedings of SIGGRAPH 2002*.

Choi K and Ko H (2002) Stable but responsive cloth. In *Proceedings of SIGGRAPH 2002*.

Cootes T and Taylor C (2001) Statistical models of appearance for medical image analysis and computer vision. http://www.isbe.man.ac.uk/∼bim/Papers/asm_aam_overview.pdf.

Demirdjian D, Ko T and Darrell T (2003) Constraining human body tracking. In *IEEE International Conference on Computer Vision*.

Fahantidis N, Paraschidis K, Petridis V, Doulgeri Z, Petrou L and Hasapis G (1997) Robot handling of flat textile materials. *IEEE Robotics Automation Mag* 4: 34–41.

Guan P, Freifeld O and Black MJ (2010) A 2D human body model dressed in eigen clothing. In *European Conference on Computer Vision, ECCV, Part I* (*Lecture Notes in Computer Science*, vol. 6311). Berlin: Springer, pp. 285–298.

Gupta SK, Bourne D, Kim K and Krishnan S (1998) Automated process planning for robotic sheet metal bending operations. *J Manufacturing Syst* 17: 338–360.

Hamajima K and Kakikura M (2000) Planning strategy for task of unfolding clothes. In *Proceedings of ICRA* vol. 32, pp. 145–152.

Kita Y, Saito F and Kita N (2004) A deformable model driven visual method for handling clothes. In *Proceedings of ICRA*.

Kobori H, Kakiuchi Y, Okada K and Inaba M (2010) Recognition and motion primitives for autonomous clothes unfolding of humanoid robot. In *Proceedings of IAS 2010*.

Liu J and Dai J (2003) An approach to carton-folding trajectory planning using dual robotic fingers. *Robotics Autonomous Syst* 42: 47–63.

Maitin-Shepard J, Cusumano-Towner M, Lei J and Abbeel P (2010) Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *Proceedings IEEE International Conference on Robotics and Automation*, 2010.

Miller S, Fritz M, Darrell T and Abbeel P (2011) Parametrized shape models for clothing. In *Proceedings of ICRA*, Berkeley, CA.

Monkman GJ (1995) Robot grippers for use with fibrous materials. *Int J Robotics Res* 14: 144–151.

Muja M and Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application (VISSAPP)*.

Needleman SB and Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol* 48: 443–453.

Osawa F, Seki H and Kamiya Y (2006) Clothes folding task by tool-using robot. *J Robotics Mechatron* 18(5): 618–625.

Osawa F, Seki H and Kamiya Y (2007) Unfolding of massive laundry and classification types by dual manipulator. *JACIII* 11: 457–463.

Rosenhahn B, Kersting U, Powell K, Klette R, Klette G and Seidel H (2007) A system for articulated tracking incorporating a clothing model. *Machine Vis Appl* 18: 25–40.

Rother C, Kolmogorov V and Blake A (2004) "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Trans Graph* 23: 309–314.

Sakoe H and Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans Acoust Speech Signal Process* 26: 43–49.

Salleh K, Seki H, Kamiya Y and Hikizu M (2007) Inchworm robot grippers in clothes manipulation optimizing the tracing algorithm. In *International Conference on Intelligent and Advanced Systems, 2007. ICIAS 2007*, pp. 1051–1055.

Salzmann M and Fua P (2009) Reconstructing sharply folding surfaces: a convex formulation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Salzmann M and Urtasun R (2010) Combining discriminative and generative methods for 3D deformable surface and articulated pose reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Sidenbladh H, Black M and Sigal L (2002) Implicit probabilistic models of human motion for synthesis and tracking. In *Computer Vision – ECCV 2002*, pp. 784–800.

Urtasun R and Darrell T (2008) Sparse probabilistic regression for activity-independent human pose inference. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pp. 1–8.

van den Berg J, Miller S, Goldberg K and Abbeel P (2010) Gravity-based robotic cloth folding. In *Proceedings 9th International Workshop on Algorithmic Foundations of Robotics (WAFR)*.

Wyrobek K, Berger E, Van der Loos HFM and Salisbury K (2008) Towards a personal robotics development platform: rationale and design of an intrinsically safe personal robot. In *Proceedings of ICRA*.

Yamakazi K and Inaba M (2009) A cloth detection method based on image wrinkle feature for daily assistive robots. In *IAPR Conference on Machine Vision Applications*, pp. 366–369.

Appendix A: Proof of Theorem 1

Let us assume for the purpose of the proof that the polygon lies/hangs in the xz-plane, and that gravity points in the $-z$ direction. The term *above* refers to having a higher z-coordinate.

From the work of Bell (2010), we know that a non-stretchable planar tree is fully immobilized if each node of the tree of which its incident edges do not positively span $\mathbb{R}^2$ is fixed. Now, let us define an *upper string* of a polygon as a maximal sequence of edges of which the extreme vertices are convex vertices of the polygon, and no part of the polygon lies above the edges (see Figure 21(a)). A given polygon P can have multiple upper strings, but has at least one.

For each upper string it holds that at its convex vertices the negative gravity vector points outside the polygon. As these convex vertices are fixed (by a gripper), the entire set of edges the string consists of is immobilized. This can be seen by adding virtual vertical edges fixed in gravity pointing downward from the non-convex vertices, which make sure that the non-convex vertices cannot move upward (per the downward-tendency assumption). The incident edges of the non-convex vertices now positively span $\mathbb{R}^2$, hence the entire string is immobilized.

Now, every point of the polygon P that can be connected to an upper string by a vertical line segment that is fully contained within P is immobilized. This is because this point cannot move downward per the non-stretchability assumption (note that the upper string is immobilized), and it cannot move upward per the downward-tendency assumption. Hence, all such points can be 'removed' from P: they have been proven immobilized. What remains is a smaller polygon P' (potentially consisting of multiple pieces) for which immobilization has not been proven (see Figure 21(b)). The smaller polygon P' has vertical edges that did not belong to the original polygon P. The points on these vertical edges are immobilized, including both incident vertices (of which the upper one may be a non-convex vertex of P that is convex in P'), as they vertically connect to the upper string.

Then, the proof recurses on the new polygon P', of which the convex vertices of the upper string(s) need to be fixed. Note that P' may have convex vertices that were non-convex in P. These need not be fixed, as they were already proven immobilized since they are part of the vertical edge of P'.

This proves the theorem. Note that convex vertices where the negative gravity vector points into the polygon will never be part of an upper string at any phase of the proof, so they need not be fixed. Also, the recursion 'terminates'. This can be seen by considering the vertical trapezoidal decomposition of the original polygon P, which contains a finite number of trapezoids. In each recursion step, at least one trapezoid is removed from P, until the entire polygon has proven immobilized.

Appendix B: Shape models used

B.1. Towels

As there is little inherent structure to a towel, its skeletal model is simply parametrized about the location of its four vertices. Only one constraint was imposed, which is common to all of our models:

- The model contour cannot have any self-intersections.

See Figure 22 for details.

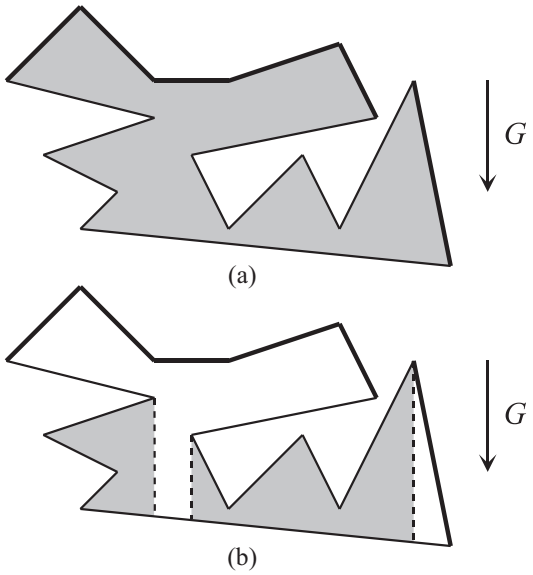

Fig. 21. A polygon with two upper strings shown thick. (b) The white part of the polygon (including the vertical dashed edges) has proven immobilized. The grey part remains.

Fig. 22. A towel model has eight total parameters, corresponding to four skeletal points. These are simply the four corners of the towel.

B.2. Pants

A skeletal model of pants was devised, whose parameters are shown in Figure 23.[8]

We found it was best to give the pants model as much freedom as possible. Therefore, only a small number of constraints were imposed, penalizing extreme deviations from the norm of:[9]

- the length of the legs relative to the height of the pants;
- the width of the legs relative to the width of the pants;
- the width of the pants relative to the height.

For the fitting of pants two different initializations were attempted: the first with the legs virtually straight, and the second with the legs widely spaced. Both models were fit, and the one with the lowest final cost function was chosen.

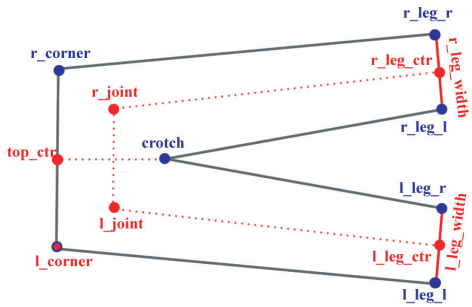

Fig. 23. The pants skeleton is defined by 14 scalar parameters, corresponding to 6 skeletal points, and 2 scalar values, denoting the width of each pant leg. The remaining landmark points are generated as follows: the right corner is an extrapolation of the distance from the left corner to the top center; the crotch is the top center mirrored about the axis spanning the left and right joints; the leg corners are determined by the line perpendicular to the leg axis, at a distance specified by the leg width.

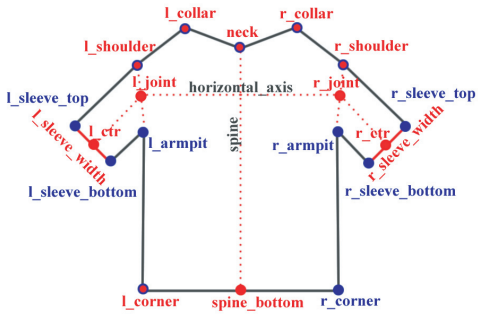

Fig. 24. A short-sleeved shirt skeleton is defined by 24 parameters, corresponding to 11 skeletal points and 2 scalar parameters for sleeve width. The remaining landmark points are generated as follows: the right corner is found by extrapolating the line from the left corner to the spine bottom; the armpit is determined by extrapolating the line from the shoulder to the shoulder joint; the sleeve corners are determined by the line perpendicular to the sleeve axis, at a distance specified by the sleeve width.

B.3. Short-sleeved shirts

A skeletal model of short-sleeved shirts was also used, detailed in Figure 24.

In order to guide the optimization, a number of constraints were imposed, restricting:

- the location of the collar points with respect to the neck and shoulders;
- the location of the shoulders with respect to the armpits;
- the angle between the spine and horizontal axis;
- the relative size and angle of the sleeves;
- the width–height ratios of the sleeves and torso.

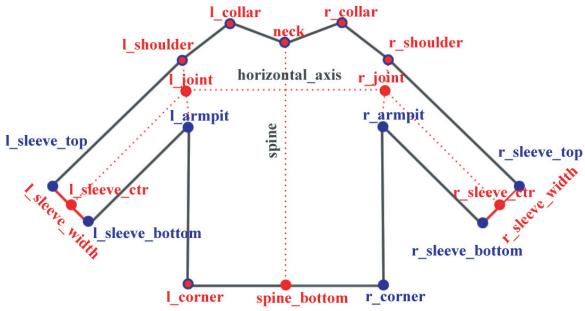

Fig. 25. A long-sleeved shirt skeleton is defined by same parameters as the short-sleeved skeleton.

Two different initializations were attempted: the first with medium-length sleeves, and the second with extremely short sleeves. Both models were run, and the one with the lowest final cost function was chosen.

In addition to the orientation, symmetric, and asymmetric phases of optimization, a fourth fine-tuning phase was run. In this phase, the location of all sleeve vertices were free to move, while the rest remained fixed. This was meant to account for the irregular shape of many T-shirt sleeves.

B.4. Long-sleeved shirts and sweaters

The skeletal model for long-sleeved shirts is detailed in Figure 25.

This model is virtually identical to the short-sleeved model, with a single constraint added:

- Each sleeve must be at least twice as long as it is wide.

Only one initialization was used, with the arms at a downward angle.

As long-sleeved shirts have the potential for drastic asymmetry, both the orientation and symmetric phases of optimization proved to be non-useful, and occasionally damaging, the former settling on erroneous angles, and the latter on vastly incorrect poses. In some cases, the error was so great that the asymmetric phase could not correct for it. For this reason, only the asymmetric phase of optimization was used on this model.

Appendix C: Black box numerical optimization

We employ a simple coordinate descent approach to optimization. For initial input parameters $P \in \mathbb{R}^K$, score function $\mathcal{C}(P)$, and initial step size δ:

$\delta_1, \cdots, \delta_K \leftarrow \delta$
$S \leftarrow \mathcal{C}(P)$
for iter$\leftarrow 1 : 100$ **do**
  for $i \in 1 : K$ **do**
    $P' \leftarrow P$
    $P'_i \leftarrow P'_i + \delta_i$
    $S' \leftarrow \mathcal{C}(P')$
    if $S' > S$ **then**
      $P \leftarrow P'$
      $S \leftarrow S'$
      $\delta_i \leftarrow -\delta_i * 1.5$
    else
      $\delta_i \leftarrow -\delta_i$
      $P' \leftarrow P$
      $P'_i \leftarrow P'_i + \delta_i$
      $S' \leftarrow \mathcal{C}(P')$
      if $S' > S$ **then**
        $P \leftarrow P'$
        $S \leftarrow S'$
        $\delta_i \leftarrow \delta_i * 1.5$
      else
        $\delta_i \leftarrow \delta_i * 0.5$
  if $\max |\delta| < 0.001$ **then**
    break
$P_{\text{out}} \leftarrow P$

with P_{out} the fit parameters.