Automated Folding of a Deformable Thin Object through Robot Manipulators

Zhenxi Cui, Kaicheng Huang, Bo Lu and Henry K. Chu*, Member, IEEE

***Abstract*— This paper presents a model-free approach to automate folding of a deformable object with robot manipulators, where its surface was labelled with markers to facilitate vision-based control and alignment. While performing the task involves solving nonconvex or nonlinear terms, in this paper, linearization was first performed to approximate the problem. By using the Levenberg–Marquardt algorithm, the task of folding a deformable thin object can be reformulated as a convex optimization problem. The mapping relationship between the motions of markers on the image and the joint inputs of the robot manipulator was evaluated through a Jacobian matrix. To account for the uncertainty in the matrix due to the deformable object, a two-stage evaluation scheme, which consists of approximate-rigidity rule and Broyden-update rule, was performed. Proper constraints were also added to avoid causing damage to the object. The performance and the robustness of the proposed approach were examined through simulation using Bullet simulator. The video of the simulation can be retrieved from the attachment. The results confirm that the thin object can be precisely folded together based on different markers labelled on the surface.**

I. Introduction

The ability to manipulate soft and delicate objects precisely with robotic manipulators can help to imitate or automate common tasks that need to be performed by humans. For instance, tying surgical knots (1D soft objects) in a surgical operation, folding clothes or garments (2D soft objects) in a laundry room, and kneading dough (3D soft objects) in a kitchen are just a few examples that remain challenging for robots to outperform humans. In contrast to rigid objects, manipulation and handling of soft objects require more considerations. The reasons for manipulation inaccuracy not only come from the manipulators and the sensory feedback but are also due to uncertainty or lack of prior knowledge on the model parameters of the soft objects. To tackle this problem, researchers in the field have proposed various solutions, including model approximation, learning through demonstrations and constrained path planning. A summary on non-rigid object manipulation and recent advancements can be found in [1] [2].

In this paper, the task needed to be tackled is shown in Fig.1, which is folding a thin and transparent object (sheet) according to the markers. It inherits all of the aforementioned challenges with additional constraints such as object transparency and fragility. To perform the task, two articulated robot arms, equipped with suction probes, are commanded to grasp and fold the sheet based on the marker information from a vision camera. Due to object transparency in nature, no other properties or information can be obtained directly. The control algorithm aims at guiding markers from their current positions to the desired positions as fast as possible by minimizing the difference between the two measurements. In addition, self-collision between the manipulating tools and over-stretching of the sheet should be avoided.

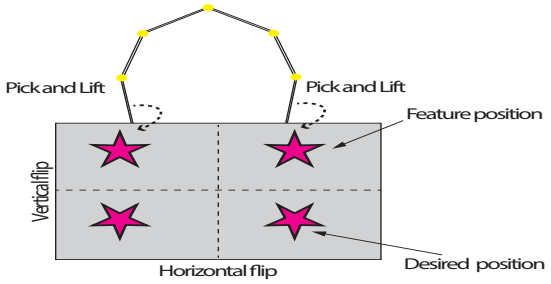


Fig. 1: 3D folding using two articulated arms

A. related works

Because of the dynamic and nonlinear properties within a soft object, establishing an accurate model to describe the deformation behavior is the first and the most crucial step in precise object manipulation. In the field, there are several techniques that can compute or simulate the deformation model before the actual manipulation. Das et al. [3] modelled the rheological deformation with interconnected mass-spring-damper elements to facilitate shape control with multiple manipulators. Li et al. [4] simulated the movement and deformation of a garment using a mesh model. Shear resistance and frictional force were first measured to ensure the accuracy between the simulation result and the real garment manipulation. Based on the information, the optimized folding trajectory was generated to achieve efficient manipulation of the garment. Comparing with the mass-spring-damper model, the finite element method is a more computationally intensive approach to provide better accuracy. The general concept is based on the thin-shell theory to govern the deformation behavior [1]. The research by Duenser et al. [5] was further elaborated to couple the robot with the elastic object as a finite element model. Through proper mapping between the change in the robot configuration and the deformed shape, precise shape control can be achieved. Another stream of the research lied on force analysis. Bai et al. [6] modelled the deformation as flexible payloads on two arms and treated as formation-control problem. In additional

The work was supported in part by the Research Grant Council of the Hong Kong Special Administrative Region, China, under Grant 25204016.

Z. Cui, K. Huang and H. K. Chu are with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong. (* Corresponding Author, email: henry.chu@ polyu.edu.hk) B. Lu was with the The Hong Kong Polytechnic University, Hong Kong. He is now with the T stone Robotics Institute, The Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong.

to the above models, deformation models based on other principles were also reported in [7] [8] [9]. No matter which deformable model is selected, the model or its modelling parameters need to be adaptive to the unpredicted changes in the object properties and the environment during the entire manipulation. To overcome this problem, some researchers focus on the adaptive estimation of deformable behavior. For instance, Navarro-Alarcon et al. [10] proposed a method that can compute the unknown deformable parameters in a quasi-static object model in real-time. The deformable 3D object was visually manipulated into its desired shape with two arms. Later, Navarro-Alarcon et al. [11] enhanced the feedback method by representing the object's 2D shape based on the truncated Fourier series and combining with a calibration scheme to estimate the local deformable model. In contrast to the aforementioned methods, various model-free and machine-learning methods were also examined the field. In [12] and [13], a PID feedback control law based on a rough object model or model-free were examined and the error can converge if the deformation is not large. Jia et al. [14] developed a dictionary-based method and a dictionary containing key features of different deformable objects was prepared offline through a RGB sensor. In their work, in addition to cloth folding, a set of cloth manipulation tasks and collaborative manipulation were also evaluated. Hu et al. [15] proposed using Fast Online Gaussian Process Regression algorithm to learn the unknown deformable parameters for predicting the required control at each time step. Manipulation tasks, such as towel folding and bending, were validated through experiments. Cherubini et al. [16] proposed a vision-based method for shaping deformable plastic materials based on data and learning-based algorithms. Their work focuses on non-prehensile manipulation actions on a plastic material, rather than shape control. A different approach was proposed by Berenson [17], which is based on the concept of diminishing rigidity to approximate the Jacobian of the deformable object. Rope winding, cloth spreading and cloth folding were successfully performed through simulation with the considerations of collision avoidance and overstretching.

B. contribution

In this paper, we proposed a model-free method to servo-control two articulated robot arms to fold a soft and thin object using visual information. Similar to classical image-based visual servoing tasks, the required joint inputs to the robot arms can be formulated as an optimization problem. While classical visual servoing controllers do not address accurate soft object deformation, this paper aims as coupling the two robot arms with the soft object as a system. Additional constraints related to the task, such as joint limits, overstretching limits and shape requirements are also taken into consideration, which could cause the objective function to become non-convex and nonlinear. In this context, we proposed to apply a linearization method to approximate the function such that the new function can be efficient for computation and can achieve fast convergence. To account for the uncertainty and system reliability, the Jacobian matrix that maps between the motions of the markers and the joint inputs was evaluated using a two-stage scheme for evaluation and refinement. Simulations were performed to evaluate the performance of the proposed algorithm for soft object handling. Different markers were labelled on the object surface and the object was successfully folded in two halves or arbitrarily along different folding axes, demonstrating the effectiveness and the robustness of the algorithm. The flexibility of this approach (arbitrary selection on the folding axes) can save time in preparing a large amount of data for training and learning.

II. Problem and Method

Considering two multi-degree of freedom (DOF) articulated manipulators with a total n joint variables, where $\theta_1 \cdots \theta_n, \theta_i \in R^1$ represent the joint rotation angle. $p_1 \cdots p_m, p_j \in R^3$ are m feature points to be controlled by the manipulators, expressed in the Cartesian coordinate system. Similarly, there are another m points, $t_1 \cdots t_m, t_i \in R^3$ that define the desired positions for these feature points. The selection of these desired points also defines the final configuration of the deformable object. The objective is to guide the features points from their initial position toward their desire position by changing the joint angles. In other words, the current position can be expressed as a function of joint inputs and the problem becomes:

$$t_i = p_i(\theta) \tag{1}$$

Let $e_i = t_i - p_i, i = 1 \cdots m$ represent the errors between the current position and the desire position for m points in the Cartesian. Define $F : R^n \longmapsto R^{3m}$ as the function that maps the joint space with the Cartesian space. The control objective essentially become to find the required joint inputs $\Delta\theta$ such that the errors in the Cartesian go to zero, and the loss function used for optimization can be represented as:

$$e = t - p = F(\Delta\theta) = 0 \tag{2}$$

Notice that the joint inputs, $\Delta\theta$, $\Delta\theta \in R^n$, should be limited to within the operable range, which is constrained by factors as the joint limits, the workspace within the field of view of the camera, material overstretching limit and collision avoidance, etc. All constraints must be satisfied before deriving the control inputs.

A. LM Algorithm

In theory, if the end effector of the manipulator is the feature point to be tracked, the mapping relationship can be easily and accurately obtained by solving the forward kinematics. However, in the current task, the feature points are located far away from the end effector and arbitrarily on a deformable object with unknown properties. The deformability and the uncertainty would lead to nonlinearity in the mapping relationship, which cannot be derived directly and the accuracy cannot be guaranteed. To resolve the nonlinearity, one method is to linearize the complicated mapping relationship through the Taylor series. To simplify

the equation, only the first linear term is kept and all higher-order terms are dropped and the relationship becomes:

$$F(\theta) \approx F(\theta) + J(\theta) * \Delta\theta$$
$$J(\theta) = \left[\frac{\partial F(\theta)}{\partial \theta_1} \cdots \frac{\partial F(\theta)}{\partial \theta_n}\right] \tag{3}$$

Here, the $J \in R^{3m\times n}$ can be viewed as the cascaded matrix that contains two robot arms and the object deformation, which are now termed as Jacobian.

Consider the problem as m pairs (t, θ) of independent and dependent variables and $\Delta\theta$ as the model parameter. The optimal model parameter is the one that can minimize the sum of squared error between the feature points and desired positions. In other word, the problem can be reformulated as a Levenberg–Marquardt (LM) problem which can be expressed as:

$$S(\Delta\theta) \approx \sum_{i=1}^{m} \left[t_i - F_i(\theta) - J_i(\theta) * \Delta\theta\right]^2 \tag{4}$$

or in vector notation,

$$S(\Delta\theta) \approx \left\|t - F(\theta) - J(\theta) * \Delta\theta\right\|^2 \tag{5}$$

Now equation 5 can be utilized as the new objective function to fit in the optimization framework. An iterative approach can be used for solving equation 5, After expansion, taking derivative of $S(\Delta\theta)$ with respect to $\Delta\theta$ and setting it to zero, the equation becomes:

$$\left(J^T J\right) \Delta\theta = J^T \left[t - F(\theta)\right] \tag{6}$$

Multiplying the inverse of $J^T J$ on both sides, the equation can be rewritten as $\Delta\theta$, which represents the required joint inputs in the next iteration.

B. Two Stage Jacobin Update Rules

In Section IIA, $\Delta\theta$ can be solved using the LM algorithm. In this section, the required Jacobian can be obtained using a novel update scheme. When the error criteria (norm error between feature points and desired point) remains large, the entire system can be considered as a rigid body and the "approximate rigidity" rule is utilized to describe the Jacobian. On the contrary, when the error is small, a quasi-Newton method, also known as Broyden method, is used to update the Jacobian to include the effect from the deformation. Based on the two-stage update rule shown in figure 2, our method can handle the deformation of soft materials with high convergence and high accuracy as demonstrated in the experimental section.

1) Approximate Rigidity : For an n-link manipulator with joint variables, define o as the origin offset between the $n-th$ coordinate frame (end-effector) and the base coordinate frame. The homogeneous transformation matrix can be expressed as:

$$T_n^0(\theta) = \begin{bmatrix} R_n^0(\theta) & o_n^0(\theta) \\ 0 & 1 \end{bmatrix}, \theta = [\theta_1 \cdots \theta_n]^T$$

For a point p located with respect to the $n-th$ coordinate frame, its coordinates with respect to the base $0-th$ coordinate frame can be computed as:

$$p^0 = R_n^0(\theta)p^n + o_n^0 \tag{7}$$

The rate of change of this point can be evaluated by taking the time derivative to become:

$$\frac{d}{dt}p^0(t) = S(\omega_{0,n}^0(t))R_n^0(\theta)p^n + \upsilon_n^0$$
$$= \omega_{0,n}^0(t)) \times r(t) + \upsilon_n^0 \tag{8}$$

Based on equation 8, we can get the linear velocity part as:

$$\upsilon_n^0 = \frac{d}{dt}o_n^0(t) = J_{\upsilon_i}\dot{\theta}_i$$
$$J_{\upsilon_i} = \begin{cases} z_{i-1}^0, & \text{for prismatic joint.} \\ z_{i-1}^0 \times \left[o_n^0 - o_{i-1}^0\right], & \text{for revolute joint.} \end{cases} \tag{9}$$

With reference to [17], there is an exponentially diminishing relationship in the rigidity between the end-effector to the point which is far away. Here we use a linearity method to approximate the deformation Jacobian. Assume the robot manipulator is known and its Jacobian can be evaluated numerically or analytically. Define $r_i, i = 1\cdots m$ as the vectors pointing from the end-effector to different feature points $p_i, i = 1\cdots m$ and the feature point Jacobian can be approximated as:

$$J_{p_i} = J_{end} + z_{0,i-1}^0 \times r_i \tag{10}$$

The Jacobian derived using equation 10 could be deviated from the true value, and so it should be used as an initial estimation and updated accordingly as discussed in the next section.

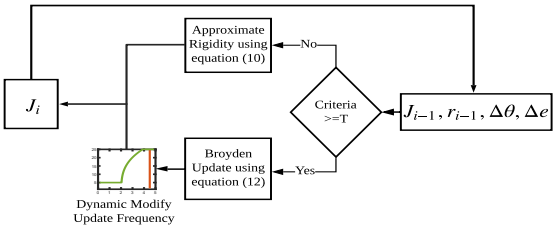
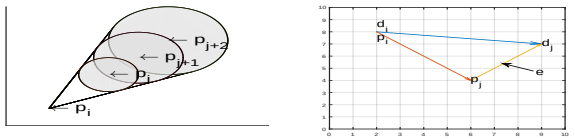


Fig. 2: Two stage Jacobian update scheme

2) Broyden Update Rules : For a time-varying Jacobian, the idea of the Broyden's method is to compute the Jacobin matrix only at the first iteration, and it can be updated using newer input and output data. The use of Broyden's method for online estimation of the Jacobian has also been reported [18] [19]. In each iteration, as the joint inputs and feature coordinates change, indexed with the subscript i ,and represented by $\Delta\theta_i$ and Δe_i, these data are substituted into equation 11. Because the last Jacobin indexed with subscript $i-1$, which can be retrieved at time i, we can update the old Jacobian to a new one which is more accurate for current Jacobin estimation. Normally, to lessen the influence from noise and measurement error, a weighting term α is utilized to adjust the weighting between update rate and the accuracy.

$$J_i = J_{i-1} + \frac{\Delta e_i - J_{i-1}\Delta\theta_i}{\left\|\Delta\theta_i\right\|^2}\Delta\theta_i^T \tag{11}$$
$$J_i = J_{i-1} + \alpha\Delta J \tag{12}$$



(a) Cone constraint (b) Shape maintenance

Fig. 3: Graphical representation on the overstretching and shape distortion constraints

In summary, when the norm error e smaller than the threshold T in figure 2 , the algorithm automatically change from branch of approximate rigidity into branch of broyden's update. Furthermore, we introduce a argument f to dynamic adjust the update frequency in branch of broyden's update. To be more specific, it can control the interval number of the sampling times between two successive jacobian update. The argument f is function of norm error e and is design as:

$$f = 20 * e^{1/3}, 5 \leq f \leq 25 \tag{13}$$

Considering the computation burden, there is a upper and lower bound on f. When the norm error e gradually approach the zero, and the update frequency will increase accordingly. Hence, the estimation accuracy gradually increase along with the decrease of the norm error.

C. Constraints Handling

Following the equations 5, 10, 11, and 12, the folding task can be performed with any type of robot manipulator. The required joint inputs can be computed according to the $\Delta\theta$ such that the feature points on the deformable object will be gradually manipulated to the desired positions in each iteration. The two-stage Jacobian update rules will ensure the accuracy and the correctness of the matrix.

Note that the computed value $\Delta\theta$ should be limited within an operable range, which is bounded by the task specific constraints. The common constraints include joint limits, collision avoidance, overstretching avoidance and filed of view (of the camera). In addition, the presence of obstacles and contact dynamics will also add additional constraints to the system.

1) Self-collision and overstretching avoidance constraints: When an object is held by two robot manipulators, it should always maintain a safe distance between the two contact points no matter how the robots are manipulated. At time t, after the joint inputs, the distance between the two points should be:

$$\left\|(p_i + \upsilon_i^0 t) - (p_j + \upsilon_j^0 t)\right\| \geq d_{min} \tag{14}$$

To illustrate the problem clearly, we express the configuration of p_j with respect to p_j. With p_i is shown as a single point, a circle with a radius of p_i is drawn around p_j. For all possible p_j that satisfy minimum distance, it can be observed that their union lies on a conical shape region. However, the complementary set of the cone that is the feasible range is a non-convex set. It will make the problem to become a non-convex and non-deterministic hard (NP-hard) problem. One popular method [20] [21] is to relax the non-convex problems into convex problem, and utilize conventional convex solver [22] [23] [24] to solve the problem.

The other constraint in the task is to avoid overstretching, which set the limit on the maximum distance at time t, represented as:

$$\left\|(p_i + \upsilon_i^0 t) - (p_j + \upsilon_j^0 t)\right\| \leq d_{max} \tag{15}$$

When solving equation 5 with the consideration of these two constraints, it would become a Least Square with Quadratic Inequality constraint (LSQI) problem. The LSQI problem is related to the ridge regression problem, which is shown as:

$$\left\|(p_i + \upsilon_i^0 t) - (p_j + \upsilon_j^0 t)\right\|^2 \leq d_{max}^2 \tag{16}$$

Instead of directly using these non-convex constraints, we first linearize them in the form of $C * \Delta\theta = d, C \in R^{3m\times n}, d \in R^{3m\times 1}$. Combining with our objective function equation 5, we can use the active-set method to solve for them.

$$\upsilon_i^0 = J_{\upsilon_i}\dot{\theta}_i \tag{17}$$

$$\upsilon_i^0 t = J_{\upsilon_i}\dot{\theta}_i t = J_{\upsilon_i}\Delta\theta \tag{18}$$

Taking equation 14 as an example, $(p_i + \upsilon_i^0 t)$ can be formulated into $(p_i + J_i\Delta\theta)$. Then the constraints in Equation 14 can be reformulated as:

$$(p_i + J_i\Delta\theta) - (p_j + J_j\Delta\theta) \geq d' \tag{19}$$

$$d' = \frac{p_i - p_j}{\|p_i - p_j\|} * d \tag{20}$$

Rearranging the p_i, p_j terms to the right hand side and merging the J_i, J_j together, the equation can be rewritten as:

$$(J_i - J_j)\Delta\theta \geq \frac{p_i - p_j}{\|p_i - p_j\|} * d - p_i + p_j \tag{21}$$

The C Matrix can be defined as $J_i - J_j$, which can be obtained in each iteration from the two-stage Jacobin update. Then, the is substituted as the new distance constraint. Following the above two steps, the original equation 14 can be reformulated into a linear form. The term d in equation 20 can be selected based on allowable deformation. Similarly, equation 15 can also be formulated into a linear form by changing the inequality sign. Therefore, the distance between the two points will be bounded to avoid the overstretching. Hence, all non-convex constraints which are difficult to handle, can be expressed in a linear form and traditional solvers such as "lsqlin" function in the Matlab can be used for solving.

2) shape distortion: When performing the folding task, it is preferable to minimize the distortion in the shape of the object. Using the desired points as the benchmark, an error $e_{shape} \in R^3$ can be defined as the difference between the feature point vector and the desired point vector, as shown in figure 3b and the objective are to maintain and minimize this deviation throughout the manipulation process.

$$(d_i - d_j) - (p_i - p_j) = e_{shape} \tag{22}$$

$$\|e_{shape}\| = \varepsilon_{tol} \tag{23}$$

At a new time instance, the joints will be rotated by $\Delta\theta$, and a new error term, e^{t+1}_{shape}, can be obtained, which could have different direction and magnitude as compared to its previous instance, e^{t}_{shape}. Assuming the change in the direction between two successive time frame is small, the error vector can be represented as:

$$\frac{e^{t+1}_{shape}}{\left\|e^{t+1}_{shape}\right\|} = \frac{e^{t}_{shape}}{\left\|e^{t}_{shape}\right\|}, \left\|e^{t+1}_{shape}\right\| = \varepsilon_{tol}\left\|e^{t}_{shape}\right\| \quad (24)$$

Based on equations 18 and 24, eqautinon 23 can be updated as:

$$(p_i + J_i\Delta\theta) - (p_j + J_j\Delta\theta) + \varepsilon_{tol}e^{t}_{shape} = d_i - d_j \quad (25)$$

Rearranging equation 25 to give:

$$(J_i - J_j)\Delta\theta = d_1 - d_2 - p_i + p_j - \varepsilon_{tol}e^{t}_{shape} \quad (26)$$

$$(J_i - J_j)\Delta\theta = (1 - \varepsilon_{tol})e^{t}_{shape} \quad (27)$$

The original non-convex equation 23 is converted into a linear form in equation 27. Following the same procedure in collision and overstretching avoidance, the active-set method can be used to solve this problem.

D. Algorithms Framework and Controller Design

For precise folding the deformable object, $\Delta\theta$ should be computed as the inputs with the consideration of specific task constraints in each iteration. Levenberg–Marquardt algorithm was used to solve for an optimal solution in the objective or loss function, subject to linearized constraints. The whole system could now be solved and the optimization controller is shown as the figure 5, and the formulated below:

$$\begin{aligned} \underset{\Delta\theta\in R^n}{Minimize}\ & \|t - F(\theta) - J(\theta) * \Delta\theta\|^2 \\ s.t.\ & \Delta\theta \in S = \{\theta, \theta_{min} \le \theta \le \theta_{max}\} \\ & \Delta\theta \in K \end{aligned} \quad (28)$$

where S is the set of joint limit and K is the set of a feasible region which is determined by equations 27 and 21. Since the problem was formulated as convex optimization, the solution gradually converges to the global solution and the implementation can be fast real-time manipulation.

III. Experimental and Result

The system was set up using a Windows 64-bit PC platform with Intel 1.8GHz i7 CPU and 16GB RAM. The experiments were implemented using the C++ version of the open-source Bullet physics library [25]. The hardware consists of two 3D articulated robot arm as shown in Figure 4(a). The positions of the two robot arm can be controlled by the θ values from each joint. We use the default feedback sensor embedded in the simulator engine to provide the coordinates of the feature points and the desired points in real-time. The sensor, however, could be replaced with other types, such as infrared sensor (SR300 camera [15]), or magnetic sensor in the real application. Our folding task is to use two articulated arms to fold a deformable thin object with markers on the surface and aligned them as precise as possible. During the folding process, the joint inputs $\Delta\theta$ should be limited or bounded by the constraints.

TABLE I: Summary on the parameters used in the simulation

Property	Damping Coefficient	Dynamic Friction Coefficient	SoftBody Model	Collisions Flags	Positions Solver Iterations
Parameters	0.705	0.5	quad with diagnal	rigid vs soft	6

Property	Dimension	Node Number in X axis	Node Number in Y axis	Mass	Gravity
Parameters	4m * 4m	31	31	4kg	-10N/kg

To validate the effectiveness and robustness of the optimization framework and controller, we examined the proposed algorithm under two different folding scenarios. First, the film was placed on top of a plain table and supported by the table surface. One side (edge) of the film was firmly grasped by two grippers attached to the end-effector of the manipulator, while the rest of the film can be moved freely. This setup mimics one of the most common scenes happened in our daily activities, such folding a cloth or bed sheet. Next, we also test the algorithm in the scenario of folding thin film in the air. In this setup, the manipulators grasped the sheets at two points (corners) and the other two corners were fixed. This is common in the collaborative case when multiple manipulators hold the same object at the same time. In this paper, we mainly focus on the folding manipulation, which is the final step of the operation. In addition to these two scenarios, to validate the robustness of the proposed model-free approach, we also tested our method on asymmetric thin film with asymmetric patterns.

Note that throughout the experiments, only the feature points and the desired points will be fed to the controller and other system and model parameters are not required. The material properties used in the simulation are summarized in Table I. The dimensions of the film is set as (4m * 4m) for better illustration, and it can be set to a smaller or larger number. The origin of the global coordinates are set on the right-hand system for all experiments with the initial configurations of the two arms are shown in figure 4 (a). The base of the two arms cannot be moved. When the arms start to move, all links can be moved freely.

1) Folding on The Table: The results from our first folding experiment are shown the figure 4. In this experiment, the constraints set in the optimization include limiting the end effector to be above the table, in addition to joint limits and overstretching constraints outlined in earlier sections. Note that the position errors or offsets along the Z axis for both end-effectors are relatively large at the beginning. Because the assumption of our experiment is that the thin film is fully spread on the table and our robot arm firmly grabs it at the beginning, as shown in figure 4 (a).

We can observe from the experimental result in figure 6 (c). The matching error can converge to the desired position quickly. This is benefit from the convex optimization controller which can make sure the $\Delta\theta$ update in the right direction. We can notice that the convergence rate on the left side and on the right side are different. This is because the left and right manipulator should maintain the shape of the object. We can also observe that for each feature point, the

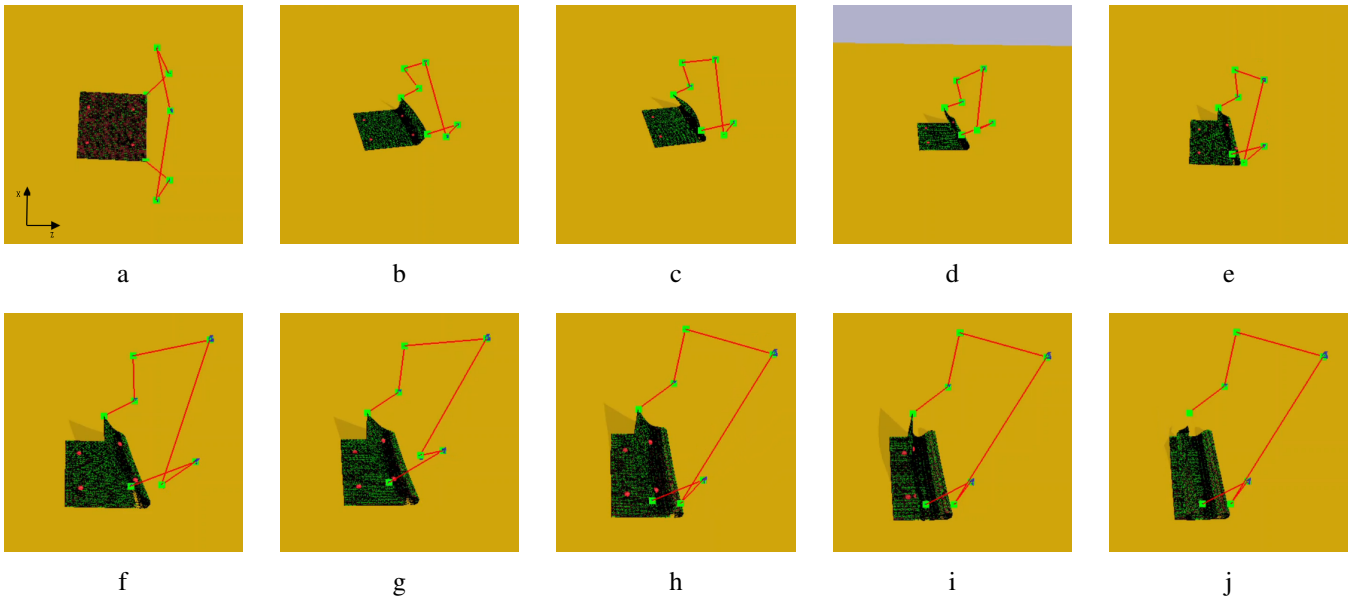


Fig. 4: A series of images showing the process of folding a thin film on a table surface (from (a) to (j), and the coordinate frame is shown in figure (a))

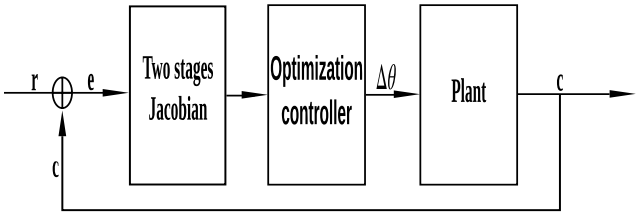
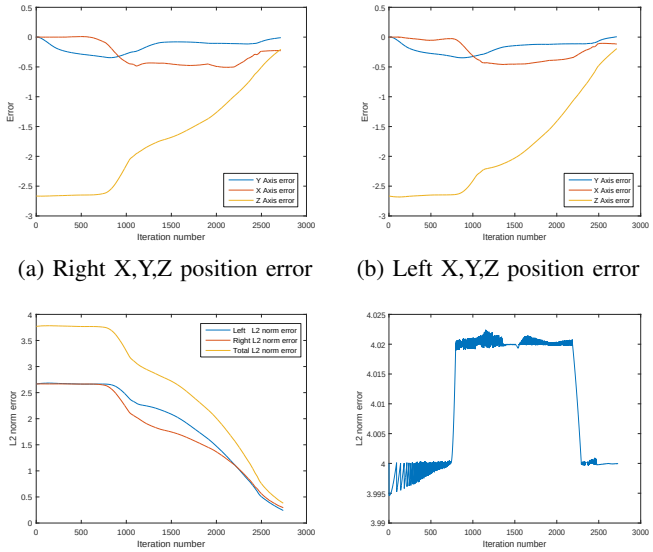


Fig. 5: The controller and the framework of the system



(a) Right X,Y,Z position error (b) Left X,Y,Z position error

(c) L2 norm error (d) The value on the stretching

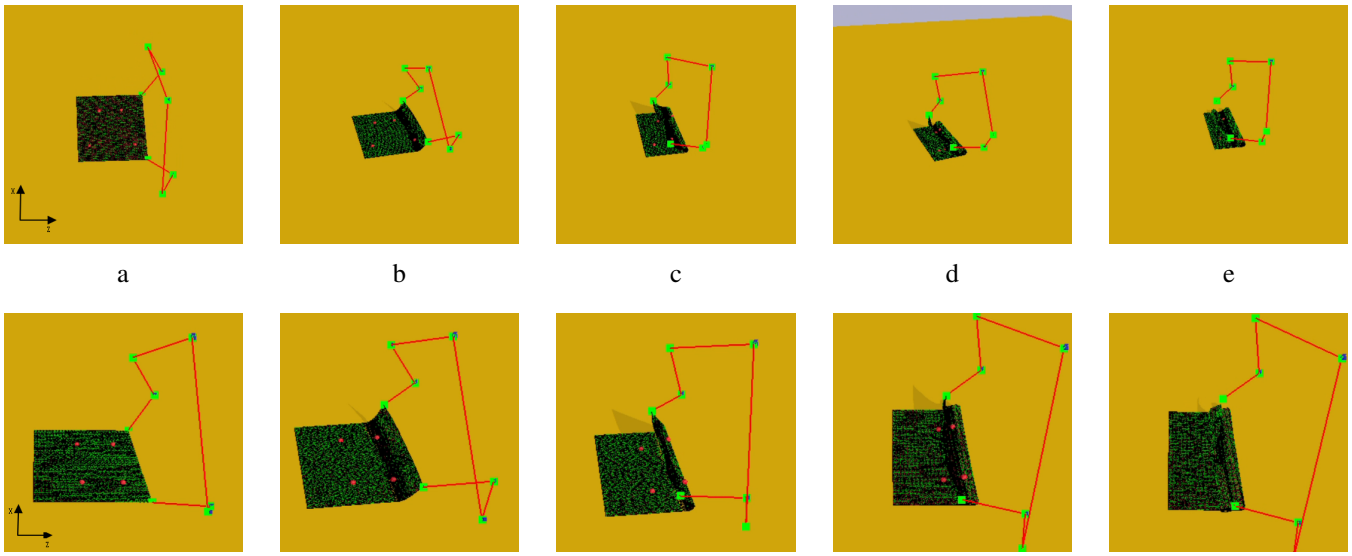
Fig. 6: The performance of different paramters during symmetric object folding

$X\,Y\,Z$ axis errors converge to the zero almost at same time in figure 6 (a) and (b). Slight discrepancy could be due to the shape constraint. Besides, we can also observe self-collision or self-folding occur at some parts of the film, especially near the ending phase. This happens when one end of the film starts to touch the surface of the film underneath, leading to perturbation to the shape of the film.

2) Folding with Asymmetric Markers and Asymmetric Substrate: In our first experiment, the markers are patterned symmetry about the Z axis, and that is also the main direction of folding. Even though our algorithm can handle the symmetry pattern, the markers in the real application can be located asymmetrically. To test our algorithm's performance on asymmetric situation, we labelled the markers asymmetry between the left and right side along the Z axis as shown in figure 7 (a). In this experiment, we also set the distance between the two end-effector to be within $4m$ and $4.02m$ to represent the constraints of overstretching and self-collision. We can notice that even in an asymmetric configuration, our algorithm can successfully fold the thin film as specified with a fast convergence rate.

We have also tested the influence on the position in our folding algorithm. The four markers were shifted to a new position on the surface. The feature ponits as well as the desired points were moved closer to the gripper as compared with the first experiment, as shown in figure 7(f). As pointed out in [17], the rigidity decreases as the distance between the feature points and the gripper increase. The folding task was also successfully completed with the consideration of self-collision and overstretching. We also notice that as the feature points get closer to the gripper, the convergence rate is faster, which is benefit from the rigidity behavior.

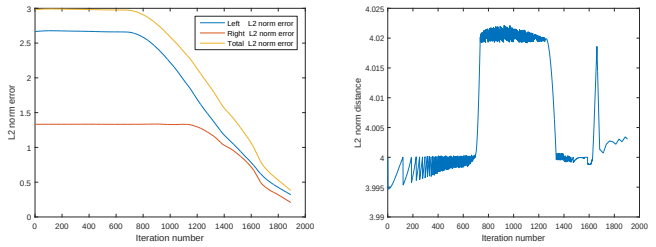
3) Folding in The Air: In the last experiment, we examined folding the film in the air, which is similar to folding a large film by two persons. Since background noises are inevitable in real-life, random noises with a zero mean and a variance ranged from 0.025 to 0.5, were added when measuring the coordinates of the feature points in the



a b c d e

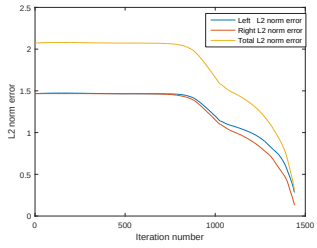
f g h i j

Fig. 7: (a) to (e) Folding film according to asymmetric markers; (f) to (j) folding asymmetric film

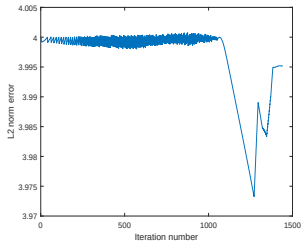


(a) L2 norm error

(b) The value on the stretching



(c) L2 norm error



(d) The value on the stretching

Fig. 8: The performance of different paramters during asymmetric film folding and folding asymmetric film

simulation. From the experimental results shown in figure 10, we can observe that the folding task can converge quickly when the variance is lower than 0.1. However, a slight divergence can be observed when it is larger than 0.2. Also, vibration or fluctuation can also be observed when the variance is large. Hence, the film can be folded in the air with the proposed algorithm, but the quality of the measured data strongly influence the system performance.

In addition, we have also examined folding films with markers located arbitrarily and asymmetrically on the surface. Following the markers, it would lead to asymmetric folding on the resultant film. In this test, comparing with the previous experiment, the desired points were shifted and rotated by an angle while the initial feature points on the film remained the same. Even with possible self-collision in the film during asymmetric folding, the simulation results confirm that the proposed algorithm can guide the manipulator to complete the folding task. As shown in Fig 10(b), the feature points gradually converge to the desired points.

IV. Conclusion and Future Work

In this paper, we proposed a new method to precisely fold a deformable thin film through an optimization framework with only visual information available. To reduce the computation burden and to streamline the process, linearization was first performed to reformulate the control function with complex and nonlinear terms into a linear form. Constraints related to the soft object handling such as overstretching limit and the tendency to minimize shape distortion were considered and the constraints were expressed in a linear form such that the entire problem was converted into a linear, convex optimization problem. A two-stage Jacobian scheme was implemented such as uncertainty from the object and the environment were also taken into account when deriving the optimal joint inputs for the robot manipulators. The performance of the proposed algorithm was examined through Bullet simulator. Different scenarios, including symmetric and asymmetric distributions of the markers on the surface, were all examined. Collaborative film folding was also examined and all experiments were successfully carried out with our proposed algorithm. This algorithm is adaptive to different kinds of robotic manipulators, providing a linearized and fast control scheme for real-time control for automated object folding and other applications.

References

[1] 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*, vol. 37, no. 7, pp. 688–716, 2018.

[2] I. Garcia-Camacho, M. Lippi, M. C. Welle, H. Yin, R. Antonova, A. Varava, J. Borras, C. Torras, A. Marino, G. Alenya *et al.*, "Benchmarking bimanual cloth manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1111–1118, 2020.

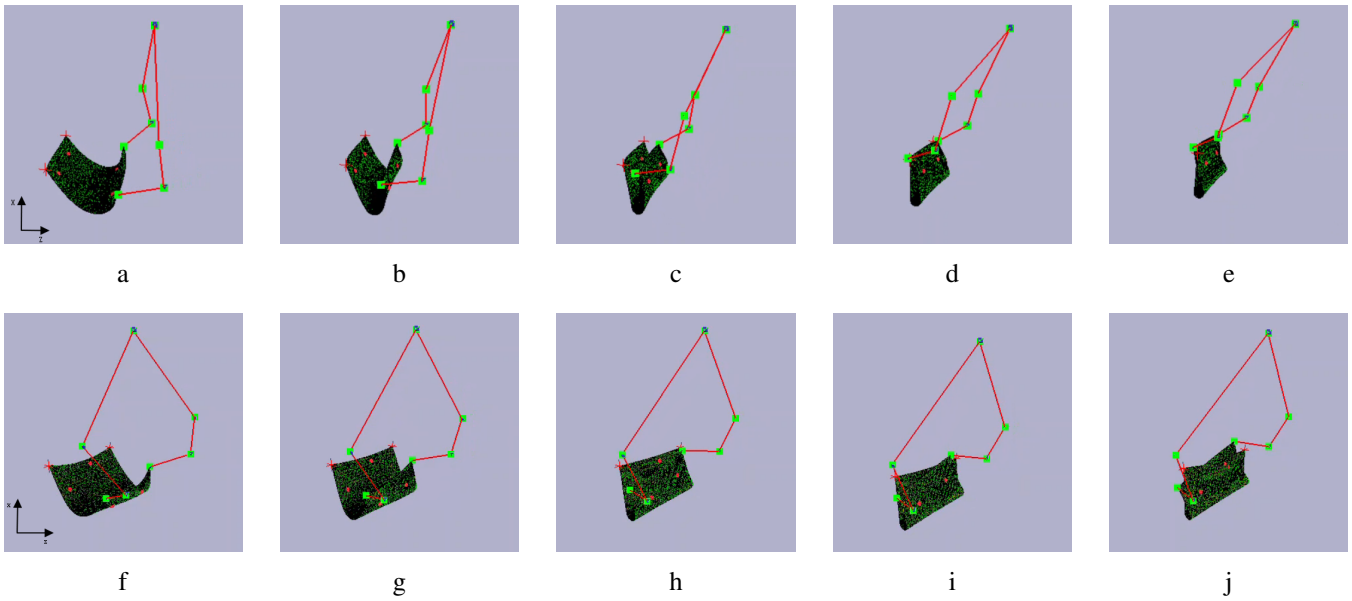
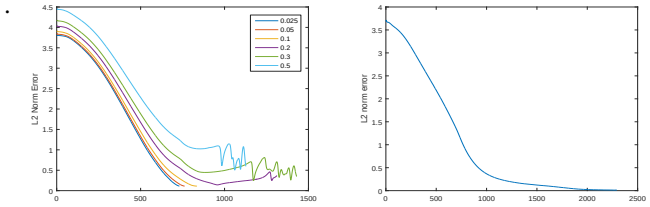


Fig. 9: Collaborative film folding in air



(a) Robust on added noise (b) Asymmetry Pattern

Fig. 10: The influence of the noise on the L2 norm

[3] J. Das and N. Sarkar, "Autonomous shape control of a deformable object by multiple manipulators," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 1, pp. 3–27, 2011.

[4] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen, "Folding deformable objects using predictive simulation and trajectory optimization," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6000–6006.

[5] S. Duenser, J. M. Bern, R. Poranne, and S. Coros, "Interactive robotic manipulation of elastic objects," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3476–3481.

[6] H. Bai and J. T. Wen, "Cooperative load transport: A formation-control perspective," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 742–750, 2010.

[7] P. Long, W. Khalil, and P. Martinet, "Dynamic modeling of cooperative robots holding flexible objects," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 182–187.

[8] F. Ruggiero, A. Petit, D. Serra, A. C. Satici, J. Cacace, A. Donaire, F. Ficuciello, L. R. Buonocore, G. A. Fontanelli, V. Lippiello *et al.*, "Nonprehensile manipulation of deformable objects: Achievements and perspectives from the robotic dynamic manipulation project," *IEEE Robotics & Automation Magazine*, vol. 25, no. 3, pp. 83–92, 2018.

[9] K. Simon and R. Basri, "Elasticity-based matching by minimising the symmetric difference of shapes," *IET Computer Vision*, vol. 12, no. 4, pp. 412–423, 2017.

[10] D. Navarro-Alarcon, H. M. Yip, Z. Wang, Y.-H. Liu, F. Zhong, T. Zhang, and P. Li, "Automatic 3-d manipulation of soft objects by robotic arms with an adaptive deformation model," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 429–441, 2016.

[11] D. Navarro-Alarcon and Y.-H. Liu, "Fourier-based shape servoing: a new feedback method to actively deform soft objects into desired 2-d image contours," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 272–279, 2017.

[12] T. Wada, S. Hirai, S. Kawamura, and N. Kamiji, "Robust manipulation of deformable objects by a simple pid feedback," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 1. IEEE, 2001, pp. 85–90.

[13] S. Hirai and T. Wada, "Indirect simultaneous positioning of deformable objects with multi-pinching fingers based on an uncertain model," *Robotica*, vol. 18, no. 1, pp. 3–11, 2000.

[14] B. Jia, Z. Hu, J. Pan, and D. Manocha, "Manipulating highly deformable materials using a visual feedback dictionary," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 239–246.

[15] Z. Hu, P. Sun, and J. Pan, "Three-dimensional deformable object manipulation using fast online gaussian process regression," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 979–986, 2018.

[16] A. Cherubini, J. Leitner, V. Ortenzi, and P. Corke, "Towards vision-based manipulation of plastic materials," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 485–490.

[17] D. Berenson, "Manipulation of deformable objects without modeling and simulating deformation," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 4525–4532.

[18] H. K. Chu, J. K. Mills, and W. L. Cleghorn, "Image-based visual servoing through micropart reflection for the microassembly process," *Journal of Micromechanics and Microengineering*, vol. 21, no. 6, p. 065016, 2011.

[19] D. Navarro-Alarcon, Y.-H. Liu, J. G. Romero, and P. Li, "Energy shaping methods for asymptotic force regulation of compliant mechanical systems," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 6, pp. 2376–2383, 2014.

[20] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[21] J. Alonso-Mora, R. Knepper, R. Siegwart, and D. Rus, "Local motion planning for collaborative multi-robot manipulation of deformable objects," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 5495–5502.

[22] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[23] M. C. Grant and S. P. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent advances in learning and control*. Springer, 2008, pp. 95–110.

[24] M. Schlueter, S. Erb, M. Gerdts, S. Kemble, and J. Ruckmann, "Midaco on minlp space applications," *Advances in Space Research*, vol. 51, no. 7, pp. 1116–1131, 2013.

[25] E. Coumans *et al.*, "Bullet physics library," *Open source: bulletphysics. org*, vol. 15, no. 49, p. 5, 2013.