Vision-Guided Robotic Folding of Deformable Garments Using Model-Based Planning and Control

Youhui (Jeffrey) Wang
Harvard College
Harvard University
Cambridge, MA, USA
jeffrey_wang@college.harvard.edu

Hangxing Zhang
Harvard College
Harvard University
Cambridge, MA, USA
hangxingzhang@college.harvard.edu

***Abstract*—Robotic manipulation of deformable clothing has long faced challenges due to its high-dimensional configuration space, unpredictable contact behavior, and sensitivity to visual errors. This paper proposes a square fabric folding framework that combines geometric vision perception, deformable object modeling, motion planning, and compliant control. We construct a fabric model based on a tetrahedral mesh and use depth maps to extract the four corners and overall orientation of the fabric, thereby calculating the gripping points and folding lines. Subsequently, these visual results are input into a constraint-based trajectory planner to generate collision-free execution trajectories for the Franka Panda robotic arm and gripper, and impedance control is used to stably adjust the contact force on the fabric. We validate the system performance under 10 sets of random initial environments with the same size of square fabric and 10 sets of constant initial environments with 10 different sizes of square fabrics, achieving success rates of 70% and 90%, respectively. The results demonstrate that the combination of geometric vision and compliant control provides an effective and interpretable solution for deformable object manipulation.**

***Index Terms*—Robotic manipulation, deformable, perception, control,**

I. Introduction

Manipulating deformable objects is one of the most challenging areas in robotics. Compared to rigid bodies, fabric deforms continuously during grasping and movement due to gravity, contact, and tension, making its state difficult to describe with low-dimensional parameters and accompanied by highly nonlinear contact behavior and sensory noise. Therefore, manipulating flexible materials is far more complex than manipulating rigid bodies in terms of perception, state estimation, and planning control. Although significant progress has been made in recent years in fabric modeling, learning control, and differentiable simulation, constructing a reliable and generalizable clothing folding system remains an open problem.

This paper proposes a square fabric folding system that combines geometric vision, deformable body modeling, task space planning, and compliant control. We use a Franka Panda arm and gripper that is similar to KUKA IIWA 14 robotic arm and a Schunk WSG 50 gripper to construct a tetrahedral flexible model of the fabric in a simulation environment. The vision module extracts the four corner points of the fabric from the image and estimates their orientation; the planning module generates a single-sided folding trajectory based on this geometric information; and the control module employs impedance control to handle local deformation and contact disturbances during grasping and folding. The experiment was conducted under 10 different initial environments and 10 different square fabric sizes under a constant environment, with overall folding success rates of 70% and 90%, respectively, indicating that the folding frame combining geometric vision and physical modeling has good stability and a certain degree of generalization.

II. Literature Review

Research on robotic cloth manipulation mainly focuses on three areas: folding strategies, planning and control of deformable objects, and visual perception methods. In folding tasks, classic methods are often based on geometric heuristics, generating fixed folding sequences by extracting the outline and key points of the clothing, such as Van den Berg et al.'s gravity folding model [7] and Miller et al.'s geometric template folding pipeline [8]. Dual-arm systems further improve cloth alignment and unfolding capabilities [9], [10], but rely heavily on the regular shape of the clothing and its relatively flat initial state. Learning methods such as reinforcement learning and imitation learning have also been used for cloth manipulation [12], [14], [15], but related reviews point out their common problems of insufficient generalization and interpretability [16], [17].

In terms of planning and control, traditional sampling planning (such as RRT [4]) struggles to handle the high-dimensional deformation space of the cloth; while optimization-based methods (such as SNOPT [2]) have strong expressive power, they are costly under contact coupling. Differentiable simulation methods (such as DiffCloth [18]) promote the integration of fabric control and gradient learning by differentiable modeling of friction and contact. Meanwhile, Drake provides a unified framework for rigid-flexible body modeling and contact solution [5], [6], providing a physical foundation for systematically constructing fabric folding tasks.

In terms of visual perception, early work relied on RGB-D contour and geometric keypoint extraction suitable for regular fabrics [8], [10]; more complex scenes emphasize dense correspondences or region semantics, such as Seita

et al.'s fabric motion prediction, and Sunil et al.'s dynamic clothing visual-tactile joint perception [15], [19]. Although deep learning methods perform well in complex situations, geometric vision still has higher interpretability and stability in simple, structured scenes.

In summary, existing methods have made significant progress in cloth folding, soft body control, and visual perception. However, few works have unified physical modeling, geometric vision, and task space planning into an interpretable and reproducible pipeline. This paper employs geometric corner detection and folding planning based on a physical model to provide a concise and systematic validation framework in this direction.

III. Methodology

A. Environment Setup

This work builds the overall system based on the open-source simulation environment specifically for dynamic-cloth-folding [21]. This environment uses MuJoCo as the physics engine and exposes observations and actions through the OpenAI Gym interface, making it a high-fidelity simulation platform specifically designed for cloth folding. In this environment, we can import the following items into the scene (Figure 1.):

- A 7-DOF Franka Emika Panda robotic arm and gripper
- A square piece of cloth placed on a flat table
- A virtual camera simulating an Intel Realsense D435 for acquiring RGB images

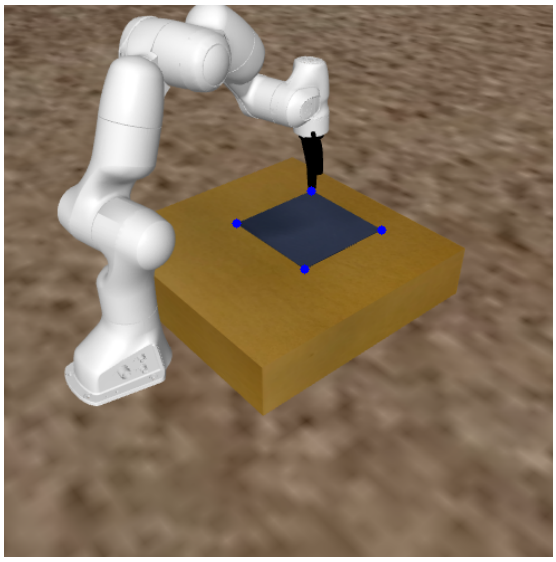

Fig. 1. Imported items in the environment [21] (A 7-DOF Franka Emika Panda robotic arm and gripper, a square fabric, a flat table, and a virtual camera for acquiring RGM images)

From a kinematic perspective, the Panda arm is highly similar to the KUKA IIWA 14 arm commonly used in the course: both are 7-DOF serial structures, both have redundant "shoulder-elbow-wrist" joint topologies, and both have the same task space dimension for the end effector. Therefore, the forward/inverse kinematics, Jacobian matrices, and task space control methods used in this environment can be directly analogous to the control of the IIWA 14, except that the underlying model is replaced by the Panda's MuJoCo XML instead of the IIWA URDF/SDF [21]. This environment provides a series of key configuration parameters for cloth folding tasks, particularly suitable for the controllable and highly repeatable experimental design style highlighted in the course:

- '`--cloth-size`': The side length of the cloth (in meters), which can be freely set
- '`--kp`, `--damping-ratio`': The stiffness gain and damping ratio of the task space OSC (Operational Space Controller)
- '`--timestep`, `--control-frequency`': The simulation step size and control frequency, used to determine how many physical steps are included in one control cycle
- '`--camera-type`' and '`--camera-config`': Camera viewpoint (top view, side view, etc.) and field of view parameters;
- A complete set of '*domain randomization*' options (e.g., randomization of lighting, materials, camera position and orientation, cloth dynamics parameters, etc.)

These functionalities are very similar to how we configure the ManipulationStation in Drake: In Drake, we build the system using '`DiagramBuilder`', load the robot and environment geometry using '`AddMultibodyPlantSceneGraph`', configure the contact model and controller, and then push the time using '`Simulator`'; in this environment, the above process is encapsulated within the Gym environment—the MuJoCo XML file acts as a "MultibodyPlant + scene graph", the OSC controller corresponds to the PD/impedance controller implemented in the task space in the course, and the high-level Python code interacts with it through the following loop [21]:

1) Call '`env.reset()`' to initialize the physical state of the Panda-cloth-camera
2) In each control cycle, calculate the desired end-effector pose or its increment based on the current observation (robot state and image)
3) Pass this desired pose (or displacement) as a Gym action into the environment, where the OSC converts it into task space forces/accelerations, which are then processed by MuJoCo, which then completes one physical simulation update;
4) Repeatedly call '`env.step(action)`' to achieve a "control-simulation" loop; this is equivalent to '`simulator.AdvanceTo(t)`' in Drake

The action space in the environment is essentially the increment of the desired end-effector pose (e.g., within each control cycle, the maximum displacement of the end-effector in the x, y, z directions is limited by '`--output-max`') [21]. This is entirely consistent with the idea we discussed in the course: planning a smooth trajectory in the task space and then handing it over to a lower-level task space controller for tracking; the higher-level controller is responsible for generating a series of key poses (pre-grasp, grasp, lift, fold,

drop) in the task space, and the lower-level OSC ensures that the Panda's joint states are pushed toward these targets given stiffness and damping parameters.

Furthermore, the observation modes provided by the environment (`--robot-observation` optional `ee`, `ctrl`, or `none`) correspond to the course's distinction between "real end-effector pose" and "controller's internal desired pose" [21]:

- When `ee` is selected, the policy/algorithm observes the Panda's real end-effector position in the world coordinate system
- When `ctrl` is selected, the controller's current desired pose is observed;
- Combined with image observation, a joint state space of "vision + end-effector state" can be constructed

Therefore, this environment is designed particularly convenient for our task that on the one hand, `--cloth-size` allows us to systematically change the cloth size, examining the adaptability of visual and geometric methods to size variations within the exact same Panda-desktop-control structure [21]. On the other hand, domain randomization and camera configuration reserve space for subsequent expansion to more complex visual conditions (such as lighting perturbations and viewpoint changes), while maintaining the usage of course content and the style of Drake programming .

B. Motion Planning and Control

1) Folding Geometry and Stage Division: First, let the four corner points of the square cloth in the tabletop coordinate system W be:

$$^{W}p_i = (x_i, y_i, z_{\text{table}})^\top, \quad i = 1, \ldots, 4,$$

where z_{table} is the tabletop height. After detecting or reading these four corner points, we first sort them in counterclockwise order, and then select a pair of opposite corner points as the "original corner point" and "goal corner point" for folding:

$$^{W}p_{\text{o}}, \quad ^{W}p_{\text{g}}.$$

Our folding goal in this paper is to grab a corner and move it to the opposite point on the same side (as shown in Figure 2.), thereby achieving a rectangular fold along the edge, so we divide the entire motion into four stages:

1) Pre-grasp: The end effector aligns at a safe height above the source corner
2) Grasp: The end effector descends along the normal and closes the grippers
3) Fold: The end effector moves the source corner to the target corner along a diagonal trajectory
4) Release: The grippers open near the target corner and slightly retract, allowing the fabric to naturally conform to the tabletop under gravity

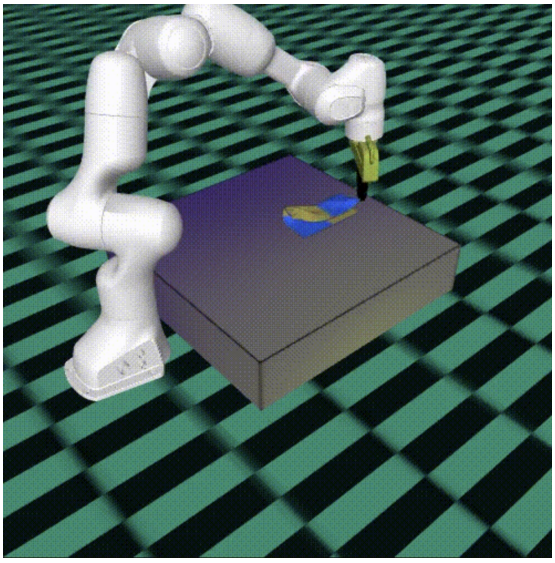

Fig. 2. Definition of a correct fold (The upper left corner was fold to the upper right corner along the upper edge)

2) Task Space Trajectory Generation: In the world coordinate system, the end effector positions $^{W}p_{\text{ee}}(t)$ for each stage of pre-grasp, grip, fold, and release are given by a piecewise continuous trajectory. To simplify the derivation, we use discrete time steps $k = 0, \ldots, N$ to represent the entire folding process.

Pre-grab Pose

Let the unit vector normal to the desktop be $e_z = (0, 0, 1)^\top$, and the pre-grab pose position be

$$^{W}p_{\text{pre}} = {}^{W}p_{\text{o}} + h_{\text{pre}} e_z,$$

where h_{pre} is the safety height. The end effector pose is selected to be aligned with the desktop from top to bottom, meaning the end effector coordinate system's z axis is aligned with $-e_z$, which is consistent with the standard settings for task space pose control in the course.

Crawling Phase

From $^{W}p_{\text{pre}}$, linear interpolation descends to the crawling height h_{grasp}:

$$^{W}p_{\text{grasp}} = {}^{W}p_{\text{o}} + h_{\text{grasp}} e_z, \quad h_{\text{grasp}} < h_{\text{pre}}.$$

The trajectory for this phase can be written as:

$$^{W}p_{\text{ee}}(s) = (1 - s)\, {}^{W}p_{\text{pre}} + s\, {}^{W}p_{\text{grasp}}, \quad s \in [0, 1].$$

Upon reaching $^{W}p_{\text{grasp}}$, the gripper closes and maintains this position for several control cycles.

Folding Phase

The core of the folding motion is to smoothly move the trajectory of the original corner point in space from $^{W}p_{\text{o}}$ to $^{W}p_{\text{g}}$, maintaining a certain height along the way to avoid large-area friction with the tabletop.

Let the folding trajectory parameter $s \in [0, 1]$, and define the in-plane interpolation value as:

$$^{W}p_{\text{edge}}(s) = (1-s)\,{}^{W}p_{\text{o}} + s\,{}^{W}p_{\text{g}}.$$

with the height uses a piecewise function:

$$h(s) = h_{\text{lift}}, \quad 0 \le s \le 1,$$

That is, maintaining a constant height h_{lift} above the tabletop during the overall folding process to reduce the drag between the fabric and the tabletop.

Therefore, the final trajectory is:

$$^{W}p_{\text{ee}}(s) = {}^{W}p_{\text{edge}}(s) + h_{\text{lift}}e_z, \quad s \in [0,1].$$

When $s = 1$, the grab point is exactly above the target corner point.

Release and Retreat Phase

Open the gripper at $^{W}p_{\text{ee}}(1)$, then translate the end a small distance and slightly lift it:

$$^{W}p_{\text{rel}} = {}^{W}p_{\text{g}} + h_{\text{rel}}e_z + \Delta p_{\|}$$

Where $h_{\text{rel}} > 0$, $\Delta p_{\|}$ is a small retreat displacement within the table plane, allowing the fabric to fall naturally to the target position under gravity without being "hooked" by the end.

In numerical implementation, we discretize the above trajectory with a fixed control frequency f_{ctrl}. For example, given a total folding duration T and a time step of $\Delta t = 1/f_{\text{ctrl}}$, the parameters for each step are:

$$s_k = \frac{k\Delta t}{T},$$

and generate

$$^{W}p_{\text{ee},k} = {}^{W}p_{\text{ee}}(s_k).$$

3) Motion Mapping and Task Space Control: The motion space in the environment does not directly control the joints, but rather controls the *increment* of the desired pose of the end effector. Let the current estimated position of the terminal be $^{W}p_{\text{ee}}^{\text{curr}}$. Then, in the k-th control cycle, we calculate the desired position $^{W}p_{\text{ee},k}^{\text{ref}}$ according to the planned trajectory. The action is defined as:

$$\Delta p_k = \text{clip}\big({}^{W}p_{\text{ee},k}^{\text{ref}} - {}^{W}p_{\text{ee}}^{\text{curr}},\ \|\Delta p\|_\infty \le \Delta_{\max}\big)\,,$$

where $\Delta_{\max}$ is the maximum displacement per step limited by the environmental parameter `--output-max`.

This increment serves as the motion input for the Gym. The task space controller within the environment approximates a desired task space velocity $\dot{x}_d$ based on the current joint state q and the Jacobian matrix $J(q)$, and generates joint-level control variables using a task space PD/impedance control law similar to that introduced in the course:

$$\tau = J(q)^{\top}\left(K_p(x_d - x) + K_d(\dot{x}_d - \dot{x})\right),$$

where K_p and K_d are derived from the environmental parameters `--kp` and `--damping-ratio`, respectively. This is completely isomorphic to the task space control of IIWA, except that the solver has been changed from Drake to MuJoCo, and the robotic arm has been changed from IIWA to Panda.

From a high-level planning perspective, we can view this as a combination of "offline trajectory + online error feedback" in the task space: the trajectory $^{W}p_{\text{ee},k}^{\text{ref}}$ provides the geometric objective, while the OSC compensates for small model errors and contact disturbances through the aforementioned control law. This structure ensures that even if local fabric deformation does not perfectly conform to the simplified geometric assumptions, the end effector can still complete the folding action relatively smoothly.

C. Perception and Vision

In this section, we describe how to estimate the four corner points of a square piece of cloth when we alter the size of a square fabric from a monocular RGB image and pass the positions of these corner points as geometric input to the Planning and Control module from the previous section.

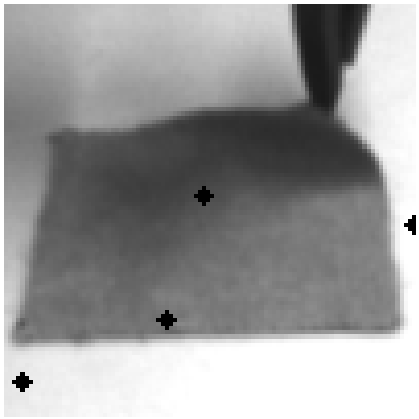

Fig. 3. Example of one raw-conversion to gray scale before corner alignment

1) Camera-Tabletop Homography and Coordinate Relationship: Since the cloth is always laid on the table, we can consider it as a two-dimensional object embedded in the tabletop plane. In class we discussed camera calibration and image-plane homography using a checkerboard or a known plane. In this work, we assume that the homography matrix H from the image plane to the table plane has been obtained through one-time calibration:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim H \begin{bmatrix} u \\ v \\ 1 \end{bmatrix},$$

Where (u, v) are pixel coordinates, (x, y) are table plane coordinates (in meters, located on the $z = z_{\text{table}}$ plane of the world coordinate system), and the symbol $\sim$ indicates that the homogeneous coordinates differ by a non-zero scale factor. The 3D coordinates of the corner points in the world coordinate system can be written as:

$$^{W}p_i = (x_i, y_i, z_{\text{table}})^{\top}$$

This step transforms the visual problem into the detection problem of the pixel coordinates of the four corner points on the image plane.

2) Cloth Region Segmentation and Outer Contour Extraction: Given a calibrated RGB image with a top-down or slightly tilted viewpoint, we first separate the cloth from the table background in the image domain by converting the RGB image to grayscale, which is easier to identify the depth color differences by the darkness of the edges (example of one raw-conversion to gray scale shown above in Figure 3.). This section employs the simple and robust binary segmentation scheme introduced in the course:

- Convert the RGB image to grayscale to enhance the separation of brightness/saturation and color components
- Based on the color darkness differences between the cloth and the tabletop, select one or more channels for thresholding to obtain an initial binary mask $M(u, v) \in \{0, 1\}$
- Perform morphological opening and closing operations on the mask to remove isolated noise points and small holes
- Select the largest connected region from the mask as the cloth region R_{cloth}

On the cloth region, we apply Canny edge detection and contour tracking to obtain the outer contour curve of the cloth

$$C = \{(u_j, v_j)\}.$$

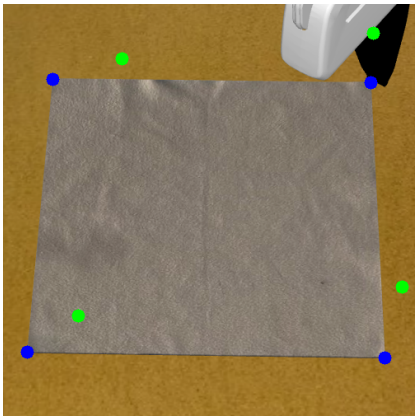

Fig. 4. Visualization of square fabric detection before corner mapping to the desktop coordinate system

3) Polygon Approximation and Four Corner Detection: Since the fabric on the table is approximately a convex quadrilateral (square fabric may have slight undulations, but the overall outline is still close to a rectangle), we can use polygon approximation and rectangle fitting to obtain the four corner points, which steps as follow:

1) Simplify the contour point set C using the Ramer–Douglas–Peucker algorithm to obtain an approximate polygon with fewer vertices

$$P = \{(u_k, v_k)\}_{k=1}^{K}$$

where K is usually much smaller than the original number of contour points
2) Apply geometric filtering to P, which first calculates the angle between adjacent sides, retaining vertices close to right angles (e.g., $90^\circ \pm 20^\circ$)
3) Calculate the side lengths between candidate vertices, filtering out excessively short noise edges
4) If multiple candidate sets still exist after filtering, further fit C using the minimum bounding rectangle, using the four vertices of the fitted rectangle as the final corner points $(u_i, v_i), i = 1, \ldots, 4$

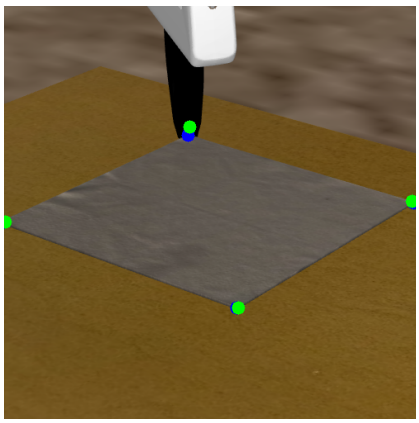

Fig. 5. Visualization of square fabric detection after corner mapping to the desktop coordinate system

4) Mapping of pixel corner points to desktop coordinates: After obtaining the image plane corner points (u_i, v_i), we use the aforementioned homography matrix H to map them to the desktop coordinate system. For each corner point,

$$\begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \\ \tilde{w}_i \end{bmatrix} = H \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}, \quad x_i = \frac{\tilde{x}_i}{\tilde{w}_i}, \quad y_i = \frac{\tilde{y}_i}{\tilde{w}_i}.$$

Then, again, construct the three-dimensional coordinates:

$${}^{W}p_i = (x_i, y_i, z_{\text{table}})^{\top}, \quad i = 1, \ldots, 4.$$

Thus, we have recovered the three-dimensional positions of the four corners in the world coordinate system from visual observation, which is directly compatible with the planning and control module in the previous section. Figure 4. shows the camera view before the corner mapping to the desktop coordinate system and Figure 5. shows the camera view after the corner mapping, which shows the mapping was pretty accurate and successful as the corner points are well overlapped.

5) Corner Sorting and Grabbing/Collapse Geometry Interface: To connect with the planning equations in the previous subsection, we need one more step to sort the four corner points consistently and select the "source corner point" and the "target corner point". The method is as follows:

1) Calculate the geometric center of the four corner points:

$$W_c = \frac{1}{4}\sum_{i=1}^{4} {}^{W}p_i$$

2) Denote the vector of each corner point relative to the center in the desktop plane coordinate system as

$$d_i = \begin{bmatrix} x_i - c_x \\ y_i - c_y \end{bmatrix},$$

and calculate its polar angle

$$\theta_i = \text{atan2}(y_i - c_y,\ x_i - c_x)$$

3) Sort the corner points according to their polar angles to obtain four points arranged in counterclockwise order: $\{{}^{W}p^{(1)}, \ldots, {}^{W}p^{(4)}\}$

4) Select one of the corner points as the source corner point (e.g., the one furthest to the left and bottom relative to the x-axis of the world coordinate system), denoted as:

$${}^{W}p_{\text{o}} = {}^{W}p^{(k)},$$

Its opposite corner point on the same edge would then be:

$${}^{W}p_{\text{g}} = {}^{W}p^{(k+1 \bmod 4)}$$

After this, ${}^{W}p_{\text{o}}$ and ${}^{W}p_{\text{g}}$ are directly fed into the folded geometry defined in the previous section such that the grasping phase uses the pre-grab/grab pose above ${}^{W}p_{\text{o}}$, the folding phase follows the edge of ${}^{W}p_{\text{o}} \rightarrow {}^{W}p_{\text{g}}$ generates the end trajectory, and during the release phase, the grippers open near ${}^{W}p_{\text{g}}$, allowing the fabric to naturally conform to the tabletop.

At this point, the vision module and the planning-control module are connected through a clear geometric interface.

D. Experiments Setup

To systematically validate the proposed fabric folding framework, we designed two progressively layered experimental environments to evaluate control and planning capabilities and vision-planning coupling capabilities respectively.

1) Experiment 1: Planning Control Tests: In the first set of experiments, the fabric size was fixed at 0.25m, and the 3D coordinates of the four corner points were directly read from the simulation environment. The planning module used the single-sided folding trajectory generation method defined in the previous section, and the control module employed task-space PD/impedance control to execute the folding action. We generate 10 different environments by the system to test for the success rate.

2) Experiment 2: Perception and Vision Tests: The second set of experiments introduced a vision module and simulated a more realistic application scenario—inconsistent fabric sizes. We maintained the basic orientation of the fabric consistent, but randomly varied its side lengths in ten trials (0.125 m, 0.15 m, 0.175 m, 0.2 m, 0.225 m, 0.25 m, 0.275 m, 0.3 m, 0.325 m, 0.35 m). In each trial, the system first detected the four corner points of the fabric from a monocular RGB image, mapped them to the world coordinate plane using a homography matrix, and then used the planning module to generate the folding trajectory. The remaining execution flow (pre-grab, grab, fold, release) was completely consistent with Experiment 1, allowing for a direct comparison of the performance differences between "known corner points" and "visually estimated corner points".

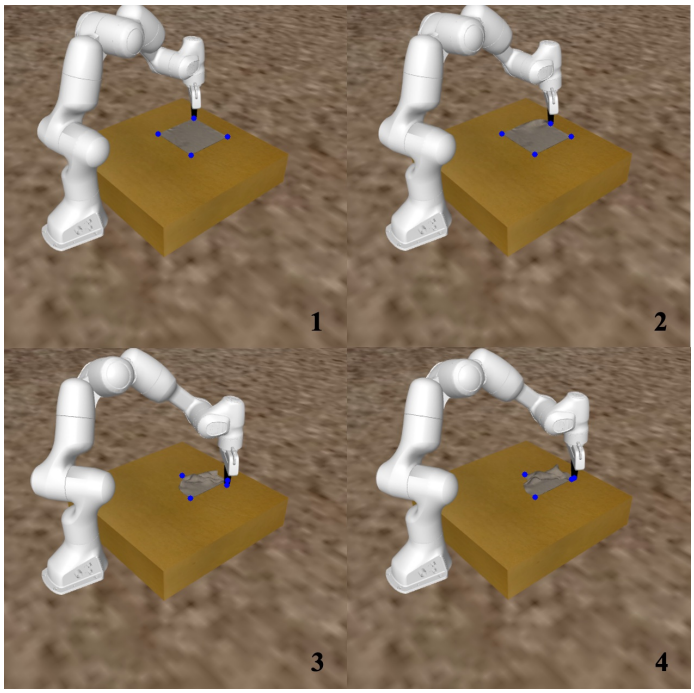

Fig. 6. Trajectories of Folding in One Successful Trial

IV. Results

A. Experiment 1: Planning Control Tests

In the first set of experiments, the system directly used the four corner points provided by the simulation environment and added random perturbations to the initial state of the fabric with side length of 0.25 m. We conducted 10 independent folding experiments under random setups.

Figure 6. shows the visualization of the folding trajectory of one successful trial, including the stages of pre-grabbing, grasping, lifting, edge folding, and release.

Figure 7. shows all the end-effectors in 10 random generated environments of experiments 1 under side length of 0.25 m. Notice that Trials 1, 2, and 3 were not successful as the corner point did not exactly fold on the opposite corner point across the edge.

Table 1. below statistically and formally exhibits the test results of the experiment 1 under 10 random generated test cases environments. For the trials that were not successful, we labeled the stage that was not successful which causes the unsuccessful fold.

In this set of experiments, 7 of the all 10 experiments successfully completed the unilateral folding operation, with a success rate of 70%. In the succeeded trials, the folding

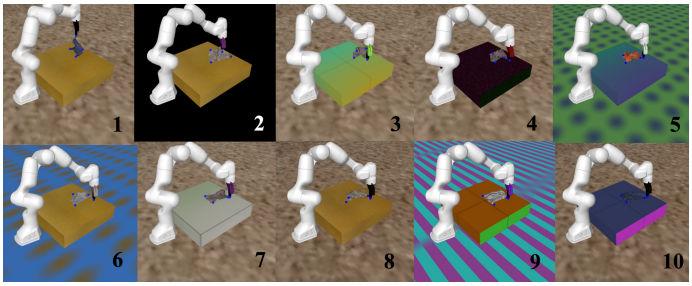

Fig. 7. End-effectors of the 10 Random Generated Environments for Test Cases under constant side length of 0.25 m (Successful: 4, 5, 6, 7, 8, 9, 10; Unsuccessful: 1, 2, 3)

Trial Number	Success	Probematic Stage
1	F	Release
2	F	Fold
3	F	Pre-grasp
4	T	-
5	T	-
6	T	-
7	T	-
8	T	-
9	T	-
10	T	-
10 Trials	**7 Successes (70%)**	

TABLE I
EXPERIMENT 1 TEST RESULTS AND SUCCESS RATE

termination state was stable, and the fabric naturally adhered to the tabletop after release, with a smooth trajectory and no obvious tension accumulation.

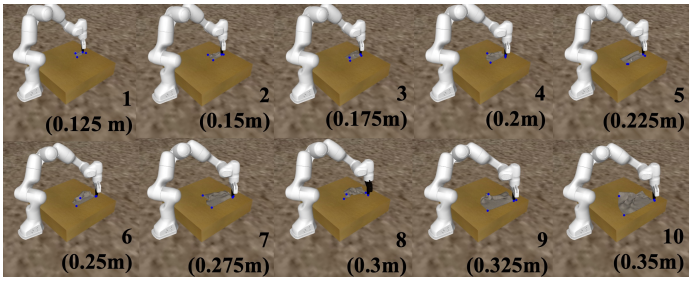

Fig. 8. End-effectors of Folding 10 Square Fabrics with 10 Different Side Lengths under the same environment (Successful: 2 (0.15 m), 3 (0.175 m), 4 (0.2 m), 5 (0.225 m), 6 (0.25 m), 7 (0.275 m), 8 (0.3 m), 9 (0.325 m), 10 (0.35 m); Unsuccessful: 1 (0.125 m))

B. Experiment 2: Perception and Vision Tests

In the second set of experiments, the environment was set constant, but fabric size varied under 10 different side lengths (0.125 m, 0.15 m, 0.175 m, 0.2 m, 0.225 m, 0.25 m, 0.275 m, 0.3 m, 0.325 m, 0.35 m), hence the corner points would have different locations, which can test the detection efficiency of the vision module.

Figure 8. exhibits the end-effector of the 10 trials of 10 different side lengths of the square fabric all under the same environment. Note that trials 2-10 all had very successful fold that left upper corner point was fold exactly on the right upper corner point along the same edge. Trial 1 with side length 0.125 m, which is the smallest square fabric in size.

Table 2. statistically and formally documents the 10 trials with 10 different square fabric side lengths under the constant environment and their corresponding success.

Trial Number	Side Length (m)	Success
1	0.125	F
2	0.15	T
3	0.175	T
4	0.2	T
5	0.225	T
6	0.25	T
7	0.275	T
8	0.3	T
9	0.325	T
10	0.35	T
10 Trials		**9 Successes (90%)**

TABLE II
EXPERIMENT 2 TEST RESULTS AND SUCCESS RATE

The overall success rate of the 10 trials in experiment 2 is 90%, with only trial 1, the square fabric with the smallest size, was unsuccessful. The unsuccessful stage, based on the video, was the folding stage. Detailed evaluations and discussions of the above results will be explained in the next section.

V. DISCUSSION

In this section, we analyze and discuss the proposed square fabric folding framework, focusing on the experimental results, the advantages and disadvantages of the method, the overall conclusion of this work, and future work.

A. Results Analysis

To summarize, this paper comprises two progressively layered experiments:

1) Under known corner points and fixed dimensions, single-sided folding is achieved solely through planning and control
2) Under varying fabric dimensions, visual corner point detection is introduced, and folding is driven by a closed-loop system

In Experiment 1, we conducted 10 folding trials on a square fabric of fixed dimensions, achieving 7 successes, a success rate of 70%. Failures mainly occurred in extreme cases at the contact edges: when the gripping point was close to the fabric edge or the initial slight wrinkles were large, the open-loop trajectory was more sensitive to local deformation, resulting in the final corner point slightly exceeding the tolerance from the target position.

In Experiment 2, the fabric posture remained similar, but folding was performed at 10 different dimensions. All corner points were automatically detected by the vision module and mapped to the world coordinate system, then the planning and control module generated the trajectory. 9 out of 10 trials were successful, increasing the success rate to 90%. This indicates that, under our current setup, the interface between geometric vision and planning-control is relatively stable, and corner detection errors are within a range that can be tolerated by

impedance control. The failure was still primarily stem from visual false detections (a corner is offset by a broken outline or shadow), introducing systematic biases at the geometric level, and this error was enlarged when the size of the fabric is small.

An additional and consistent observation is that the larger the fabric size, the longer the average time required to complete one fold. This is because in our trajectory design, the end point uses linear interpolation along the path from the source corner to the target corner, and the folding displacement is approximately proportional to the side length. With a fixed control frequency and maximum single-step displacement limit, the trajectory deviation number increases linearly with fabric size, leading to an increase in overall operation time as size increases. This trend is particularly evident in the comparison of different sizes in Experiment 2.

B. Evaluations

Overall, the framework we proposed in this paper has the following advantages:

1) Clear modularity and strong interpretability. The system implements deformable body modeling, motion planning, and visual perception in layers. In the first part of the experiment, stable folding is achieved solely through planning and control; in the second part, a vision module is added to adapt to size changes. Compared to end-to-end methods, this structure facilitates the identification of positioning error sources and modular improvements.

2) Combination of physical modeling and compliant control. Regardless of the simulation backend used, we model the robotic arm and cloth within a unified dynamic framework and handle contact and deformation through a task space control strategy similar to impedance control. This makes it easier to analyze the causes of failure than completely black-box policy learning and also provides a foundation for the future introduction of differentiable simulation or MPC.

3) The phased experimental design facilitates quantitative analysis. Through the design of "planning and controlling first, then adding vision", the theoretical upper limit of the system and the actual performance after visual intervention can be clearly distinguished, thus revealing the bottleneck effect of the vision module.

However, our framework also has the following limitations:

1) The task object is relatively simple. The geometric topology of square fabric is simple, but it still differs from the structure of real clothing (such as sleeves and collars), making it less difficult than the task of folding complete clothing.

2) Visual methods are sensitive to scale changes. Corner detection relies on geometric heuristics and proportional assumptions, making it sensitive to camera intrinsics, viewpoint, and depth noise. Errors are significantly amplified when scale changes. Compared to methods utilizing dense correspondences or semantic regions, it lacks robustness.

3) The task is still a single-step folding. It does not involve multi-step folding, repeated folding, or multi-fabric interaction, which are crucial in real clothing handling processes.

C. Conclusion

The combined results of the two sets of experiments show that, under the conditions of fixed dimensions and known corner points, our planning-control pipeline can complete the single-sided folding of a square fabric with a success rate of 70%. By introducing visual corner point detection and allowing for variations in fabric size, the overall success rate increases to 90%, indicating that the framework combining geometric vision and compliant control is feasible and robust under the current task and error levels. Simultaneously, we also observed that fabric size significantly affects trajectory length and operation time, suggesting that future research needs to make a more nuanced trade-off between time efficiency and geometric accuracy.

D. Future Directions

Based on the above analysis, future work can be done in the following directions:

1) From square fabric to real clothing. Gradually introduce clothing with semantic components, such as T-shirts and pants, into the existing framework, and introduce folding templates and intermediate standardized states based on region semantics.

2) Improve the scale and deformation robustness of the visual module. Building on the current geometric pipeline, combine multi-scale features, dense correspondences, or simple learning modules to improve the stability of corner detection under large-scale changes and complex lighting conditions.

3) Introduce closed-loop vision-force feedback and online adjustment. Use vision and force sensing for online fine-tuning during execution to compensate for deformation errors during folding; simultaneously consider incorporating "folding time" into the planning objective, optimizing path length and execution time while ensuring accuracy, especially in large-size fabric scenarios.

ACKNOWLEDGEMENT

This project is the result of equally joint work by Youhui (Jeffrey) Wang and Hangxing Zhang. While all the work are discussed, completed, and polished by both contributors, Youhui (Jeffrey) Wang was intended to focus on the motion planning and control part of this project and Hangxing Zhang was intended to focus on the learning part of this project that tackles arbitrary initial conditions cases. When either contributor faced any challenges, both contributors will come together, discuss, and figure out collaboratively. Additionally, the initial planning, this paper, and the final presentation and the video are all done together.

This work is intended for the final project of the course MIT 6.4212 Robotic Manipulation in Fall 2025 Term. We appreciate all the assistance and guidance of the course staffs and Professor Tomas Lozano-Perez on this project and throughout the term.

REFERENCES

[1] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[2] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[3] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[4] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep. 98-11, Iowa State University, 1998.

[5] R. Tedrake, *Robotic Manipulation: Perception, Planning, and Control*. MIT Course Notes for 6.421, 2024. [Online]. Available: https://manipulation.mit.edu

[6] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

[7] J. van den Berg, S. Miller, K. Goldberg, and P. Abbeel, "Gravity-based robotic cloth folding," *Algorithmic Foundations of Robotics IX*, Springer Tracts in Advanced Robotics, vol. 68, pp. 409–424, 2010.

[8] S. Miller, J. van den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, "A geometric approach to robotic laundry folding," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 249–267, 2012.

[9] S. Bersch, P. Azad, T. Uelker, and R. Dillmann, "Bimanual robotic cloth manipulation for laundry folding," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 1413–1419.

[10] J. Stria et al., "Garment perception and its folding using a dual-arm robot," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 61–67.

[11] V. L. De Gusseme, R. Vanherle, and J. De Schutter, "Effective cloth folding trajectories in simulation with only provisional grasping," *Frontiers in Neurorobotics*, vol. 16, art. 989702, 2022.

[12] M. Shehawy, A. Rana, J. Hernandez-Gonzalez, and E. Chinellato, "Flattening and folding towels with a single-arm robot based on reinforcement learning," *Robotics and Autonomous Systems*, vol. 160, p. 104300, 2023.

[13] R. Hoque et al., "Learning to fold real garments with one arm: A case study in cloud-based robotics research," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[14] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held, "FabricFlowNet: Bimanual cloth manipulation with a flow-based policy," in *Proc. Conference on Robot Learning (CoRL)*, 2021, pp. 192–202.

[15] D. Seita et al., "Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 9651–9658.

[16] H. A. Kadi and K. Terzić, "Data-driven robotic manipulation of cloth-like deformable objects: The present, challenges and future prospects," *Sensors*, vol. 23, no. 5, art. 2389, 2023.

[17] V. Longhini, W. Yuan, and M. Tognon, "Unfolding the literature: A review of robotic cloth manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 8, 2025.

[18] Y. Li, T. Du, K. Wu, J. Xu, and W. Matusik, "DiffCloth: Differentiable cloth simulation with dry frictional contact," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 1–17, 2022.

[19] N. Sunil, S. Sundaralingam, A. Yao, and D. Fox, "Reactive in-air clothing manipulation with confidence-aware dense correspondence and visuotactile affordance," in *Proc. Conference on Robot Learning (CoRL)*, 2025.

[20] Z. Zhou, Z. Wang, and D. Held, "Learning efficient robotic garment manipulation with standardization," *arXiv preprint*, arXiv:2503.xxxxx, 2025.

[21] J. Hietala, "Dynamic Cloth Folding," GitHub repository, https://github.com/hietalajulius/dynamic-cloth-folding. Accessed: Jan. 15, 2025.