7 Appendix

7.1 System Design Choices

One of the main intentions of this paper is to show the capabilities of local visuotactile perception and control for deformable object manipulation, rather than overall task success or efficiency. While both edge grasping and tactile sliding have high success rates if initialized properly, the transitions require more engineering to make smooth. We choose to use a state machine such that whenever an action fails, our visual and tactile checks (if accurate) can restart the system where appropriate.

State Machine. In the state machine (See Fig. 2), the robot tries to pick up the towel from the ground with its right gripper, then grasp a corner (the lowest point of the towel) with its left gripper. After each of these steps, visual perception is used to ensure that the grasp is successful, meaning that there is cloth in the expected region of the depth image. If at any point the towel is on the ground, then the state machine restarts from the initial pick. Once a corner is grasped, the robot uses our grasp affordance network to find an edge grasp whose affordance value is above a certain threshold. If no grasp is found, the towel is rotated. If a full rotation is reached, then the towel is released to the floor and the state machine restarts. The gripper approaches the edge with a fixed orientation. After grasping an edge (confirmed using tactile), the gripper slides to the adjacent corner, where the end condition is determined from tactile. From this position, the robot can unfold or fold the towel using open-loop commands because we use a towel of a fixed size.

Motivation for Hanging. While ideally we would grasp edges or corners directly from the table-top if visible, these grasps are usually unavailable for our hardware. Other papers use a compliant surface or smaller, tweezer-like grippers that can more easily slide under a layer of cloth in order to grasp an edge [20]. Our tactile grippers, while unable to directly grasp a single layered edge from the ground due to their larger size, enable other skills (sliding, tensioning). Picking the towel off the table is an important step so that edge grasps are exposed.

Corner Detection. We originally intended to use corner detection in both tactile networks as an end condition for tactile sliding and to confirm direct corner grasps. However, we found that sliding over the thicker corner required a wider gripper opening that compromised tactile perception of the edge during the rest of sliding. Therefore, we chose to use shear in the tactile signal of the stationary gripper as the indicator of reaching the opposite corner. Corner detection accuracy in the grasp network was not sufficiently high for direct corner grasping for our task (Fig. 7). The requirements for that initial pick of a corner are less stringent, since it is not necessary to grasp on a "clean" corner. For example the robot can pick up a corner folded in on itself and still successfully complete the task. Therefore, we found picking the lowest point and using vision to confirm that the towel is being held was sufficient for our task.

Individual Action Evaluation. We evaluate the success of individual actions because our system can sense and handle failures of actions, but some transitions would require further engineering to smooth out. We evaluate our edge grasp policy starting from a corner grasp and our tactile sliding controller starting from an edge grasp. We vary initial configurations across trials (cloth configuration for edge grasping and edge position for tactile sliding).

7.2 Failure Modes

We summarize several major failure modes during different phases of the system.

Corner Grasping. The main failure modes of corner grasping are slightly missing the towel (be-

cause of movement while sensing or errors from the camera) or the lowest point not being a corner (if the towel is partially folded). This can be improved with modified tactile verification.

Edge Grasping. Edge grasping fails if the tactile classifier falsely determines an edge is grasped (especially if grasping a compressed fold). Another common failure mode is that the towel is rotated fully without a viable edge grasp. Sometimes, the edge is visible but sticks to another layer of cloth. This can be improved with more dexterous grippers or manipulation skills to separate an edge from other parts of the cloth with vision and tactile feedback.

Grasping and Sliding. The edge grasping to sliding transition is difficult because some successful edge grasps can be challenging for sliding. The position of the edge is not a consideration for the edge grasp network, while it is a factor that highly influences sliding success. For example, when the edge is too close to the tip of the gripper, it can be difficult to adjust back. Additionally, the tactile sliding network performs better on the thin edges of the towel while the edge grasp affordance network tends to give thicker edges higher affordances. These can be improved with a tactile edge adjustment module, which adjusts the edge pose with tactile regrasps [34] before sliding, and dynamic manipulation [35, 16] to unfurl the cloth during sliding.

7.3 Network Architectures and Training Details

Tactile Perception Networks. We choose a small network architecture for our tactile sliding network to optimize for speed. Specifically, our base structure is C6-C16-C32-MP-F512-F128, where C represents convolutional layers (with 5x5 kernel sizes), MP represents max-pooling layers (with kernel size of 2x2), and F represents fully-connected layers. We use ReLU as our activation function. We add separate final fully-connected layers for pose estimation, and classification. Pose estimation uses a loss function of Mean Square Error (MSE) and classification uses cross-entropy loss. We use the Adam [36] optimizer. For handling video sequences, we use a similar structure, with stacked frames in the color channel, following [37]. In our experiments, we stack 5 frames in the end of the video, with interval of 5 frames (e.g. frames 10, 15, 20, 25, 30). We train our model with NVIDIA Titan X Pascal GPU, and the training time for our tactile perception networks is about 5 hours.

Visuotactile Networks. We choose U-Net[1] as our network for training visuotactile affordances. We use MSE loss for the whole image to train affordance in simulation and use MSE loss for individual pixels for fine-tuning with real grasps. For fine-tuning, we also include the loss of the pixels around the selected pixel (with distance in 3 pixels) to improve training efficiency. For better generalization and preventing over-fitting, we reduce the number of channels by half in each layer, and remove the skip connection in the last two Up-sampling layers. Based on our experiments, this modification produces more generalized and smoother affordance. During experiments, we also test fully convolutional networks [29]. In comparison, we find that U-Net provides more resolution, which is crucial for successful edge grasps. We train our models with NVIDIA Titan X Pascal GPU. The training time for our visuotactile affordance networks in simulation is about 8 hours, and the training time for fine-tuning is about 1 hour.

7.4 Data Augmentation for Tactile Perception

For both the tactile sliding and grasp supervision networks described in Sec. 4.1, we augment the data we collect and label using the process shown in Fig. 6. We randomly vary the maximum depth threshold and further augment the images by introducing a random translation and rotation. For the tactile sliding pose network, the hand-labeled edge is transformed as well. Different categories have their own transformation parameters. For example, no fabric images have minimal transformations because the imprint in the depth image is caused by the convexity of the gel surface, which is always in the same position.

[1] See https://github.com/milesial/Pytorch-UNet for details.

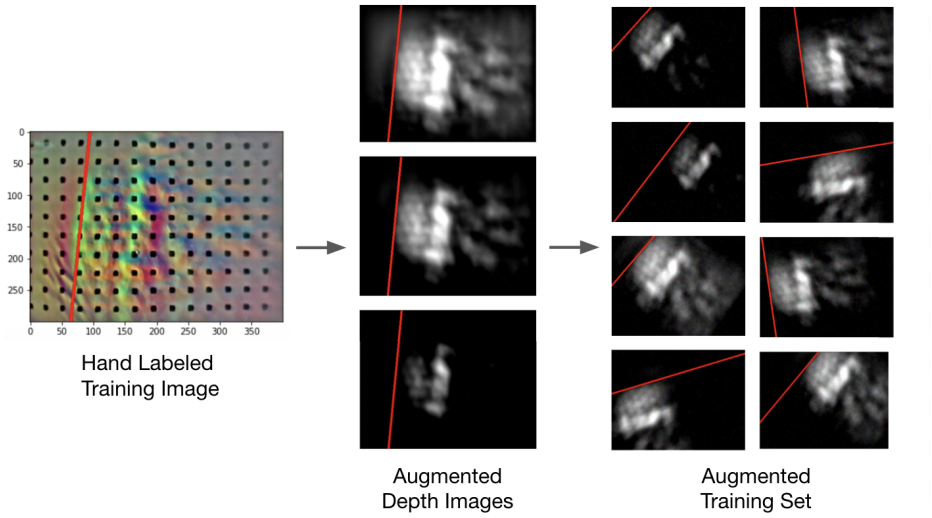

Figure 6: **Dataset augmentation for tactile perception.** We augment the data by varying the depth scale and transformation to improve results with a limited amount of labeled data.

7.5 Tactile Grasp Supervision Network Results

Tactile classification accuracy is evaluated on a held out test set evenly distributed across classes. We use a network that best differentiates between edge and non-edge grasps. The network performs well on most categories except for folds (See Fig. 7), which can look like most other categories. However, the network is sufficiently capable of distinguishing edges and folds for the purposes of our grasp supervision network.

7.6 Training Affordance Network in Simulation

The ground-truth labels of the affordances in simulation are determined for each pixel by geometric computing. Specifically, for each scene, we extract the depth image, the point cloud of the visible and the whole cloth, and the undeformed coordinates of each point in the local cloth frame. The parallel-jaw gripper is represented as a 3D box. Given a grasping point, we calculate the ground-truth affordance using the following criteria:

- **Edge percentage.** The percentage of neighboring cloth nodes that are classified as an edge, in the gripper box. Computationally, the edge is determined based on the distance to the cloth boundary, in the undeformed local cloth frame.
- **No collision.** Whether the gripper will collide with the cloth before grasping. This checks whether the orientation of the local cloth is aligned with the gripper, to grasp without the cloth being pushed away. Computationally, we assign some thickness to each finger of the gripper in the box region to simulate the real hardware. We check whether surrounding points collide with these finger regions.
- **Single layer.** Whether the gripper would grasp a single layer of fabric. A single layer of cloth is desired to start sliding. Computationally, we cast rays from one finger to the other finger and check whether each ray collides with a single layer of points. For points colliding with each ray, we set a threshold based on the grid size to determine whether these points are adjacent or apart in the undeformed coordinates in the local cloth frame.
- **Reachability.** Whether the point is reachable from the robot arm on the right side without colliding with other parts of the cloth. Computationally, we check the sweeping region from the right side of the candidate gripper box, to determine whether there are cloth points in collision.

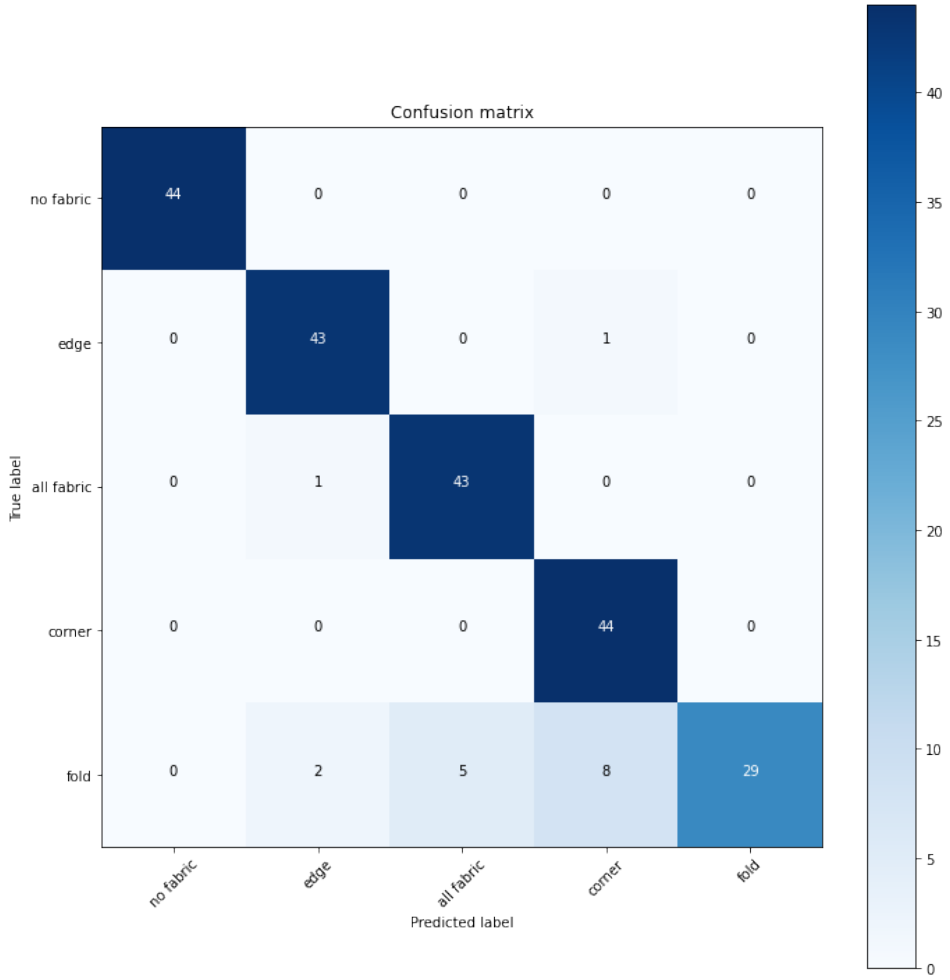

Figure 7: **Confusion matrix for tactile grasp supervision.**

An example is shown in Fig. 8 for visualization. All of these criteria can be determined in simulation because we have access to the full state of the cloth. The tactile sensor implicitly checks all of these criteria on the real system when it classifies a grasp as an edge or non-edge grasp. The simulated dataset contains 200 cloth configurations rotated in increments of 15°, yielding 4800 depth images with corresponding affordance images (Fig. 9).

To test the feasibility of fine-tuning on the real robot system where we can only test one pixel at a time during training, we try fine-tuning our network in a slightly different simulation environment. The second simulated dataset has slightly different camera location, cloth size, and cloth stiffness. We test several different network parameters including the addition of a replay buffer, which layers are frozen, balanced sampling from the replay buffer, adding random crops for data augmentation, and updating multiple nearby pixels at once during training. Our key finding was that fine-tuning with single pixel updates is feasible with an experience replay buffer. Without the replay buffer, training converges with 88,000 grasps. With the replay buffer, we achieve comparable performance with 3,000 grasps.

7.7 Sim-to-Real Transfer

The real depth images on the robot have regions of zero-depth due to the offset in location between the projector and sensor of the depth camera. Also, items in the background affect the affordance.

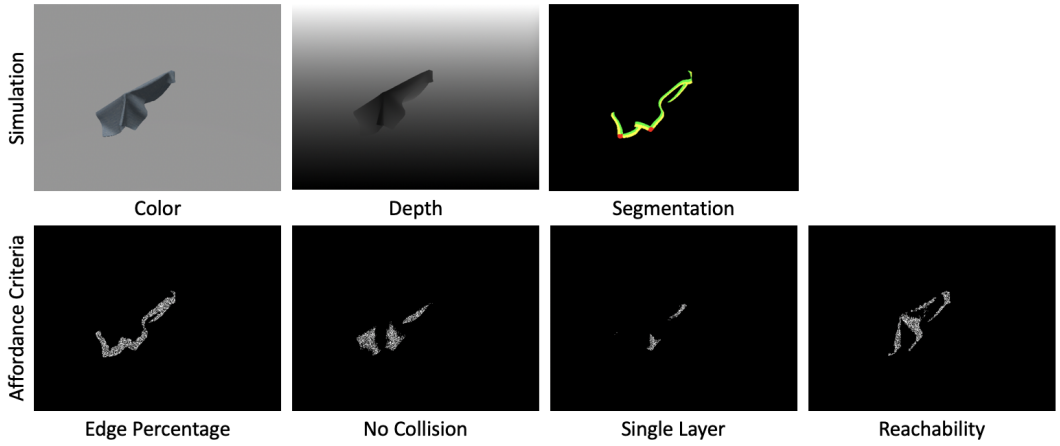

Figure 8: **Affordance calculation in simulation.** *Top*: the color image, depth image, and segmentation of a piece of cloth from the simulated environment. The segmentation is visualized with corners in red, outer edges in yellow, inner edges in green, following the visualization in [20]. *Bottom*: the criteria used to calculate the affordance. The final affordance combines these together.

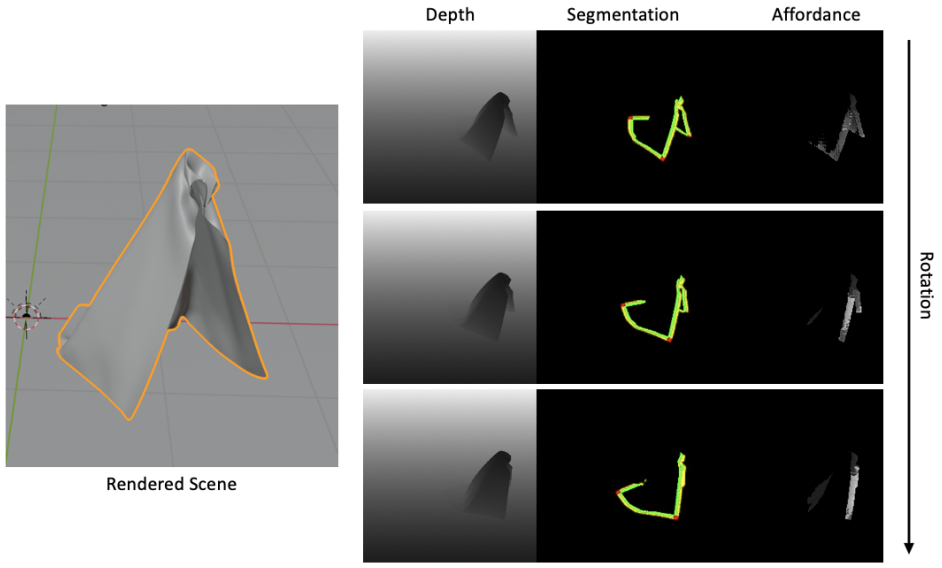

Figure 9: **Simulated dataset for affordance network.** With rotation, the affordance becomes higher when the edge is aligned with gripper, and can be grasped with no collision.

We found that the original network tends to assign high affordances to the zero-depth regions (Fig. 10). Therefore, we mask out all non-cloth regions in both simulation and deployment and add random black rectangles with different sizes and orientations to the simulated depth images during training for robustness against regions with zero-depth (Fig. 11).

7.8 Robot Implementation and Control

Grasping with the UR5s uses position-control while tactile sliding uses velocity-control for smooth behavior, which runs at around 30 Hz. For the grippers, we use Dynamixel motors. They are position-controlled with current limits to protect the sensor. For tactile classification with a multi-frame input, we sent decreasing width commands over time to gradually close the gripper, and collect the tactile signals under different gripper widths.

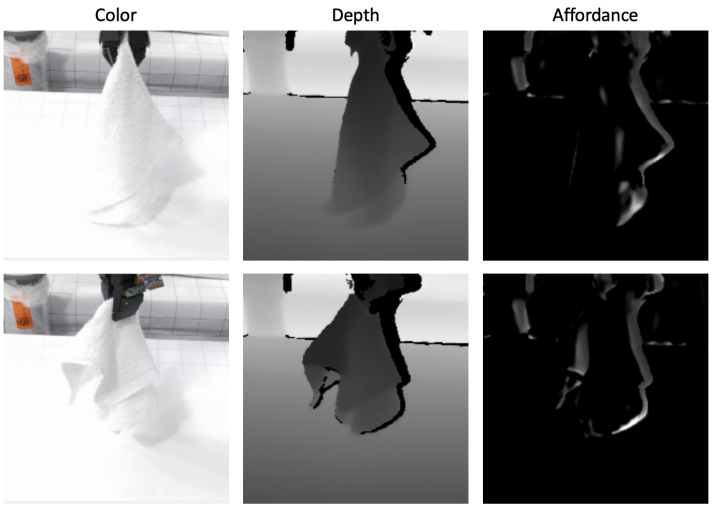

Figure 10: **Direct sim2real transfer without additional modifications.** Without additional modification, the Sim2Real network attends to the black region caused by the offset between the projector and the sensor of the depth camera.

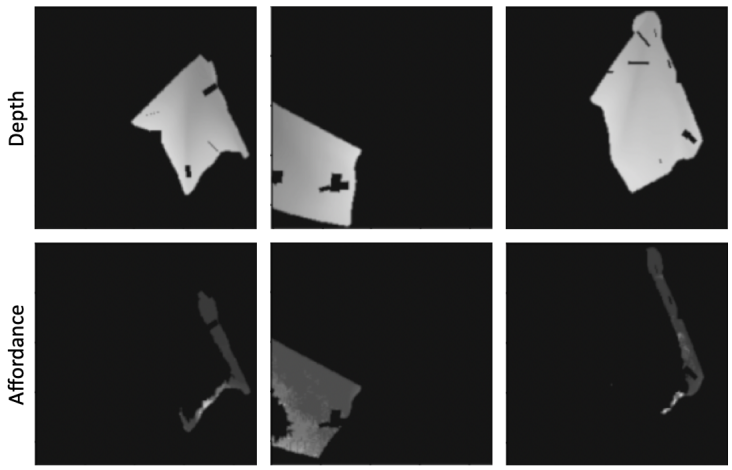

Figure 11: **Additional modification with mask and black blocks for Sim2real transfer.** With the added cloth mask and random black blocks, the network can predicate the affordance as desired.

7.9 Segmentation-Based Affordance Baseline

We train a network based on [20] that takes a visual depth image as input and outputs the segmentations of corners, outer edges, and inner edges. We collected 10,000 data points on our system, with a painted cloth to label different segmentation to provide ground-truth labels [20]. To assign affordances to each point, we have a cost metric that includes edge direction uncertainty, right side reachability, and collision-free surface normals.

7.10 Vertical Tactile Sliding

To test the robustness of vertical tactile sliding, we varied the initial edge position across experiments with 5 trials per initial condition (Table 2). The controller was able to consistently follow the entire length of the thin edge of the towel, but sliding success along the thicker edge was dependent on starting pose. The closer the edge was to the tip of the gripper (less initial cloth coverage), the more likely the cloth was to slip out before the gripper slid to the edge.

Initial Cloth Coverage of Tactile Sensor (%)	Distance Traveled (%)	
	Thin Edge	Thick Edge
100%	100%	100%
75%	100%	92.1%
50%	100%	90.4%
25%	100%	82.9%

Table 2: Vertical tactile sliding results given different starting configurations (n = 5 trials/configuration). Initial edge position indicates the fraction along tactile sensor that is covered in fabric. With an initial cloth coverage of 100%, the cloth edge is aligned with the sensor. The cloth edge is centered within the sensor at 50%.

7.11 Horizontal Tactile Sliding

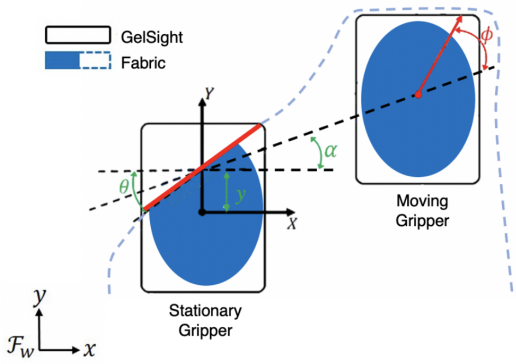

Figure 12: **Cloth sliding dynamics.** Diagram of the cloth sliding modeling.

We use horizontal sliding to slide along longer towels. Horizontal sliding has more complex dynamics than vertical sliding because the towel is more likely to slip out of the grip due to gravity, and the controller needs to compensate. We model the dynamics as shown in Fig 12. Similar to the cable following system from [28], the system's state is $\mathbf{x} = [y\ \theta\ \alpha]^T$, where y is the fabric edge height from the target edge position on the gripper, θ is the fabric edge angle with respect to the gripper, and α is the angle between the two grippers (Fig. 12). The control input $\mathbf{u} = [\phi]$ is the pulling angle. The linear dynamic model is of the form:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{1}$$

where A and B are the linear coefficients of the model.

To learn a dynamics model, we collect data using a simple proportional controller ($\phi = -Ky$) supplemented with uniform noise. The dataset has 7100 observations from 30 different following runs with a variety of starting states and pulling angles. We perform both linear regressions and sparse Gaussian Process (GP) regressions. From initial testing, we were not sure if a linear model could capture the complex dynamics of cloth sliding, so we tested controllers based on the non-linear GP dynamics model as well.

We compare the two best performing controllers for cable following for cloth edge following. The first baseline controller is Linear–quadratic Regulator (LQR) using the linear dynamics model. The second time-varying LQR (tvLQR) controller linearizes the GP model about the current state. For the LQR parameters, we use $\mathbf{Q} = [100000, 1, 0.1]$ and $\mathbf{R} = [0.1]$, to prioritize regulating y and θ (so the cloth does not fall).

When we tested these controllers on the physical system, we found that both controllers performed comparably. However, The tvLQR network with a linear kernel for the $\dot{y}$ model had similar commanded actions to LQR based on the linear model, but it runs slower due to the time required to calculate the predictive gradient for the GP. Therefore, we found the linear model was favorable.

Horizontal sliding experiments are initialized by hand-feeding the towel to the grippers. With an optimal starting position, the LQR controller based on the linear model consistently travels to the end of the workspace in all five trials. We adjust the y setpoint to be much closer to the inner edge than with cable following. When the initial position is in the center of the gripper (closer to the tip compared to the setpoint), the gripper traverses an average of 76% of the workspace (43 cm out of 56 cm).

7.12 Potential for Generalization

In this work, we generate local techniques for perception and control more tenable for generalization across types of cloth because they do not rely on global structure like most previous works. However, in order to apply the same framework to a different towel, individual modules might need to be updated to handle different size, dynamics, texture, etc. (Sec. 6). Greater variety in the training set could improve generality. This approach focusing on local features can also be useful for more complex clothing items like shirts or pants. Tactile supervision in our framework only works if different categories (edge, corner, fold, ...) can be reliably differentiated. If a feature is unique, like the hem of a shoulder strap, a zipper, or a shirt collar, then we could similarly use tactile feedback to confirm grasps of local features. However, it would be difficult to differentiate, for example, a sleeve hem and a waist hem on a T-shirt, which might be necessary for certain manipulation tasks. A reasonable way to approach this more general problem would be to use a visual prior to inform of the global structure of the cloth [21] and use tactile sensing to confirm a grasp on a hem.