Learning Arbitrary-Goal Fabric Folding with One Hour of Real Robot Experience

Robert Lee
Queensland University of Technology
r21.lee@hdr.qut.edu.au

Daniel Ward
The University of Queensland
daniel.ward1@uq.net.au

Vibhavari Dasagi
Queensland University of Technology
vibhavari.dasagi@hdr.qut.edu.au

Akansel Cosgun
Monash University
akansel.cosgun@monash.edu

Jürgen Leitner
LYRO Robotics Pty Ltd
juxi@lyro.io

Peter Corke
Queensland University of Technology
peter.corke@qut.edu.au

Abstract: Manipulating deformable objects, such as fabric, is a long standing problem in robotics, with state estimation and control posing a significant challenge for traditional methods. In this paper, we show that it is possible to learn fabric folding skills in only an hour of self-supervised real robot experience, without human supervision or simulation. Our approach relies on fully convolutional networks and the manipulation of visual inputs to exploit learned features, allowing us to create an expressive goal-conditioned pick and place policy that can be trained efficiently with real world robot data only. Folding skills are learned with only a sparse reward function and thus do not require reward function engineering, merely an image of the goal configuration. We demonstrate our method on a set of towel-folding tasks, and show that our approach is able to discover sequential folding strategies, purely from trial-and-error. We achieve state-of-the-art results without the need for demonstrations or simulation, used in prior approaches. Videos available at: https://sites.google.com/view/learningtofold

1 Introduction

The ability to interact with objects is crucial for building robotic systems that can work in the real world. The field of robotic manipulation has progressed significantly in recent years, in particular for tasks that require visual input, such as pick and place [1], grasping [2], tossing [3], and dexterous manipulation [4]. A major contributing factor is the success of learning-based approaches, in no small part due to the availability of large datasets, and the progression of computation and simulation. Despite recent advances, manipulation of deformable objects remains challenging as limited data exists for real-world interaction with deformable objects. Sim-to-real transfer is a promising direction for learning to solve complex robot tasks, yet requires a fast and accurate (enough) simulator. While there is progress, simulations of interactions with deformable objects are still slow and computationally expensive to create, due their incredibly complex dynamics.

In this paper we present a method for a robot to learn to manipulate deformable objects directly in the real world, without reward function engineering, human supervision, human demonstration or simulation. We formulate the task of folding a piece of fabric into a goal configuration (Fig. 1) as a reinforcement learning (RL) problem with top-down, discrete folding actions. We develop a method that is sample efficient, can learn effectively from sparse rewards, can achieve arbitrary unseen goal configurations given by a single image at test time. We also propose a novel self-supervised learning approach, in which the robot collects interaction data in the real world without human interaction. In particular, the contributions of this paper are:

- A fully-convolutional, deep Q-learning approach which leverages discretized folding distances for sample-efficient real world learning.
- A self-supervised learning pipeline for learning fabric manipulation via autonomous data collection by the robot in the real world.

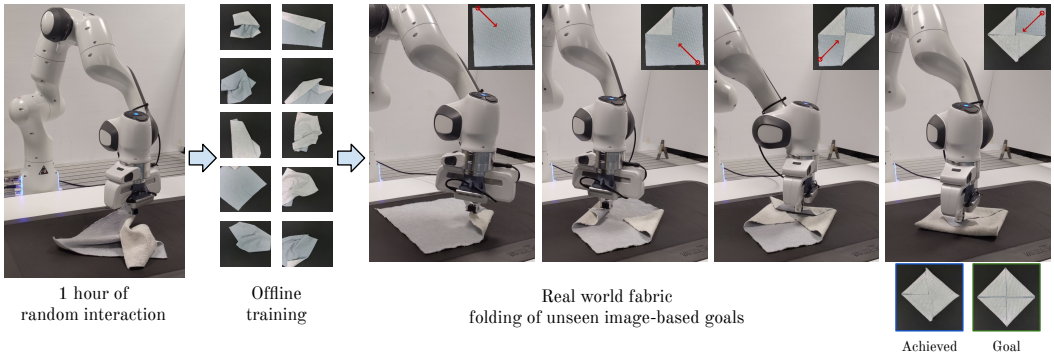

Figure 1: The full fabric folding system. First, we collect an hour of random, self-supervised experience on the real robot, which is used to train a fully-convolutional neural network agent. Then, we deploy the trained agent on the robot with goals that were unseen during data collection and training.

2 Related Work

Deformable object manipulation is a challenging area in robotics research [5, 6, 7, 8, 9, 10, 11]. Fabric is a specific case that is of particular interest to robotics research, with a wide variety of real world applications. While specially designed tools for cloth manipulation exist [12], in this work we are interested in manipulation with robot arms without significant modification to the end-effector.

Cusumano *et al.*[13], using a dual arm robot, demonstrated an approach that manipulates clothes to desired configurations via a Hidden Markov Model and a mesh simulation. While the previous paper used a continuous action state space, others used discrete actions [14, 15, 16]. Similarly, our work proposes a discrete action space for fabric manipulation, making learning behaviours more sample-efficient. Li *et al.*[17] optimized folding trajectories in simulation given predefined grasp and drop points in order to perform real world folding. In this work however, we aim to learn the optimal grasp location and fold vector in order to reach a given goal image, directly in the real world.

Recently, learning-based methods have dominated research in fabric manipulation, often using either human demonstrations [18, 19, 20], sim-to-real transfer [21, 22, 23, 24, 25] or both [26]. Human demonstrations have been shown to improve performance in fabric manipulation tasks [26, 20]. Lasley *et al.*[27] used imitation learning for a bed-making task, where they showed that the trained policy had a comparable performance to the supervisor and outperformed a heuristic method based on contour detection. Jangir *et al.*[20] learned goal-conditioned RL policies in simulation with human demonstrations, utilizing ground truth state information, which is difficult to obtain in the real world. Instead, we propose to learn folding policies from real world vision only. Matas *et al.*[26] used an end-to-end deep RL approach with human demonstrations, relying on domain randomization for sim-to-real transfer. In ablation studies, demonstrations were shown to be crucial.

Effective use of simulation can enable the learning of policies without human demonstrations [21, 22, 23, 24, 25]. Wu *et al.* [21] learned continuous, folding policies in simulation without demonstration, by modelling the conditional relationship between pick and place actions. As learning with this continuous action space in the real world is prohibitively expensive, we discretely model the action space to enable sample efficient learning of complex folding tasks in the real world.

Seita *et al.*[22] framed the fabric flattening problem as imitation learning where an algorithmic supervisor provides data in the form of paired observations. They showed that using RGB as input transferred better than using only depth images. Our work utilizes only RGB images. Ganapathi *et al.*[23] train a dense object descriptors model in simulation and showed successful transfer to two different robots in multi-step fabric smoothing and folding tasks. They collected task-agnostic data similar to our work, however their data was collected in simulation while we collect data directly on the real robot. Furthermore, unlike our approach, they utilize human demonstrations to guide the policy through intermediate goals to solve sequential folding tasks.

Hoque *et al.*[24] learned an image prediction model in simulation, which can be used to solve arbitrary goals at test time via Model Predictive Control (MPC). They apply domain randomization to transfer fabric smoothing policies to a real-world surgical robot. However, their approach fails to transfer folding tasks successfully due to the sim-to-real gap. Yan *et al.*[25] also used MPC along with model-based RL to transfer simulated policies to a real robot for cloth smoothing tasks. Both previous papers learn these models from offline, random actions in simulation. Our approach similarly uses random actions for data collection, but on a real robot to learn arbitrary folding tasks.

3 Problem Formulation

We propose to formulate the fabric manipulation problem as a Markov Decision Process (MDP), where an agent interacts with an environment by choosing an action a_t from state s_t according to the policy $\pi(s_t)$ from the full set of possible actions a'. The environment then transitions to a new state s_{t+1} and the agent receives a reward according to reward function $R_{a_t}(s_t, s_{t+1})$. Reinforcement learning aims to obtain a policy that maximizes the expected sum of future rewards. We learn a state-action value function $Q(s_t, a_t)$, which predicts the expected sum of discounted future rewards after taking action a_t from state s_t. We can optimize this function via the typical Q-learning formulation, i.e. minimizing the temporal difference error between $Q(s_t, a_t)$ and a target value y_t, estimated from data collected by the agent:

$$y_t = R_{a_t}(s_t, s_{t+1}) + \gamma Q(s_{t+1}, \underset{a'}{\operatorname{argmax}}(Q(s_{t+1}, a'))) \tag{1}$$

Where γ is the discount factor. We evaluate the greedy policy by choosing the action that maximises the predicted state-action value:

$$a_t = \underset{a'}{\operatorname{argmax}}(Q(s_{t+1}, a')) \tag{2}$$

4 Approach

In this section we describe our method for learning fabric folding skills directly in the real world, with no reward function engineering, supervision, demonstration or simulation. In order to achieve this, we develop a method that is sample efficient, can learn effectively from sparse rewards, can achieve arbitrary unseen goal configurations given by a single image at test time, and requires no human interaction during data collection.

Our approach involves a two-stage process: data collection and offline training. First, the specific details of our algorithm are described in Sections 4.1 and 4.2. Then, in sections 4.3 and 4.4 the execution of our algorithm, data collection and the training procedure respectively, is described.

4.1 Action and Observation Spaces

The effective workspace of the robot is captured by a downward-facing wrist-mounted camera, allowing the agent to observe a perspective RGB image at each timestep. The choice of action space is crucial for learning efficiency owing to the complex dynamics involved in fabric manipulation. We use a discrete action space that consists of pixel grasp locations and predefined sets of fold angles and fold distances. The discrete action space, combined with our fully convolutional state-action value function, enables us to learn folding behaviour without simulation and a limited amount of offline data from a real robot.

4.2 State-Action Value Function

We parameterize our action-value function, Q_θ, as a feed-forward fully convolutional network, inspired by Zeng *et al.* [28, 3]. The network is trained to evaluate grasping and fixed distance pushing actions without knowledge of the input orientation. We rotate the input images to achieve a discrete set of actions of different direction or orientation. This, in combination with the weight sharing properties and spatial preservation of fully convolutional networks, makes our approach highly sample efficient. We apply this approach to fabric manipulation, while expanding the action space to variable distance folds, by exploiting the proportionality of scale when folding fabric.

The input to the network is an RGB image, of resolution $H \times W$, of the state, channel-wise concatenated with the goal image. The network's output is a heatmap of the same resolution, $H \times W$, and represents the Q-value for a unit action performed at each pixel in the input image. We rotate the image to a canonical orientation prior to estimating the heatmap. Unit actions exist in this canonical orientation. In this way, we can generate heatmaps for arbitrary orientations by transforming the input image, and thus cover the full range of rotation by discretizing into a k fold angle bins.

In addition to action directions, a variable folding distance between pick and place is required in order to perform a range of folding tasks. We make the assumption that the visual appearance of folds in fabric scale proportionally with the length of the fold action. To address this, we extend the prior approach by also scaling the input image by s different scaling factors, which transforms it into the reference frame of the unit action, resulting in a heatmap for actions performed with distance relative to that image. The unit action produced by the network is then scaled by the reverse of the

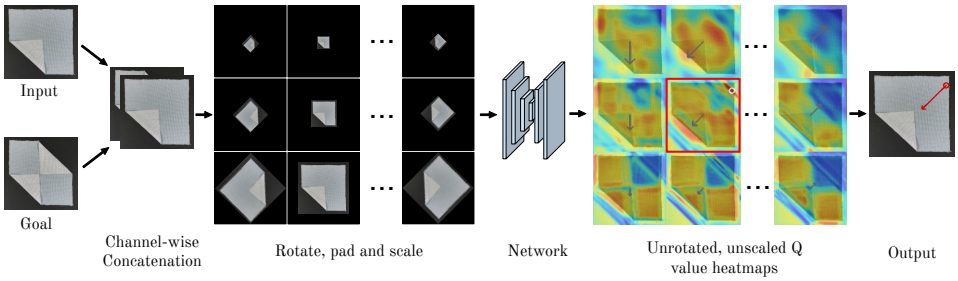

Figure 2: Overview of our approach. We create an expressive fabric folding policy by manipulating the input image. We scale and rotate the image and pick the max value over the output heatmaps

image scaling. Scaling the input by half results in the executed fold covering twice the distance. This relationship is visualized in Figure 3 and given by:

$$D_a = \frac{1}{\beta} D_u \tag{3}$$

Where D_a is the resulting fold distance, β is the image scale factor, and D_u is the unit fold distance.

In summary, to retrieve the next action from this pipeline, we define a fixed number of rotation bins (k) and distance bins (s). As shown in Figure 2, the input image is rotated and scaled, and are passed through the network. The action is selected using Equation 2, where we find the pixel location, rotation and scaling that corresponds to the maximum Q value for the given observation.

4.3 Random Experience Data Collection

We train our policy on a data-set of real robot experience collected from random interactions. Actions are selected by choosing a pixel on the fabric uniformly at random, as well as a discrete distance to fold. We bias the direction of our random actions to move towards the centre of the workspace. However as the pixel location and distance is chosen uniformly at random, the actions do not always result in moving the fabric directly to the centre; it will often move past the centre to the other side of the workspace. Therefore, this bias prevents the random actions from pulling the fabric out of view, and continually moves the fabric around the workspace. We note that it does not bias the training data, as the network itself does not observe the angle of the fold, but rather the image that is rotated to the corresponding angle.

During data collection, an automatic reset is performed by the robot every T timesteps to obtain a wider distribution of experience. This involves grasping the centre of the fabric and dropping it from a fixed height. Randomness is introduced as the fabric lands differently each time.

We collect experience at every timestep t in tuples of observation o_t, action a_t and the next observation o_{t+1}. This data is then used for offline training of the Q-network.

4.4 Offline Training

We use Batch RL for this work: once the robot has collected the dataset of random experience, we utilize this data to train our Q-network in an offline manner [29, 30, 31, 32]. In order to learn

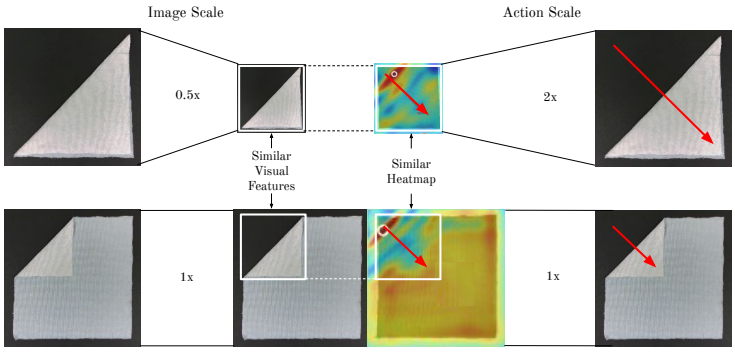

Figure 3: Image scaling is utilized to produce varying fold distances, using a single trained CNN. Note the visual similarity of the regions of the image highlighted in white. The fold on the left is twice the size of that on the right, and so when scaled they appear visually similar. Thus, in the downscaled image on the left, the heatmap produced closely resembles the corresponding region of the heatmap on the bottom right.

a goal conditioned Q-function from sparse rewards, we utilize Hindsight Experience Replay [33]. When computing the target Q value (Equation 1), we sample experience tuples (o_t, a_t, o_{t+1}) from the dataset and randomly select as a goal either another observation from the dataset or o_{t+1}, with equal probability. We use a sparse reward of 1 when the goal is the next observation and 0 otherwise.

In order to prevent overfitting in the low data regime and improve robustness of our learned policy, we employ simple data augmentation techniques to widen the distribution of training examples, similar to Bruce *et al.* [34]. To reduce sensitivity to slight translations and rotations of the fabric in the image, we apply a small amount uniform random translation and rotation to observations sampled from the dataset for training.

Typical instability issues with fixed batch RL arise more prominently in the continuous case [30, 29, 35], further motivating our choice to discretize the fabric folding action space. However, training on offline data can still be a challenge, as the agent is never allowed to collect further evidence for the function it has approximated, and the distribution of training data might differ from the distribution of states the trained policy might experience. The difficulties of such offline training methods is reduced in our case. Despite the true action-space of the task consisting of pixel location, discrete angle and distance, the network does not observe the latter two. It only encodes a single function, the value of folding each pixel a fixed distance to the right. This is further simplified by the weight sharing properties of convolutional networks, where kernels are moved across the image. Thus, the training data is not required to exhaustively cover the state-action space as learned features are spatially invariant across the image.

5 Experiments

In this section we describe the experimental evaluation of our method and discuss the results. Assessed quantitatively and qualitatively, we demonstrate the performance of our model on six different folding tasks. Finally, the ability of our approach to generalize to higher resolution discrete action-spaces and the limitations of our approach are explored.

5.1 Experimental Setup

We use the Franka Emika Panda robot arm with standard gripper, equipped with an eye-in-hand RGB camera, for data collection and testing. We use images of dimension $H = 200, W = 200$. With a camera height of 45cm, this corresponds to a square workspace of 0.33×0.33 meters.

Our fabric is a square, 30cm tea towel. The folding actions are performed on the robot via motion planning. Using force sensing, the robot attempts a top-down grasp at the 3D point corresponding to the selected pixel, and then performs a straight line fold in the specified direction and distance, before lowering and dropping the fabric. During data collection, the robot performs automatic resets with $T = 10$ timesteps, by grasping it in a random location and dropping it from a height of 40cm.

During data collection and training, we use $k = 8$, resulting in 45°bins. With the unit action in our approach being defined as 13cm (corresponding to the original size), the image is scaled by a factor of 2x, 1x and 0.5x before being passed as input into the CNN, resulting in $n = 3$ corresponding action distances of 6.5cm, 13cm, and 26cm (the largest covering almost the full width of the fabric). However, as explored in Section 5.6, we evaluate the ability of our method to act (without further training) with a larger, higher resolution set of discretized actions. In this case, the angles are discretized into $k = 16$, with 22.5°bins. The images are scaled by 2x, 1.5x, 1x, 0.66x and 0.5x, resulting in $n = 5$, with action distances of 6.5cm, 8.6cm, 13cm, 19.5cm, 26cm. As in [21], when acting, we color-mask the fabric and choose pick locations only within the mask.

We utilize a fully-convolutional encoder of 4 layers (32 filters of size 5, stride of 2 for the first three layers followed by stride of 1 for the final layer), and construct the heatmaps by interleaving 2 convolutional layers (32 filters of size 3, stride of 1) with bilinear upsampling. We train our model with standard DQN [36]. Following Zeng *et al.*[28, 3], and because the folding tasks we explore can be achieved in relatively few actions, we use a γ of 0.5, and the Q-learning loss for each action is propagated through the single pixel corresponding the selected action.

We collect 300 samples of training data, which with an average of 12 seconds per action is equivalent to 60 minutes of data collection time on the real robot. We use Huber loss [37] for our loss function, with a learning rate of 0.0001 and a batch size of 10. Early stopping is used to end offline training, when the loss consistently drops below an experimentally derived threshold of 0.05, which occurs after approximately 20,000 gradient steps.

5.2 Folding Tasks

We consider six folding tasks which are unseen during training and executed in the real world. At the start of each trial the fabric is flattened and centered in the workspace. If the goal state is reached, the episode is ended and considered successful. All goals are shown in Figure 4. Of the six goals, (e) and (f) involve multiple fabric layers. These tasks are particularly challenging as the fabric is self-occluding, while the goal is provided as a single image.

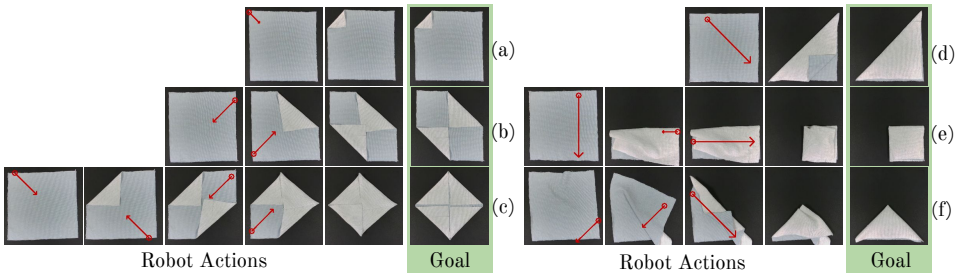

Figure 4: Success examples for each of the six goals. a) Small Inward. b) Double Inward. c) All Corners Inward. d) Triangle. e) Double Straight. f) Double Triangle. Robot actions are visualized as red arrows.

5.3 Baselines

We compare our performance to a random action baseline, an ablated version of our method, and prior work by Ganapathi *et al.* [23]. For the random baseline, we employ the same random action policy utilized during data collection. When the fabric is in the unfolded and open state, i.e. the initial condition in these experiments, this random behaviour occasionally results in folds from the outer edges of the fabric towards or across the centre.

For the ablation experiment, we evaluate the effect of our scaling technique for achieving discrete distance folds. Instead of performing image scaling to produce the fold distance, for each rotated image, the Q-network instead produces three heatmaps, one for each discrete distance. It is trained using the same procedure, setup and dataset as our approach.

Three of our folding tasks (folds (b), (d) and (f)) are the same as Ganapathi *et al.* [23]. Thus, we include their evaluation results for comparision. However, their approach was trained on purely simulated data with an order of magnitude more image samples, and requires human demonstration at test time, which ours does not. Furthermore, they evaluate their algorithm on two robots, a Da Vinci Surgical Robot Research System (dVRK) and a YuMi. These factors make the results not directly comparable, but we include them for reference. Our results are more closely comparable to those on the YuMi because the robot is more similar to our Panda system. The dVRK, however, was designed for a very different and more precise application domain.

5.4 Performance Metrics

Following Ganapathi *et al.* [23] we assess our performance based on fold success rate. The success criteria for a fold is the fabric being "visually consistent with the target image". It is challenging to apply quantitative metrics such as intersection over union (IoU) or structural similarity between the fabric and goal arrangement because of the deformable, self-occluding nature of the fabric and the limitations observing a 3D phenomenon in 2D images. IoU is particularly sensitive to the initial state of the fabric and requires registration of images for accurate comparison, thus we do not use it to determine fold success, however as our initial conditions are similar across trials, it can be used to quantify approximate progress towards goals, as seen in Figure 5. We also include mean IoU of the best state across all trials in Table 1 for comparison.

5.5 Folding Task Results

Table 1 presents the success rate of our method on the six folding tasks. Our approach demonstrates the learned policy by consistently outperforming the random policy and ablated baseline and succeeding at all folds which do not involve self-occlusion of the fabric in all runs. On the more challenging *double straight* and *double triangle* folds, we achieve 6 and 1 successes out of 10 trials respectively. We visualize successful trajectories performed by our approach, for all goals, in Figure 4.

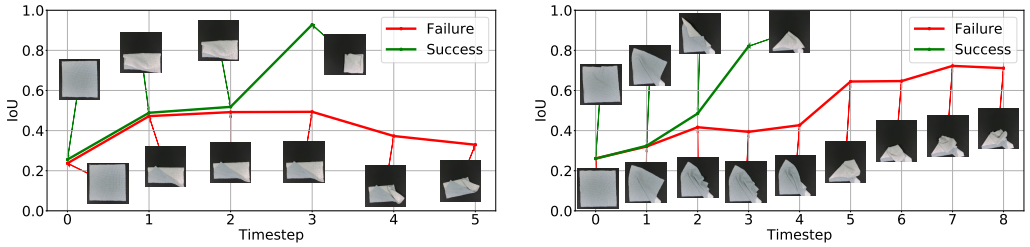

Figure 5: A comparison of IoU over time between successful and failed for the Double Straight (Left) and Double Triangle (Right) folds.

	(YuMi)	(dVRK)	Random		Ours (No scale)		Ours	
Goal	Success	Success	Success	*IoU*	Success	*IoU*	Success	*IoU*
a) Small Inward	-	-	1	0.9	0	0.95	10	0.95
b) Double Inward	8	9	0	0.7	2	0.73	10	0.88
c) Four Corners Inward	-	-	0	0.53	0	0.65	10	0.87
d) Single (Triangle)	8	9	2	0.58	0	0.7	10	0.89
e) Double Straight	-	-	0	0.34	0	0.54	6	0.76
f) Double Triangle	6	8	0	0.38	0	0.53	1	0.69

Table 1: The success rate for the six folding tasks of our folding experiments. Each value is the number of successes out of 10 conducted trials. We observe consistently greater performance over the random baseline and comparable performance to previous work on similar folding tasks. Mean IoU across trials is included for the experiments we conducted.

In our approach we account for various fold angles and action distances by discretizing input rotations and scales respectively. In achieving consistent performance across multiple different folding tasks which involve folds of various angles or sizes this approach is validated. For example *small inward* and *single (triangle)* fold require the same action of different distances and the *four corners inward* fold requires multiple folds of different angles, and our method successfully selects the correct angles and distances to achieve the goals. Without the scaling technique that our method employs, the folding task becomes an inherently harder learning problem, as shown by the ablation experiment in Table 1.

The ablation baseline approach fails to reach all goals except the *double inward* fold, which it achieves twice. This approach was unable to consistently apply the correct fold distance in the appropriate situations. However, despite failing all other goals consistently, we note that the approach has learned some folding ability, as shown by the failed attempts in Figure 6, as well as the IoU values in Table 1. We note also that this approach does not have the same action-space generalization ability that ours maintains; the model must be trained on a specific discretized action-space, while our approach can generalize to higher resolution action-spaces at test time. As expected, the random baseline rarely achieves the goal, and often fails irreversibly after the first action. The IoU scores reflect this, as the visually best state in a given run is often the initial configuration, which for a goal like *small inward* is already close to the goal (and thus the IoU is high).

For the more complex goals in our test set (*double straight, double triange*), despite being unable to directly observed the required folds, the system must infer the next best action in the sequence that will achieve the goal image configuration. Occasionally, the policy will make small adjustment actions. In the case of the *double straight* task in Figure 5, the network is sensitive to the overlapping top layer at the bottom right corner of the image. In the successful cases, it is able to reduce the visual appearance of this overlap via small adjustments and proceed to the goal. However, in the failed case, the policy seems unable to correctly adjust, and therefore does not reach the goal.

In the *double triangle* task shown in Figure 5, the policy chooses to fold the top right corner over in two steps. However, this strategy often fails due to the complexity of folding two layers at once. We also note that, in a real robot setting the grasp quality and reliability must be considered. The challenges of grasping multiple layers of fabric in the *double straight* and *double triangle* folds impacted performance, which is a physical limitation of the gripper.

5.6 Generalization

The complexity of learning to fold is significantly reduced by discretizing the input into eight possible rotations and 3 possible scales. This set of discretizations enables our method to be sample

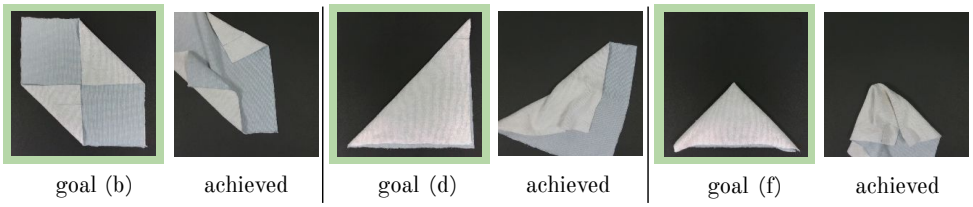

Figure 6: Examples of failed attempts performed by the ablated baseline approach. While the learned policy does not successfully complete the folding tasks according to our success metric, visually we see that it performs informed actions leading to configurations that approximately resemble the goal.

efficient and is expressive enough to solve a wide range of goals, as shown in Figure 4. However, for more complex goals, or when more precision is required, our method can be scaled up to higher resolution discretizations without retraining the model. Our method is able to effectively utilize a larger action space at test time. Figure 7 presents the outcomes for two folding tasks which were hand-designed to require an angle and scale not present in the discrete options. The ability to specify an arbitrary discretization resolution at test time without retaining the model demonstrates its generalization ability. In both cases, the model is able to more accurately perform folds it is unable to achieve with the lower resolution action-space used for data collection and training.

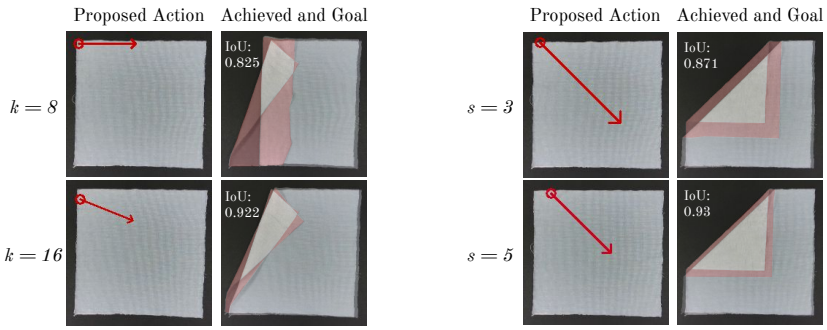

Figure 7: The capability of our model to perform under a higher resolution action-space discretization than experienced during training. Goal is shown overlayed over the state achieved by our approach, with error highlighted in red. Left: we increase the number of angle bins from 8 to 16. Right: We increase number of fold distance bins in the distance discretization from 3 to 5.

6 Conclusion

In this work, we present a framework for learning fabric folding directly on a robot in the real world, without requiring human demonstrations or simulated data. Our offline reinforcement learning approach is able to solve unseen, complex, sequential fabric folding tasks, with only one hour of real data collected in advance. We achieve this by utilizing a goal-conditioned, fully convolutional network, trained offline with Hindsight Experience Replay. By discretizing rotation and scale, and exploiting the visual proportionality of folding distance, our approach is extremely sample efficient, yet expressive enough to solve complex fabric folding tasks. Further, we show the accuracy of the behaviors can be increased after training by simply increasing the discretization resolution of the action-space, allowing our approach to solve a wider range of goals.

In future work, we aim to extend our approach and address several limitations. We would like to improve in the area of long-horizon sequential and complex, partially occluded goals. It would be desirable if the system could detect a successful final state, or whether the configuration is unable to be recovered from. Metrics for success in fabric folding are limited, and it would be valuable to define a more effective quantitative metric. Furthermore, while we trained our model with a single cloth, our approach is not specific to a particular material. Thus, we would like to explore the capability of our method to generalize to cloths of different color and material by exploring data augmentation techniques, as well as interacting with a variety of fabrics during the data collection stage. Applying this approach to other tasks such as smoothing is another interesting direction, which may be achievable by widening the distribution of actions during data collection, or by algorithmic techniques such as time reversal [38], which could be used to apply folding experience directly to unfolding. Exploring our approach for other deformable objects such as rope or cables is another promising direction.

References

[1] D. Morrison, A. W. Tow, M. Mctaggart, R. Smith, N. Kelly-Boxall, S. Wade-Mccue, J. Erskine, R. Grinover, A. Gurman, T. Hunn, et al. Cartman: The low-cost cartesian manipulator that won the amazon robotics challenge. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7757–7764. IEEE, 2018.

[2] D. Morrison, P. Corke, and J. Leitner. Learning robust, real-time, reactive robotic grasping. *The International Journal of Robotics Research*, 39(2-3):183–201, 2020.

[3] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 2020.

[4] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[5] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, 37(7):688–716, 2018.

[6] A. Cherubini, V. Ortenzi, A. Cosgun, R. Lee, and P. Corke. Model-free vision-based shaping of deformable plastic materials. *The International Journal of Robotics Research*, page 0278364920907684, 2020.

[7] M. Staffa, S. Rossi, M. Giordano, M. De Gregorio, and B. Siciliano. Segmentation performance in tracking deformable objects via wnns. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 2462–2467. IEEE, 2015.

[8] V. E. Arriola-Rios and J. L. Wyatt. A multimodal model of object deformation under robotic pushing. *IEEE Transactions on Cognitive and Developmental Systems*, 9(2):153–169, 2017.

[9] Y. Li, D. Xu, Y. Yue, Y. Wang, S.-F. Chang, E. Grinspun, and P. K. Allen. Regrasping and unfolding of garments using predictive thin shell modeling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[10] Y. Li, X. Hu, D. Xu, Y. Yue, E. Grinspun, and P. K. Allen. Multi-sensor surface analysis for robotic ironing. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[11] 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 *Proceedings of the 27th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[12] F. Osawa, H. Seki, and Y. Kamiya. Clothes folding task by tool-using robot. *Journal of Robotics and Mechatronics*, 18(5):618–625, 2006.

[13] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O'Brien, and P. Abbeel. Bringing clothing into desired configurations with limited perception. In *2011 IEEE International Conference on Robotics and Automation*, pages 3893–3900. IEEE, 2011.

[14] B. Willimon, S. Birchfield, and I. Walker. Model for unfolding laundry using interactive perception. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4871–4876. IEEE, 2011.

[15] L. Sun, G. Aragon-Camarasa, P. Cockshott, S. Rogers, and J. P. Siebert. A heuristic-based approach for flattening wrinkled clothes. In *Conference Towards Autonomous Robotic Systems*, pages 148–160. Springer, 2013.

[16] L. Sun, G. Aragon-Camarasa, S. Rogers, and J. P. Siebert. Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 185–192. IEEE, 2015.

[17] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen. Folding deformable objects using predictive simulation and trajectory optimization. In *Proceedings of the 27th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[18] D. Seita, N. Jamali, M. Laskey, R. Berenstein, A. K. Tanwani, P. Baskaran, S. Iba, J. Canny, and K. Goldberg. Deep Transfer Learning of Pick Points on Fabric for Robot Bed-Making. In *International Symposium on Robotics Research (ISRR)*, 2019.

[19] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2146–2153. IEEE, 2017.

[20] R. Jangir, G. Alenyà, and C. Torras. Dynamic cloth manipulation with deep reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4630–4636. IEEE, 2020.

[21] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel. Learning to manipulate deformable objects without demonstrations. *arXiv preprint arXiv:1910.13439*, 2019.

[22] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, K. Yamane, S. Iba, J. Canny, and K. Goldberg. Deep Imitation Learning of Sequential Fabric Smoothing From an Algorithmic Supervisor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[23] A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, et al. Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images. *arXiv preprint arXiv:2003.12698*, 2020.

[24] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg. Visuospatial foresight for multi-step, multi-task fabric manipulation. *arXiv preprint arXiv:2003.09044*, 2020.

[25] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto. Learning predictive representations for deformable objects using contrastive estimation. *arXiv preprint arXiv:2003.05436*, 2020.

[26] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 734–743, 2018.

[27] M. Laskey, C. Powers, R. Joshi, A. Poursohi, and K. Goldberg. Learning robust bed making using deep imitation learning with dart. *arXiv preprint arXiv:1711.02525*, 2017.

[28] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4238–4245. IEEE, 2018.

[29] S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

[30] R. Agarwal, D. Schuurmans, and M. Norouzi. Striving for simplicity in off-policy deep reinforcement learning, 2020. URL https://openreview.net/forum?id=ryeUg0VFwr.

[31] S. Cabi, S. Gómez Colmenarejo, A. Novikov, K. Konyushkova, S. Reed, R. Jeong, K. Zolna, Y. Aytar, D. Budden, M. Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv*, pages arXiv–1909, 2019.

[32] V. Dasagi, R. Lee, J. Bruce, and J. Leitner. Evaluating task-agnostic exploration for fixed-batch learning of arbitrary future tasks. *arXiv preprint arXiv:1911.08666*, 2019.

[33] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba. Hindsight experience replay. In *Advances in neural information processing systems*, pages 5048–5058, 2017.

[34] J. Bruce, N. Sunderhauf, P. Mirowski, R. Hadsell, and M. Milford. Learning deployable navigation policies at kilometer scale from a single traversal. In *Conference on Robot Learning*, pages 346–361, 2018.

[35] N. Y. Siegel, J. T. Springenberg, F. Berkenkamp, A. Abdolmaleki, M. Neunert, T. Lampe, R. Hafner, and M. Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

[36] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[37] P. J. Huber. Robust estimation of a location parameter. *Ann. Math. Statist.*, 35(1):73–101, 03 1964. doi:10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

[38] S. Nair, M. Babaeizadeh, C. Finn, S. Levine, and V. Kumar. Trass: Time reversal as self-supervision. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 115–121, 2020. doi:10.1109/ICRA40945.2020.9196862.