Folding Deformable Objects using Predictive Simulation and Trajectory Optimization

Yinxiao Li, Yonghao Yue, Danfei Xu, Eitan Grinspun, Peter K. Allen

***Abstract*— Robotic manipulation of deformable objects remains a challenging task. One such task is folding a garment autonomously. Given start and end folding positions, what is an optimal trajectory to move the robotic arm to fold a garment? Certain trajectories will cause the garment to move, creating wrinkles, and gaps, other trajectories will fail altogether. We present a novel solution to find an optimal trajectory that avoids such problematic scenarios. The trajectory is optimized by minimizing a quadratic objective function in an off-line simulator, which includes material properties of the garment and frictional force on the table. The function measures the dissimilarity between a user folded shape and the folded garment in simulation, which is then used as an error measurement to create an optimal trajectory. We demonstrate that our two-arm robot can follow the optimized trajectories, achieving accurate and efficient manipulations of deformable objects.**

I. Introduction

Robotic folding of a garment is a difficult task because it requires sequential manipulations of a highly unconstrained, deformable object. Given the garment shape, the robot can fold it by following a folding plan [14][13]. However, the layout of the same folding action can vary in terms of the material properties such as cloth hardness and the environment such as friction between the garment and the table. Given the starting and ending folding positions, different folding trajectories will lead to different results. In this paper, we propose a novel method that learns optimal folding trajectory parameters from predicted thin shell simulations of similar garments, which can then be applied to a real garment folding task (see Figure 1). The contributions of our paper are:

- A fast and robust algorithm that can detect garment key points such as sleeve ends, collar, and waist corner, automatically. These key points can be used for folding plan generation.
- An online optimization algorithm that learns optimal trajectories for manipulation from mathematical model evolution combined with predictive thin shell simulation.
- A novel approach that adjusts the simulation environment to the robot working environment for the purpose of creating a similar manipulation result.
- The trajectories are general in that they can be scaled to accommodate similar garments of different size.
- Experimental results with a Baxter robot showing successful folding trajectories for a number of different garments including sweaters, pants, and towels.

All the authors are with Department Computer Science, Columbia University, New York, NY, USA {yli@cs., yonghao@cs., dx2143, eitan@cs. allen@cs.}columbia.edu

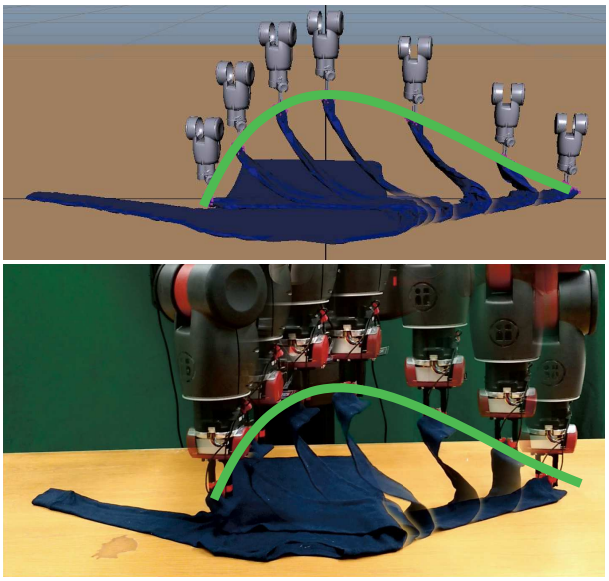

Fig. 1. Comparison of our simulation of robotic manipulation (TOP) and real robot implementation (BOTTOM). The green curves show the virtual and the real trajectories for folding.

Figure 2 shows the complete pipeline of garment manipulation, as well as the key steps of garment folding. The garment folding is the final step of the entire pipeline of garment manipulation which contains grasping, visual recognition, regrasping, unfolding, placing flat, and folding. Our previous work [9][10][11] has successfully addressed all the stages of the pipeline with the exception of the final folding task. This paper specifically addresses the robotic folding task (purple rectangle in Figure 2) with the goal of finding optimal trajectories to successfully fold garments.

We begin by assuming the garment is placed flat on the table initially, as shown in our previous work [11]. By detecting the key points of the garment (see section V-A), a pre-defined folding plan is used to create optimal trajectories for folding the garment. After several steps, we obtain a desired folding result in the real world using the Baxter robot, which is comparable to the result from the simulation.

II. Related Work

There are many challenges associated with the manipulation of a deformable object such as a garment. One of the challenges is unfolding a garment from an arbitrary state and placing it flat on a table. Many researchers start with simple garments such as a towel [6]. By iteratively looking for the lowest corner point of the towel, the robot

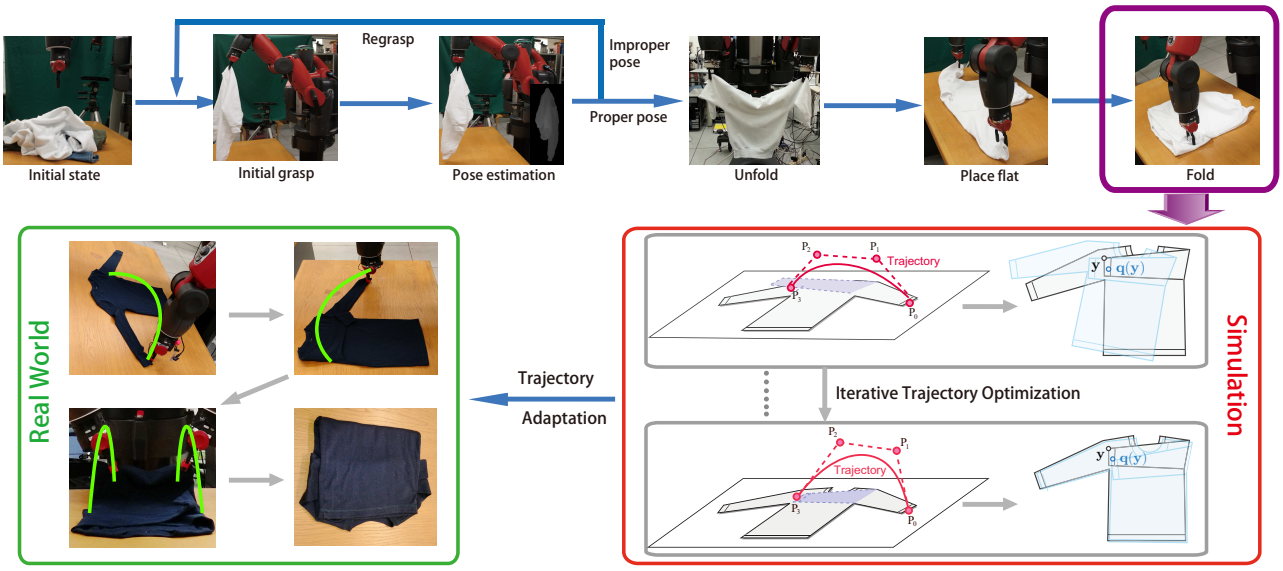

Fig. 2. TOP ROW: The entire pipeline of dexterous manipulation of deformable objects. In this paper, we are focusing on the phases of garment folding, as highlighted in the purple rectangle. BOTTOM ROW: Details of the folding procedure. We apply off-line simulation with iterative trajectory optimization to find the best trajectory for a specific folding action by comparing the result (light blue contour) with template (black contour). Similar steps are repeated until the garment is folded in the simulator. Then all the folding trajectories are exported, adapted, and implemented on a real robot. Green arcs illustrate the actual trajectories of robotic arms.

is able to unfold it and place flat on a table. Then the towel can be quickly folded by symmetric information. For more complicated garments such as sweaters and pants, their states (poses) have to be recognized first by either image-based perception [21][9][4] or 3D shape matching [8][10], etc.. Deformable objects such as a garment have large dimensional state spaces which are hard to track and recognize. Therefore, for such tasks, many researchers employ a large database which contains exemplars of different states of an object from off-line simulation or real garments as the training data. By using SIFT feature [21][9] or volumetric features [4][10], the state of a garment is recognized and tracked. After regrasping several times, the garment can be unfolded and placed flat on a table by a dual-arm robot [3][18][11], a prerequisite for garment folding.

With the garment fully spread on the table, attention is turned to parsing its shape. S. Miller *et al.* have designed a parametrized shape model for unknown garments [14][13]. Each set of parameters defines a certain type of garment such as a sweater or a towel. The contour-based garment shape model was further improved by J. Stria *et al.* using polygonal models [17]. The detected garment contour is matched to a polygonal model by removing non-convex points using a dynamic programming approach. Folding is the ultimate goals of garment manipulation and only a few researchers have achieved this. F. Osawa *et al.* used a robot to fold a garment with a special purpose table that contains a plate that can bend and fold the clothes assisted by a dual-arm robot. Another folding method using a PR2 robot was implemented by J. van den Berg *et al.* [19]. The core of their approach was the geometry reasoning with respect to the cloth model without any physical simulation. Similar work done by J. Stria *et al.* [18] using two industrial arms and a polygonal contour model. Kita, *et al.* [7] use a humanoid robot to fold a garment starting from a random configuration.

None of the previous works focus on trajectory optimization for garment folding, which brings uncertainty to the layout given the same folding plan. One possible case is that the garment shifts on the table during one folding action so that the targeted folding position is also moved. Another case is that an improper folding trajectory causes additional deformation of the garment itself, which can accumulate.

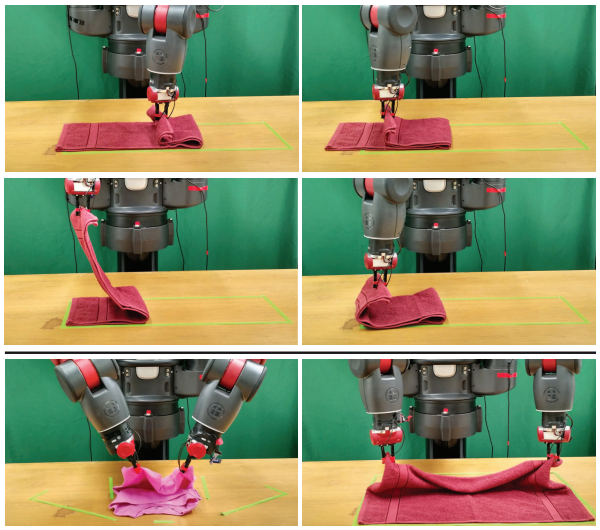

Fig. 3. Failure example with improper folding trajectories. FIRST ROW: Folding trajectory is low and flat that causes drift to the towel and long-sleeve T-Shirt. SECOND ROW: Folding trajectory is too high when the gripper approaching the target folding position that piles up the towel. THIRD ROW: Dual-arm folding. If the distance between the two arms is too close, the folding may fail.

Figure 3 shows a few failure examples with improper trajectories. We use green tape on the table to show the original

position of the garments. The first two rows show that if the moving trajectory is too low and close to the garment, the folded part will fall down, pull the rest, and cause drift of the whole garment. These cases usually happen when the folding step is lengthy without trajectory optimization. The third row shows a case where the folding trajectory is too high, which will cause extra wrinkles or even piling up. The last row shows two cases using two arms to fold. If the arms are close, the part in between loses tension, and will fall down and pull the rest away. The focus of this paper is to create trajectories for folding that will overcome these problems.

III. Simulation Environment

A. Folding Pipeline in Simulation

In the model simulation, we use a physics engine [1] to simulate the movement and deformation of the garment mesh models. We assume there is only one garment for each folding task, which has been placed flat on a table. A virtual table is added to the scene which the garment lies on, as shown in Figure 1, top.

During each folding step, the robot arm picks up a small part of the mesh, moves it to the target position following a computed trajectory, and places it on the table to simulate an entire folding scenario. If the part of the garment to be folded is relatively wide, then both left and right arms may be involved. The trajectory is generated using a Bézier curve, which will be discussed in section IV.

Most of the garment mesh models are built from our test garments. A garment mesh is created by first extracting the contour of the garment (see section V-A). Then by inserting points on the inside of the garment contour, we triangulate a mesh by connecting these points. Lastly, we mirror the mesh to construct a two-sided garment mesh.

B. Parameter Adaptation

There are two key parameters needed to accurately simulate the real world folding environment. The first is the material properties of the fabric, and the second is the frictional forces between the garment and the table.

1) Material properties: Through many experiments, we found that the most important property for the garments in the simulation environment is shear resistance. It specifies the amount the simulated mesh model resists shear under strain; when the garment is picked up and hung by gravity, the total length will be elongated due to the balance between gravity force and shear resistance. An appropriate shear resistance measure allows the simulated mesh to reproduce the same elongation as the real garment. This parameter will bridge the gap between the simulation and the real world for the garment mesh model.

For each garment, we follow the steps described below to measure the shear resistance. Figure 4 shows an example.

- Manually pick one extremum part of the garment such as the sleeve end of a T-shirt, the waist part of a pair of pants, and a corner of a towel.
- Hang the garment under gravity and measure the length between the picking point and the lowest point as L_1
- Slowly put down the garment on a table and keep the picking point and the lowest point in the previous step at maximum spread condition. Measure the distance between these two points again as L_2. The shear resistance fraction is defined as the following

$$shear\_frac = (L_1 - L_2)/L_2 \tag{1}$$

- We then pick up and hang the virtual garment in *Maya*, adjusting the *Maya* shear parameter such that the shear fraction as calculated in the simulator is identical to the real world.

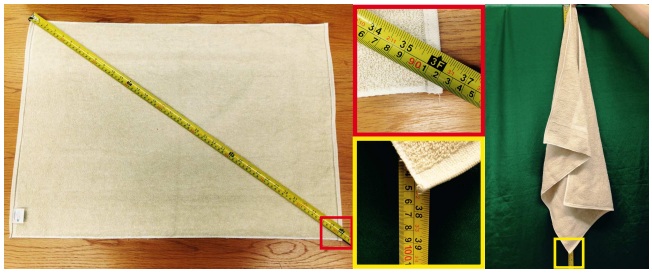

Fig. 4. Method for measuring the shear resistance. LEFT: Diagonal length measurement. MIDDLE: Zoomed in regions. RIGHT: The garment is hanging under gravity.

2) Frictional forces: The surface of the table can be rough if covered by a cloth sheet or slippery if not covered, which leads to variance in friction between the table and garment. A shift of the garment during the folding can possibly impair the whole process and cause additional repositioning. Adjusting the frictional level in the simulation to the real world is crucial and necessary for trajectory optimization.

To measure the friction between the table and the garment, we do the following steps.

- Place a real garment on the real table of length L_t.
- Slowly lift up one side of the real table, until the garment in the real world begins to slide. The lifted height is H_s. The friction angle is computed as,

$$\angle_{Friction} = \sin^{-1}(H_s/L_t) \tag{2}$$

- In the virtual environment, the garment is placed flat on a table with gravity. Assign a relatively high friction value to the virtual table. Lift up one side of the virtual table to the angle of $\angle_{Friction}$.
- Gradually decrease the frictional force in the virtual environment until the garment begins to slide. Use this frictional force in the virtual environment as it mirrors the real world

With these two parameters, we obtain similar manipulation results for both the simulation and the real garment.

IV. Trajectory Optimization

The goal of the folding task is specified by the initial and folded shapes of the garment, and by the starting and target positions of the grasp point (as in Figure 5). Given the simulation parameters, we seek the trajectory that effects the

desired set of folds. We first describe how to optimize the trajectory for a single end effector, and then discuss the case of two end effectors.

A. Trajectory parametrization

We use a Bézier curve [5] to describe the trajectory. An n-th order Bézier curve $\mathbf{T}(u)$ has $(n+1)$ control points $\mathbf{P}_k = (P_{k,x}, P_{k,y}, P_{k,z})^T \in \mathbb{R}^3$, defined by

$$\mathbf{T}(u) = \sum_{k=0}^{n} B_k^n(u)\mathbf{P}_k, \tag{3}$$

where $B_k^n(u) = \binom{n}{k}(1-u)^{n-k}u^k$ are the Bernstein basis functions.

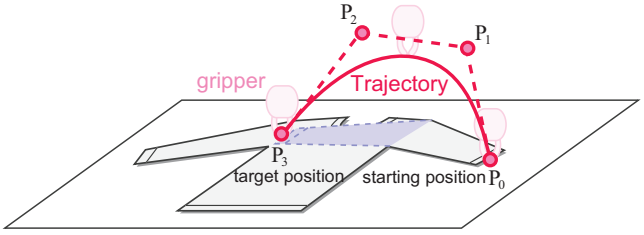

Fig. 5. An example of the folding task: we want to fold a sleeve into the blue target position, by using a robotic gripper to move the tip of the sleeve (grasp point) from the starting position ($\mathbf{P}_0$) to the target position ($\mathbf{P}_3$), following a trajectory, shown as the red curve. $\mathbf{P}_1$ and $\mathbf{P}_2$ are knot points that form the Bézier trapezoid.

We use $n = 3$ for simplicity, but our method can be easily extended to deal with higher order curves. $\mathbf{P}_0$ and $\mathbf{P}_3$ are fixed to the specified starting and target positions of the grasp point (as in Figure 5). The intermediate control points $\mathbf{x} = (\mathbf{P}_1^T, \mathbf{P}_2^T)^T$ can then be adjusted to define a new trajectory using the objective function defined below. The update rule is described in section IV-B.

$$\mathbf{x}_{opt} = \underset{\mathbf{x}}{\operatorname{argmin}} \{\underbrace{l_{\mathbf{x}} + \alpha D(\mathcal{S}_t, \mathcal{S}_{\mathbf{x}})}_{C(\mathbf{x})}\}^2. \tag{4}$$

Here $C(\mathbf{x})$ is a cost function with two terms. The first term penalizes the trajectory length $l_{\mathbf{x}}$, thus preferring a folding path that is efficient in time and energy. The second term seeks the desired fold, by penalizing dissimilarity $D(\mathcal{S}_t, \mathcal{S}_{\mathbf{x}})$ between the desired folded shape $\mathcal{S}_t$, compared to the shape $\mathcal{S}_{\mathbf{x}}$ obtained by the candidate folding trajectory $\mathbf{x}$, as predicted by a cloth simulation; we used a physical simulation engine [1], for the cloth simulation. The weight α balances the two terms; we used $\alpha = 10^3$ in our experiment. See section IV-B for optimization details.

Intuitively, dissimilarity measures the difference between the desired folded shape and the folded garment in simulation. We define the dissimilarity term as

$$D(\mathcal{S}_t, \mathcal{S}_{\mathbf{x}}) = \frac{1}{|\mathcal{S}_t|}\int_{\mathcal{S}_t} \|\mathbf{q}(\mathbf{y}) - \mathbf{y}\| dA, \tag{5}$$

where $|\mathcal{S}_t|$ is the total surface area of the garment mesh including both sides of the garment, $\mathbf{y} \in \mathcal{S}_t$ is a point on the target folded shape $\mathcal{S}_t$, $\mathbf{q}(\mathbf{y}) \in \mathcal{S}_{\mathbf{x}}$ is the corresponding point on the simulated folded shape, and dA is the area measure, see Figure 6. Our implementation assumes $\mathcal{S}_t$ and $\mathcal{S}_{\mathbf{x}}$ are given as triangle meshes, and discretizes (5) as

$$\tilde{D}(\mathcal{S}_t, \mathcal{S}_{\mathbf{x}}) = \frac{1}{|\mathcal{S}_t|}\sum_i \|\mathbf{q}_i - \mathbf{y}_i\| A_i, \tag{6}$$

where $\mathbf{y}_i$ is the barycenter of i-th triangle on the target shape, $\mathbf{q}_i$ is the (corresponding) barycenter of i-th triangle on the simulated shape, and A_i is the area of the i-th triangle on the target shape.

To compute the trajectory length $l_{\mathbf{x}}$, we use the De Casteljau's algorithm [5] to recursively subdivide the Bézier curve $\mathbf{T}$ into a set of Bézier curves $\mathbf{T}^{(j)}$, until the deviation between the chord length ($\|\mathbf{P}_0^{(j)} - \mathbf{P}_3^{(j)}\|$) and the total length between the control points ($\sum_{i=0}^{2} \|\mathbf{P}_i^{(j)} - \mathbf{P}_{i+1}^{(j)}\|$) for each subdivided curve $\mathbf{T}^{(j)}$ is sufficiently small. Then, $l_{\mathbf{x}}$ is approximated by summing up the chord lengths of all the subdivided curves: $l_{\mathbf{x}} \approx \sum_j \|\mathbf{P}_0^{(j)} - \mathbf{P}_3^{(j)}\|$.

We initialize $\mathbf{P}_1$ and $\mathbf{P}_2$ as

$$\mathbf{P}_1 = \frac{2}{3}\mathbf{P}_0 + \frac{1}{3}\mathbf{P}_3 + h\|\mathbf{P}_0 - \mathbf{P}_3\|\mathbf{e}_v, \tag{7}$$

$$\mathbf{P}_2 = \frac{1}{3}\mathbf{P}_0 + \frac{2}{3}\mathbf{P}_3 + h\|\mathbf{P}_0 - \mathbf{P}_3\|\mathbf{e}_v, \tag{8}$$

where $\mathbf{e}_v$ is the unit vector in the upward vertical direction, h is a constant value of $1/3$, which means the initial trajectory will have equal horizontal extent between knot points.

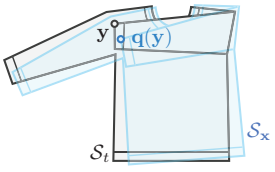

Fig. 6. The dissimilarity captures the misalignment between $\mathcal{S}_t$ and $\mathcal{S}_{\mathbf{x}}$ by integrating the distance between the corresponding points $\mathbf{y} \in \mathcal{S}_t$ and $\mathbf{q}(\mathbf{y}) \in \mathcal{S}_{\mathbf{x}}$ over the garment.

B. Optimization.

To optimize equation (4), we apply a secant version of the Levenberg-Marquardt algorithm [12][15]. For the current trajectory generated by $\mathbf{x}$, we estimate the derivative $\nabla C(\mathbf{x})$ of the cost function $C(\mathbf{x})$ numerically, by sampling slightly modified trajectories $\mathbf{x} + \delta\mathbf{e}_j$, where $\mathbf{e}_j, 1 \le j \le \dim(\mathbf{x})$, are the orthonormal bases, and we used $\delta = 10^{-1}$ in our implementation.

The secant version of Levenberg-Marquardt algorithm iteratively builds a local quadratic approximation of $\{C(\mathbf{x})\}^2$ based on the numerical derivative, and then takes a step toward an improved state. The direction of the step is a combination of the steepest gradient descent direction and the conjugate gradient direction. We use the specific approach described by Madsen et al. [12] (see §3.5 therein). The iterative procedure terminates when the improvement in $\{C(\mathbf{x})\}^2$ becomes sufficiently small.

C. Multiple arms.

In the case of using multiple arms, we associate an individual trajectory $\mathbf{x}_i$ to each of the arms R_i. We then extend the state variable to $\mathbf{x} = (\mathbf{x}_1^T, ...)^T$. The rest of the optimization procedure is the same as the single arm case.

Note that both single and dual-arm trajectories are in 3D space. The optimization for dual-arm trajectories is able to find a solution which will overcome failures such as shown in Figure 3 bottom.

V. Application to Garment Folding

A. Key Points Localization

We assume the garment is placed roughly in the center of the table, as shown in the Figure 7, bottom. Our first step is to segment the garment from the background. Since we can easily obtain a table or a surface with homogeneous color [14][17], a fast color-based supervised image segmentation method is suitable for our task. We apply a marker-based watershed method [20] with the four corners of the table initialized as background and the center initialized as the foreground. More complicated scenes can employ advanced image segmentation algorithms such as GrabCut [16].

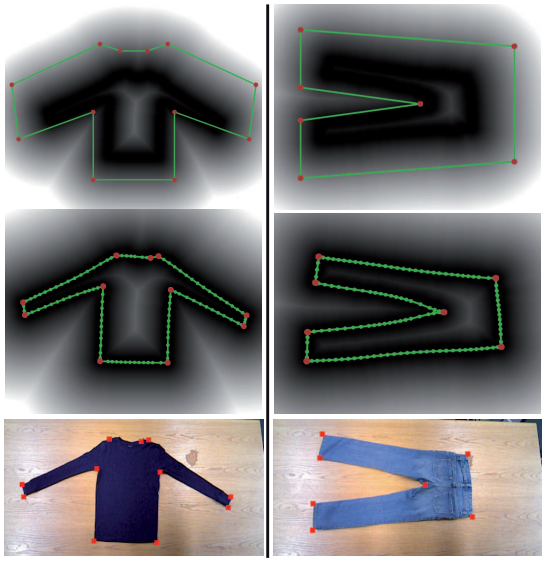

Fig. 7. Red dots are the predefined key points for the garment, such as sleeve ends. Left column is a long-sleeve t-shirt example, and the right column is a pants example. TOP: Initialized contour. Each garment category is initialized with a different contour. MIDDLE: Fitting results. The contour shrinks onto the boundary of the garment according to the distance field. BOTTOM: Fitting results mapped back to the original input image.

To register the feature points, such as the corner points of sleeves, we employ a 2D registration technique to register a pre-defined garment template (as in Figure 7 top row) with the captured garment mask. Our 2D registration is based on our 3D non-rigid registration code for thin shell models [11], and can deal with garment masks that have curved contours.

We first initialize a distance field using the segmented mask from the previous step. The category of the garment can be easily recognized by using a template matching algorithm, which leads to an associated garment initial template. We register the template with the garment mask by minimizing the following energy function:

$$E_T(S,\bar{S}) = E_{\text{fit}}(S,T) + \kappa E_{\text{stretch}}(\bar{S},\mathbf{x}) + \beta E_{\text{bend}}(\bar{S},\mathbf{x}), \tag{9}$$

where T is the target garment mask, S is the current deformed 2D contour, and $\bar{S}$ is a reference 2D contour, which is from the previous state. $E_{\text{fit}}(S,T)$ penalizes discrepancies between the contour and the target mask. The last two terms seek to limit and regularize the deformation of the contour in order to preserve the angle features, which includes the stretching and bending energies, weighted by user specified coefficients κ and β.

We represent the contour as a closed loop consisting of line segments. In this discrete representation, E_{fit} is computed as

$$\tilde{E}_{\text{fit}}(S,T) = \sum_i \left(\text{dist}_T(\mathbf{x}_i)\right)^2 l_i, \tag{10}$$

where l_i and $\mathbf{x}_i$ are the length and the midpoint of the i-th line segment of the deformed contour. E_{stretch} is computed as

$$\tilde{E}_{\text{stretch}}(\bar{S},\mathbf{x}) = \frac{1}{2}\sum_i \left(\frac{l_i}{\bar{l}_i} - 1\right)^2 \bar{l}_i, \tag{11}$$

where $\bar{l}_i$ is length of the i-th line segment of the reference contour. E_{bend} is computed as

$$\tilde{E}_{\text{bend}}(\bar{S},\mathbf{x}) = \frac{1}{2}\sum_j \left(\frac{\theta_j}{\bar{\theta}_j} - 1\right)^2 \bar{L}_j, \tag{12}$$

where θ_j and $\bar{\theta}_j$ are the angles between the adjacent line segments at j-th vertex of the deformed and the reference contours, respectively. $\bar{L}_j$ are the average length of the adjacent line segments at j-th vertex of the reference contour.

Our registration iteratively updates S by using the secant version of the Levenberg-Marquardt algorithm [12][15]. Initially, each vertex in the template polygon is marked as a *feature point* (red points in Figure 7, top row), and is assigned a unique ID to identify its semantic meaning. At the end of each step, we subdivide each contour segment if its length is larger than a threshold, and mark the newly added vertex as *non-feature points* (green points in Figure 7, middle row). Next, we merge any pair of adjacent segments if they do not share a feature point and their length is below a threshold. The subdivide and merge operations guarantee that the garment contour is sufficiently but not overly sampled. Then, we update the reference contour $\bar{S}$ by S.

We repeat the iteration until the reduction in $E_T(S,\bar{S})$ becomes sufficiently small. Finally, the positions as well as the semantic meanings of the feature points in the garment mask are identified by retrieving the feature points via the unique point ID in the registered contour.

VI. Experimental Results

To evaluate our results, we tested our method on several different garments such as long-sleeve t-shirts, pants, and towels for multiple trials, as shown in Figure 8 left. These garments require both single and dual-arm folds. A high resolution video of our experimental results is online at `http://www.cs.columbia.edu/~yli/IROS2015`.

A. Robot Setup

In our experiments, we use a Baxter research robot, which is equipped with two arms with seven degrees of freedom. We mount a Prime Sense Xtion range sensor [2] on top of the Baxter head panel, which has been calibrated to the robot base frame.To improve grasp stability and form a closed loop controller, we add tactile sensors to the grippers.

B. Measurement of parameters

To make the off-line simulation better approximate the real scenario, as described in Section III, we manually measure the stretch resistance of each garment and friction on the table. Figure 8, left shows a picture of all the test garments we used in different colors, sizes, and material properties. Figure 8, right table shows the measured parameters of each test garment, including stretch percentage and Friction angle, and corresponding *Maya* parameters. For common garments, these parameters do not have a significant variance. Therefore, we suggest that if researchers use simulators such as *Maya*, the average values of each column are a reasonably good initialization.

C. Garment manipulation and folding

Figure 9 shows three successful folding examples from the simulation and the real world, including a long-sleeve shirt, a pair of pants, and a medium size towel. We show six key frames for each folding task. The folding poses from the simulation are in the first row of each group with an optimized trajectory. We also show corresponding results from the real world. The green tape contour on the table indicates the original position of the garment.

Each garment is first segmented from the background and key points are detected from the binary mask, which takes $3-5$ secs on a regular CPU. The algorithm discussed in Sec. V-A. Given the key points, a corresponding multi-step folding plan is created For each garment, we have optimized trajectories for each folding step. Here, we map these optimized trajectories to our scenario according to the generated folding plan. Then the Baxter robot follows the folding plan with optimized trajectories to fold the garment. We can see that the deformation of the real garment and the simulated garment is very similar. Therefore, the final folding outcome is comparable to the simulation.

Table I shows statistical results of the garment folding test. Each time one or two robotic arms fold the garment counts as one fold. We run 10 trials for each test garment. It turns out that the folding performance of the Long-Sleeve T-Shirts and Towels are very stable with our optimized trajectories. Jeans and pants are less stable because the shear resistance of the surface is relatively high, and sometimes is difficult to bend, leading to unsuccessful folding. In the successful folding cases for jeans and pants, we sometimes ended up with small wrinkles, but the folding plan was still able to complete successfully. We also show the average time to fold a garment in the last row. The robot is able to fold most garments in about 1.5 minutes.

Garment Type	# of folds	Success Rate	Avg. Time (sec)
L-S T-Shirt (large)	3	$10/10$	121
L-S T-Shirt (small)	3	$10/10$	118
Jeans	2	$7/10$	88
Pants	2	$8/10$	88
Large Towel	2	$10/10$	90
Medium Towel	2	$10/10$	88
Small Towel	2	$10/10$	83
Average	**2.3**	$\mathbf{9.3/10}$	**97**

TABLE I. Results of folding test for each garment . We show the number of folding steps, successful rate, and total time of each garment. Each garment has been tested 10 times. L-S stands for Long-Sleeve. The time is the average over all successful trials for each garment.

There is a trade-off between doing contour fitting at each step and total time spent to fold a garment. In this work, we start with one template and then assume that each step after that the folded garment is close to that in the simulation. Our results in Table I verify that this method works well and is able to save time since we only do the contour fitting once. With our simulated trajectories, the Baxter robot is able to fold a garment under predefined steps correctly. An alternative method could use the contour fitting at each step but this would require more time and computation.

We note that some failures due to the motor control error from the Baxter robot. When the robot executes an optimized trajectory, its arm suffers from a sudden drop or jitter. Such actions will raise pull forces to the garment, leading to drift and inaccurate folding. This can be solved by using an industrial level robotic arm with more accurate control. We also note that failures can be recognized with the correct sensing suite, and we are currently investigating ways to effect online error recovery for such failures. One difference between the simulation and the real world we found is that moving a point on the mesh in the simulation is different from using a gripper to grasp a small area of a real garment and move it. In the future, we hope to be able to simulate a similar grasp effect for the trajectory optimization.

VII. Conclusion

In this paper, we propose a novel solution to find an optimal trajectory for manipulation of deformable garments. We first create a simulation environment that is comparable to the real world. By minimizing a quadratic objective function that measures dissimilarity between simulated folded shape and user specified shape iteratively, we obtain an optimized trajectory. The trajectory is then mapped to a real robot and executed accordingly. Experimental results demonstrate that with our optimized trajectories, the Baxter robot can manipulate the garment efficiently and accurately.

Acknowledgments We'd like to thank J. Weisz and J. Varley, for many discussions. We'd also like to thank NVidia Corporation, Intel Corporation, and Takktile LLC for the hardware support. This material is based upon work supported by the National Science Foundation under Grant No. 1217904 and in part by the JSPS Postdoctoral Fellowships for Research Abroad.

References

[1] Maya, http://www.autodesk.com/products/autodesk-maya/.

Garment Type	Stretch (%)	Friction Angle (°)	Maya Shear Resistance	Maya Friction
Long-Sleeve T-Shirt (large)	2.9	24.3	200	0.7
Long-Sleeve T-Shirt (small)	2.9	24.7	200	0.7
Jeans	2.9	19.1	200	0.5
Pants	1.7	21.9	340	0.6
Large Towel	2.2	18.7	260	0.5
Medium Towel	3.1	22.3	190	0.6
Small Towel	1.1	24.3	530	0.7
Average	**2.4**	**22.2**	**274**	**0.6**

Fig. 8. LEFT: A picture of our test garments. RIGHT: Results for each unfolding test on the garments. We show the results of stretch percentage, Friction angle of the table, and the corresponding parameters in Maya by each test. The last row shows the average of each measurement component.

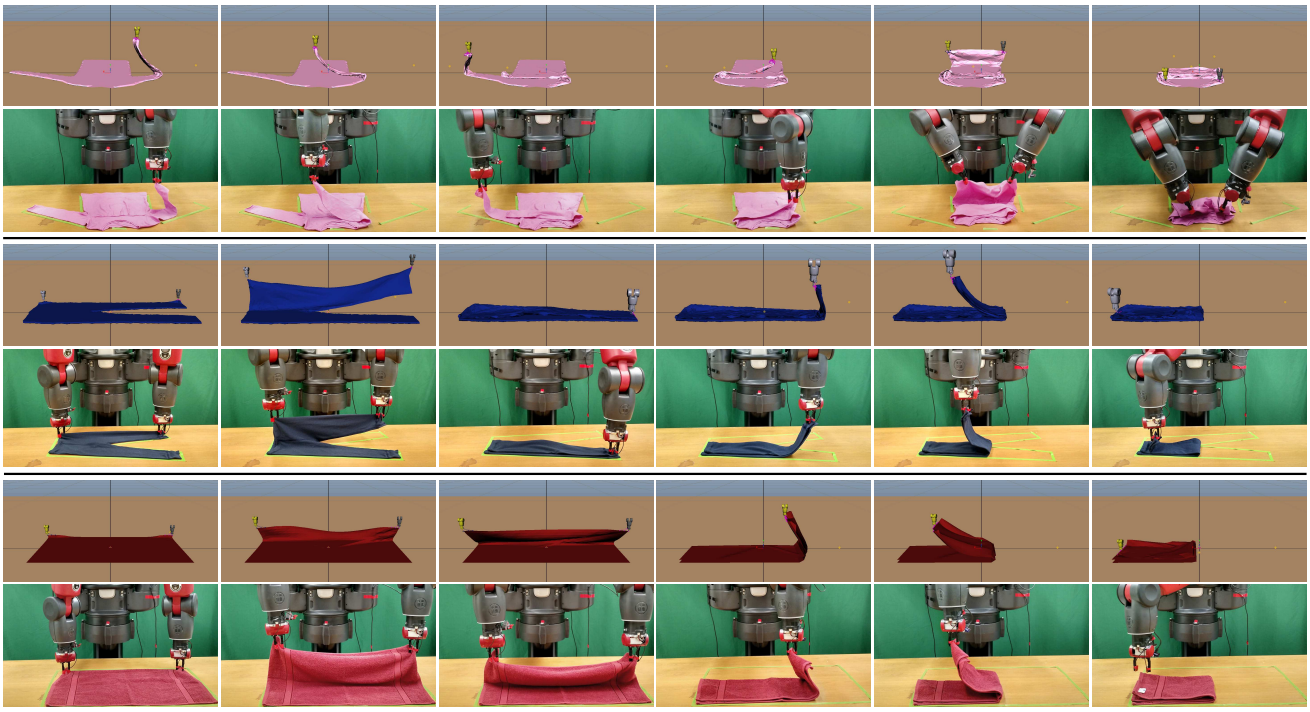

Fig. 9. Successful folding examples with optimized folding trajectories from off-line simulation. The first row of each group is from the simulation and the second row is from the real world (Green tape shows the original garment contour position). TOP GROUP: Long-sleeve shirt folding with 3 steps. MIDDLE GROUP: Long pants folding with 2 steps. BOTTOM GROUP: Medium size towel folding with 2 steps.

[2] Prime Sense, http://en.wikipedia.org/wiki/PrimeSense/.

[3] M. Cusumano-Towner, A. Singh, S. Miller, J. F. OBrien, and P. Abbeel. Bringing clothing into desired configurations with limited perception. In *Proc. ICRA*, 2011.

[4] A. Doumanoglou, T-K Kim, X. Zhao, and S. Malassiotis. Active random forests: An application to autonomous unfolding of clothes. In *Proc. ECCV*, 2014.

[5] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 1988.

[6] J. Lei J. Maitin-Shepard, M. Cusumano-Towner and P. Abbeel. Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *Proc. ICRA*, 2010.

[7] Y. Kita, F. Kanehiro, T. Ueshiba, and N. Kita. Strategy for folding clothing on the basis of deformable models. 2014.

[8] Y. Kita, T. Ueshiba, E-S Neo, and N. Kita. Clothes state recognition using 3d observed data. In *Proc. ICRA*, 2011.

[9] Y. Li, C-F Chen, and P. K. Allen. Recognition of deformable object category and pose. In *Proc. ICRA*, June 2014.

[10] Y. Li, Y. Wang, M. Case, S-F Chang, and P. K. Allen. Real-time pose estimation of deformable objects using a volumetric approach. In *Proc. IROS*, September 2014.

[11] Y. Li, D. Xu, Y. Yue, Y. Wang, S-F Chang, E. Grinspun, and P. K. Allen. Recognition, regrasping, and unfolding of deformable objects using predictive thin shell modeling. In *Proc. ICRA*, May 2015.

[12] K. Madsen, H. B. Nielsen, and O. Tingleff. Methods for non-linear least squares problems (2nd ed.). Technical report, Technical University of Denmark, 2004.

[13] S. Miller, J. Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel. A geometric approach to robotic laundry folding. *IJRR*, 2012.

[14] S. Miller, M. Fritz, T. Darrell, and P. Abbeel. Parametrized shape models for clothing. In *Proc. ICRA*, Sept. 2011.

[15] J. Nocedal and S. Wright. *Numerical Optimization Second Edition*. Springer, 2006.

[16] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, August 2004.

[17] J. Stria, D. Prusa, and V. Hlavac. Polygonal models for clothing. In *Proc. Towards Autonomous Robotic Systems*, 2014.

[18] J. Stria, D. Prusa, V. Hlavac, L. Wagner, V. Petrik, P. Krsek, and V. Smutny. Garment perception and its folding using a dual-arm robot. In *Proc. IROS*, Sept. 2014.

[19] J. van den Berg, S. Miller, K. Goldberg, and P. Abbeel. Gravity-based robotic cloth folding. In *Proc. Intl. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2010.

[20] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.

[21] B. Willimon, I. Walker, and S. Birchfield. A new approach to clothing classification using mid-level layers. In *Proc. ICRA*, 2013.