Adaptive Shape Servoing of Elastic Rods using Parameterized Regression Features and Auto-Tuning Motion Controls

Jiaming Qi†, Guangtao Ran†, *Member, IEEE*, Bohui Wang, *Senior Member, IEEE*, Jian Liu, *Member, IEEE*, and David Navarro-Alarcon, *Senior Member, IEEE*

Abstract—The robotic manipulation of deformable linear objects has shown great potential in a wide range of real-world applications. However, it presents many challenges due to the objects' complex nonlinearity and high-dimensional configuration. In this paper, we propose a new shape servoing framework to automatically manipulate elastic rods through visual feedback. Our new method uses parameterized regression features to compute a compact (low-dimensional) feature vector that quantifies the object's shape, thus, enabling to establish an explicit shape servo-loop. To automatically deform the rod into a desired shape, the proposed adaptive controller iteratively estimates the differential transformation between the robot's motion and the relative shape changes; This valuable capability allows to effectively manipulate objects with unknown mechanical models. An auto-tuning algorithm is introduced to adjust the robot's shaping motions in real-time based on optimal performance criteria. To validate the proposed framework, a detailed experimental study with vision-guided robotic manipulators is presented.

Index Terms—Deformable objects, robotic manipulation, visual servoing, sensorimotor models, adaptive control.

I. Introduction

THE manipulation of deformable objects is currently an open (and hot!) research problem in robotics [1] that has attracted many researchers due to its applicability in many scenarios, e.g. positioning fabrics [2], shaping food materials [3], assembling compliant components [4], manipulating deformable linear objects (DLO) [5], etc. The feedback control of the object's non-rigid configuration is referred in the literature as *shape servoing* [6], a frontier problem that presents three main challenges: (i) The efficient (viz. compact) characterization of its deformable shape (which is typically represented with high-dimensional visual feedback vectors); (ii) The identification of its motion-deformation model (which depends on the, generally unknown, mechanical properties of the object); (iii) The adaptive modulation of the robot's motions during the active shaping task (as soft objects might be delicate and thus easy to damage).

†These authors contributed equality to this work. This work is supported by the Research Grants Council (RGC) under grant number 14203917.

Jiaming Qi and Guangtao Ran are with Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China. (e-mail: 18B904030@hit.edu.cn; ranguangtao@hit.edu.cn).

Bohui Wang is with the School of Cyber Science and Engineering, Xi'an Jiaotong University, Xi'an 710049, China. (e-mail: wang31aa@126.com).

Jian Liu is with the School of Automation, Southeast University, Nanjing 210096, China. (e-mail: bkliujian@seu.edu.cn).

David Navarro-Alarcon is with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong. (e-mail: dna@ieee.org).

For shape servoing tasks, feature extraction is necessary to accurately describe the object with a small number of feedback coordinates. Traditional features to represent deformations include: centroids, distance, angles, curvature, etc [7]. However, these geometrically constructed features are local, thus, they cannot describe the overall shape of an object. The development of global features presents an advantage in this problem. Image moments were used in [8] to characterize object's contours, yet, with a limited real-time performance due to its complicated calculations. In [9], principal component analysis was used to project raw Fast Point Feature Histograms into a new space with higher variance and a lower dimension. A catenary-based feature descriptor was developed for tethered mobile robots in [10], [11], however, it is only applicable to specific shapes. A Fourier-based approach was developed in [12] to characterize contours with low-dimensional features. Bézier curves and Non-Uniform Rational Basis Splines (NURBS) are an interesting option to describe complex contours, however, they have not been sufficiently studied in the literature as a way to establish an explicit shape servo-loop. Other types of representations rely on machine learning, e.g. [13], however, these methods require very large data sets to generalize to different situations (which is difficult to guarantee in many applications). The design of a computationally efficient feature extraction algorithm is a key problem in shape servoing.

To execute shape servoing tasks, a controller requires a model that captures the relationship between robot's motions and the produced object's deformations. In [7], [14], a least-squares regression method was used to calculate the deformation Jacobian matrix; Yet, it requires prior knowledge of the object model to properly define a regression structure. There are other approaches that do not require prior knowledge of the structure, e.g., in [15] a Broyden-like method was used to online estimate this deformation matrix. Although these type of methods are easy to implement, they are prone to enter local minima. Kalman Filter (KF) is well-known to estimate unknown variables using a series of measurements in the presence of noise and uncertainty. Based on this idea, the method in [16] constructed a state-space model with the elements of Jacobian matrix and used KF to adaptively estimate its unknown values. Since the manipulation of deformable objects involves nonlinear models with considerable uncertainties, KF is a good option to robustly estimate these unknown terms.

Deformable objects exhibit more complex behaviours than rigid objects, therefore, it is advantageous to control its transient behavior. This can be achieved through performance
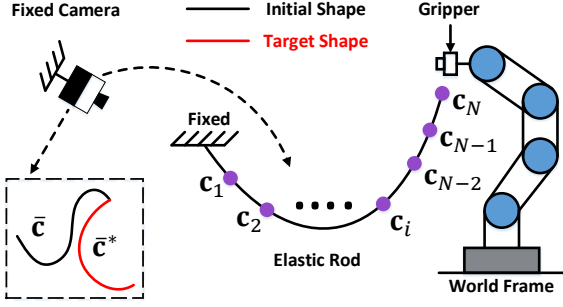
Fig. 1. Representation of active manipulation of the elastic rod, where a vision sensor continuously measures the feature vector **s**, which should be accurately deformed into target feature **s*** within the controller.

adjustment methods, which have been used in motor speed regulation problems [17] to achieve accurate tracking while considering various performance criteria (e.g., overshoot, rising, settling time) [18]. This idea is also useful in the manipulation of soft objects to achieve various dynamic requirements (e.g., soft objects may support rapid deformations, while stiffer materials may need to be slowly manipulated). Performance adjustment can also help to avoid damage, e.g., due to over-stretching and over-compression of an object [19]. However, most of existing shape servoing methods adopt classical constant gain controls [6], which limit the types of dynamic responses they can achieve.

This paper presents a new vision-based framework to automatically deform elastic rods into desired shapes. To quantify the configuration of these DLOs, we propose a general representation method based on parametric regression features, which enables to characterize shapes with a compact feedback-like vector. To the best of the authors' knowledge, this is the first time that a shape controller uses Bézier/NURBS features to establish an *explicit* shape servo-loop (both features were used in [20], but only to project a contour into an image plane, which largely differs from our approach). Our proposed method presents an adaptive control strategy that: (i) iteratively estimates the unknown deformation model (based on an unscented Kalman filter approach), and (ii) online adjusts the controller's parameters (which modulate the transient response). The stability of the system is proved using the Lyapunov theory. To validate the proposed framework, we report an experimental study with robotic manipulators.

II. PRELIMINARIES

Notation. Column vectors are denoted with bold small letters **m** and matrices with bold capital letters **M**. Time evolving variables are represented as $\mathbf{x}_k$, where the subscript k denotes the discrete time instant. $\mathbf{I}_n$ is an $n \times n$ identity matrix. $\otimes$ represents the Kronecker product.

To derive our method, some conditions are now introduced:

- The 2D image feedback centerline of the rod is measured with a static camera in an eye-to-hand configuration [21] (depicted in Fig. 1), and is denoted by:

$$\bar{\mathbf{c}} = [\mathbf{c}_1^\mathsf{T}, \dots, \mathbf{c}_N^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{2N}, \ \mathbf{c}_i = [c_{ui}, c_{vi}]^\mathsf{T} \in \mathbb{R}^2 \quad (1)$$

where N is the number of the centerline points, c_{ui} and c_{vi} are the pixel coordinates of the ith image point.

- During the manipulation task, the rod is rigidly grasped by the robot, and remains all the time within the observable range of the camera with no occlusions.
- The robot is commanded with classical kinematic controls $\Delta \mathbf{r}_k \in \mathbb{R}^q$ [22] that render stiff behaviours and satisfy the incremental position $\mathbf{r}_k = \mathbf{r}_{k-1} + \Delta \mathbf{r}_k$, with bounded commands $\|\Delta \mathbf{r}_k\| \leq \epsilon_r$, for $\epsilon_r > 0$.
- The rod is manipulated slowly such that its shape is determined by the equilibrium of its potential/elastic energy terms only.

Problem Statement. Given a desired 2D constant centerline $\bar{\mathbf{c}}^*$, design a vision-based kinematic controller $\Delta \mathbf{r}_k$ to automatically deform an elastic rod such that its feedback shape $\bar{\mathbf{c}}$ approximates $\bar{\mathbf{c}}^*$, without identifying the mathematical model of the object or the camera's parameters.

III. METHODS

A. Feedback Shape Parameters

The *naïve approach* to shape servoing is to synthesise a regulator that attempts to drive the full coordinates of $\bar{\mathbf{c}}$ into $\bar{\mathbf{c}}^*$. The main problem within this approach is that $\bar{\mathbf{c}}$ is not efficient for real-time control as its dimension is very large. Therefore, it is necessary to compute a reduced-dimension feature $\mathbf{s} \in \mathbb{R}^p (p \ll 2N)$, that can be used for feedback control. Our proposed idea is to fit the 2D feedback centerline to a continuous curve $\mathbf{f}(\rho) \in \mathbb{R}^2$, for ρ as a parametric variable representing the curve's normalized arc-length $0 \leq \rho \leq 1$. Then, a point along the curve can be expressed as $\mathbf{c}_i = \mathbf{f}(\rho_i)$, with ρ_i as the arc-length between the start point $\mathbf{c}_1$ and point $\mathbf{c}_i$ (see Fig. 1), for $\rho_1 = 0, \rho_N = 1$. The proposed parametric curve fitting is modelled as follows:

$$\mathbf{f}(\rho) = \sum_{j=0}^{n} \mathbf{p}_j B_{j,n}(\rho) \quad (2)$$

where the positive scalar $n \in \mathbb{N}$ is the fitting order, and the vector $\mathbf{p}_j \in \mathbb{R}^2$ represents the shape parameters of $\bar{\mathbf{c}}$. The function $B_{j,n}(\rho)$ models the chosen regression parameterization, which can take the following different forms:

1) Polynomial Parameterization. It can easily represent smooth regular shapes, however, if n is selected too large, curve overfitting may occur. The regression function is as follows:

$$B_{j,n}(\rho) = \rho^j \quad (3)$$

2) Bézier Parameterization. As shown in Fig. 2, an n-degree Bézier curve is defined with a polynomial expression using $n + 1$ control points as follows:

$$B_{j,n}(\rho) = C_n^j (1 - \rho)^{n-j} \rho^j \quad (4)$$

where C_n^j denotes the binomial coefficients. Here, $\mathbf{p}_j$ represents "Bézier control points", which are used to obtain a smooth fitting. When many control points are used, the degree of the curve and its computational cost are increased.

3) NURBS Parameterization [23]. It has local shape description properties, thus, the number of control points is independent from the degree of curves. This model can describe
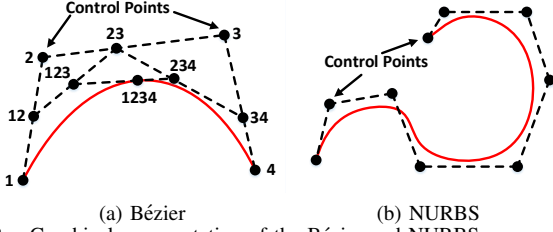
Fig. 2. Graphical representation of the Bézier and NURBS curves.

complex curves more accurately than polynomial/Bézier; Its regression function is as follows:

$$B_{j,n}(\rho) = \frac{N_{j,m}(\rho)\,\omega_j}{\sum_{l=0}^{n} N_{l,m}(\rho)\,\omega_l} \tag{5}$$

where $N_{j,m}(\rho)$ is the Bézier parameterization (4), and ω_j are scalar weights. Along this work, we set $m = n$ for simplicity.

4) Sinusoidal parameterization [24]. It is described as:

$$B_{j,n}(\rho) = \begin{cases} 1, & j = 0 \\ \cos(\frac{j+1}{2}\rho), & j > 0,\ j \text{ is odd} \\ \sin(\frac{j}{2}\rho), & j > 0,\ j \text{ is even} \end{cases} \tag{6}$$

When j is even, (6) denotes the well-known Fourier-based parameterization [12].

By using the curve fitting model (2) (with any function in (3)–(6)), we can linearly parameterize the object's centerline as $\bar{\mathbf{c}} = \mathbf{Bs}$, for a "tall" regression matrix $\mathbf{B}$ satisfying:

$$\mathbf{B} = [\mathbf{B}_1^\intercal, \ldots, \mathbf{B}_N^\intercal]^\intercal \in \mathbb{R}^{2N \times 2(n+1)}$$
$$\mathbf{B}_i = [B_{0,n}(\rho_i), \ldots, B_{n,n}(\rho_i)] \otimes \mathbf{I}_2 \in \mathbb{R}^{2 \times 2(n+1)} \tag{7}$$

and $\mathbf{s} = [\mathbf{p}_0^\intercal, \ldots, \mathbf{p}_n^\intercal]^\intercal \in \mathbb{R}^{2(n+1)}$ as a vector of features that characterizes the shape. This feature vector can be computed from sensor feedback at every iteration as follows:

$$\mathbf{s} = \boldsymbol{\mathcal{B}} \cdot \bar{\mathbf{c}}, \quad \text{for } \boldsymbol{\mathcal{B}} = (\mathbf{B}^\intercal \mathbf{B})^{-1} \mathbf{B}^\intercal \tag{8}$$

To invert $\mathbf{B}^\intercal \mathbf{B}$, a sufficient number N of data points must be used such that $2N > 2(n+1)$. Fig. 3 shows a schematic diagram of the overall manipulation strategy.

Remark 1. Although only four regression functions are given, there are other curve descriptors (e.g., B-spline, rational approximation, etc) that can be expressed as $\bar{\mathbf{c}} = \mathbf{Bs}$.

B. UKF-Based Approximation of the Deformation Model

Since we consider regular (i.e. mechanically well-behaved) elastic objects, it is reasonable to assume that small robot motions $\Delta\mathbf{r}_k$ will produce small changes $\Delta\bar{\mathbf{c}}_k$. We locally model this situation (around the current operating point) with the expression, $\Delta\bar{\mathbf{c}}_k = \boldsymbol{\mathcal{D}}_k \cdot \Delta\mathbf{r}_k$ where $\boldsymbol{\mathcal{D}}_k$ is introduced to model the local deformation properties of the elastic object undergoing quasi-static manipulation by the robot. By using this relation, we obtain the following motion model:

$$\Delta\mathbf{s}_k = \boldsymbol{\mathcal{B}}\boldsymbol{\mathcal{D}}_k \cdot \Delta\mathbf{r}_k = \mathbf{J}_k \cdot \Delta\mathbf{r}_k \tag{9}$$

where $\Delta\mathbf{s}_k = \mathbf{s}_k - \mathbf{s}_{k-1} \in \mathbb{R}^p$ denotes the features' changes, and $\mathbf{J}_k = \boldsymbol{\mathcal{B}}\boldsymbol{\mathcal{D}}_k \in \mathbb{R}^{p \times q}$ represents a Jacobian-like matrix that transforms robot motions into feature changes; This structure is difficult to analytically compute (i.e., with $\partial\mathbf{s}/\partial\mathbf{r}$), as the deformation properties of the DLO are generally *unknown*. Instead of identifying the full mechanical model, in this paper

we design an algorithm that computes local approximations of $\mathbf{J}_k$ in real-time. To this end, let us define the augmented state $\mathbf{x}_k = [\partial s_1/\partial\mathbf{r}, \ldots, \partial s_p/\partial\mathbf{r}]^\intercal \in \mathbb{R}^{pq}$, where $\partial s_i/\partial\mathbf{r} \in \mathbb{R}^{1 \times q}$ denotes the ith row of $\mathbf{J}_k$. The discrete system (9) can then be transformed into the state-space model:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \boldsymbol{\eta}_k, \quad \mathbf{y}_k = \mathbf{M}_k\mathbf{x}_k + \boldsymbol{\nu}_k \tag{10}$$

where the system output is defined as $\mathbf{y}_k = \Delta\mathbf{s}_k$, and which assumes the process and measurement noises to be zero-mean Gaussian white noise, i.e., $\boldsymbol{\eta}_k \sim N(\mathbf{0}, \mathbf{U}_k)$ and $\boldsymbol{\nu}_k \sim N(\mathbf{0}, \mathbf{W}_k)$. The measurement matrix $\mathbf{M}_k$ is defined as:

$$\mathbf{M}_k = \text{diag}(\Delta\mathbf{r}_k^\intercal, \cdots, \Delta\mathbf{r}_k^\intercal) \in \mathbb{R}^{p \times pq} \tag{11}$$

We use a modified UKF [25] (which has good dynamic performance and robustness to external disturbances [26]) to compute local approximations of $\mathbf{x}_k$ in real-time. UKF adopts the unscented transform (UT) to propagate mean and covariance through a nonlinear transformation among a minimal set of sample points (i.e., sigma points) [27]. Then, the weighted sum method is used for the transformed samples to obtain the posterior mean and covariance of state vectors with a second-order accuracy. The procedure can be summarized as follows:

Step 1: The variable $\mathbf{x}_{k-1}$ with mean $\hat{\mathbf{x}}_{k-1}$ and covariance $\hat{\mathbf{P}}_{k-1}$ is estimated by sigma points defined by

$$\chi_{k-1}^0 = \hat{\mathbf{x}}_{k-1}, \quad \chi_{k-1}^i = \hat{\mathbf{x}}_{k-1} + (a\sqrt{pq\hat{\mathbf{P}}_{k-1}})_i$$
$$\chi_{k-1}^{i+pq} = \hat{\mathbf{x}}_{k-1} - (a\sqrt{pq\hat{\mathbf{P}}_{k-1}})_i, \quad i = 1, \ldots, pq \tag{12}$$

where $(\bullet)_i$ is the ith column of the cholesky matrix decomposition. $a > 0$ is a scaling parameter specifying the distribution of the sigma points around $\hat{\mathbf{x}}_{k-1}$.

Step 2: Prediction. Sigma points pass through the process model to generate transformed points:

$$\chi_{k|k-1}^i = \chi_{k-1}^i, \ i = 0, 1, \ldots, 2pq,$$
$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2pq} \varpi_i \chi_{k|k-1}^i \tag{13}$$
$$\hat{\mathbf{P}}_{k|k-1} = \sum_{i=0}^{2pq} \varpi_i (\tilde{\chi}_{k|k-1}^i)(\tilde{\chi}_{k|k-1}^i)^\intercal + (\mathbf{U}_k + \Delta\mathbf{U}_k)$$

where $\Delta\mathbf{U}_k > 0$ is a positive-definite matrix to improve the approximation stability [25]. $\varpi_0 = 1 - \frac{1}{a^2}, \varpi_i = \frac{1}{2pqa^2}, i = 1, \ldots, 2pq$, are scalar weights and $\sum_{i=0}^{2pq} \varpi_i = 1$. $\tilde{\chi}_{k|k-1}^i = \chi_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}$.

Step 3: Update. Classical KF can be used to update the measurement signal as follows:

$$\hat{\mathbf{y}}_k = \mathbf{M}_k\hat{\mathbf{x}}_{k|k-1}$$
$$\hat{\mathbf{P}}_{yy,k} = \mathbf{M}_k\hat{\mathbf{P}}_{k|k-1}\mathbf{M}_k^\intercal + \mathbf{W}_k$$
$$\hat{\mathbf{P}}_{xy,k} = \hat{\mathbf{P}}_{k|k-1}\mathbf{M}_k^\intercal \tag{14}$$
$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$$
$$\hat{\mathbf{P}}_k = \hat{\mathbf{P}}_{k|k-1} - \mathbf{K}_k\hat{\mathbf{P}}_{yy,k}\mathbf{K}_k^\intercal$$

where $\tilde{\mathbf{y}}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k$, and $\mathbf{K}_k = \hat{\mathbf{P}}_{xy,k}\hat{\mathbf{P}}_{yy,k}^{-1}$ is Kalman gain.

Step 4: Repeat steps 1–3 for the next update.

After completing an iteration, we de-vectorize $\hat{\mathbf{x}}_k$ to get an estimation $\hat{\mathbf{J}}_k$ of the unknown Jacobian matrix.
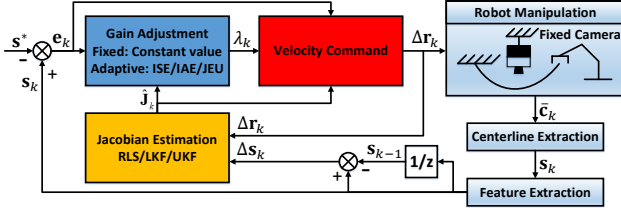
Fig. 3. The block diagram showing the overall control workflow.

Remark 2. In practice, the robot's motion $\Delta \mathbf{r}_k$ is bounded, which implies that the elements of $\mathbf{M}_k$ are also bounded. Also, note that the occlusion-free observations of the deformable object (with intrinsic regularity) imply that the noise matrices $\mathbf{U}_k$ and $\mathbf{W}_k$ are similarly bounded. These conditions enable us to apply Theorem 1 from [25] (we refer the reader to the original source), which ensures that (after the UKF's algorithm has been sufficiently iterated) the estimation error $\tilde{\mathbf{J}}_k = \mathbf{J}_k - \widehat{\mathbf{J}}_k$ is bounded by a small positive scalar $\|\tilde{\mathbf{J}}_k\| \leq \epsilon_J$. This initialization of UKF can be easily achieved by performing small babbling motions at the starting configuration of the system, i.e., before the shape control experiment, see [12].

Remark 3. Throughout this paper, we assume that $\mathbf{J}_k$ and its estimation $\widehat{\mathbf{J}}_k$ are both full column rank. This condition physically implies that the manipulated object is away from ill-conditioned configurations, e.g., around a fully-stretched shape or when the object suddenly breaks.

C. Adaptive Shape Servoing Controller

In this section, we design the motion command $\Delta \mathbf{r}_k$ to minimize the shape error $\mathbf{e}_k = \mathbf{s}_k - \mathbf{s}^*$ between the feedback feature $\mathbf{s}_k$ and a constant target $\mathbf{s}^*$. To this end, let us introduce the following quadratic performance index:

$$Q = \mathbf{e}_k^\mathsf{T} \mathbf{e}_k + \lambda \Delta \mathbf{r}_k^\mathsf{T} \Delta \mathbf{r}_k \tag{15}$$

where $\lambda > 0$ is a weight that can be used to specify the magnitude of the motion input $\Delta \mathbf{r}_k$. From (9), we can obtain the following differential error model:

$$\mathbf{e}_k - \mathbf{e}_{k-1} = \mathbf{J}_k \Delta \mathbf{r}_k, \quad \mathbf{e}_k + \mathbf{e}_{k-1} = 2\mathbf{e}_{k-1} + \mathbf{J}_k \Delta \mathbf{r}_k \tag{16}$$

which we use for deriving the control input. To this end, let us compute the extremum $\partial Q / \partial \Delta \mathbf{r}_k = \mathbf{0}$, replace $\mathbf{J}_k$ with $\widehat{\mathbf{J}}_k$, and solve the expression for $\Delta \mathbf{r}_k$:

$$\Delta \mathbf{r}_k = -\mathbf{\Phi}^{-1} \widehat{\mathbf{J}}_k^\mathsf{T} \mathbf{e}_{k-1}, \quad \mathbf{\Phi} = \lambda \mathbf{I}_q + \widehat{\mathbf{J}}_k^\mathsf{T} \widehat{\mathbf{J}}_k \tag{17}$$

The performance of this controller can be regulated by adjusting the values of the positive gain λ. This valuable property is beneficial when manipulating soft/delicate objects during the shaping motions. For example, for $\lambda \approx 0$, the control input takes the form $\Delta \mathbf{r}_k \approx -\widehat{\mathbf{J}}_k^+ \mathbf{e}_{k-1}$ with the largest magnitude, where $\widehat{\mathbf{J}}_k^+$ denotes the standard pseudo-inverse matrix. While $\lambda \to \infty$, $\Delta \mathbf{r}_k \approx \mathbf{0}$ makes the system converge slowly. Intuitively speaking, manually setting λ can flexibly make the system adapt to various working conditions. However, simply increasing λ may cause the slow motion, while decreasing λ may cause overshoot (due to the excessive error). Therefore, λ should be adjusted automatically to always be optimal to achieve the ideal dynamic response. For this issue, a parameter optimization algorithm based on the gradient-descent method to adjust λ is given as follows:

i) ISE (integral squared error): It is a well-known criteria for obtaining optimal controller parameters [28]. It penalizes large errors with the metric:

$$H = \frac{1}{2} \sum_{k=1}^{\infty} \mathbf{e}_k^\mathsf{T} \mathbf{e}_k \tag{18}$$

which is used for computing the parameter tuning criterion:

$$\nabla_\lambda H = \sum_{k=1}^{\infty} \mathbf{e}_{k-1}^\mathsf{T} \widehat{\mathbf{J}}_k \left(\mathbf{I}_q - \mathbf{\Phi}^{-1} \widehat{\mathbf{J}}_k^\mathsf{T} \widehat{\mathbf{J}}_k \right) \mathbf{\Phi}^{-2} \widehat{\mathbf{J}}_k^\mathsf{T} \mathbf{e}_{k-1} \tag{19}$$

ii) IAE (integral absolute error): It defines a metric based on the *magnitude* of the error

$$H = \sum_{k=1}^{\infty} \|\mathbf{e}_k\| \tag{20}$$

which makes it sensitive to small errors. The parameter tuning criterion is defined as follows:

$$\nabla_\lambda H = \sum_{k=1}^{\infty} \frac{\mathbf{e}_{k-1}^\mathsf{T} \widehat{\mathbf{J}}_k \left(\mathbf{I}_q - \mathbf{\Phi}^{-1} \widehat{\mathbf{J}}_k^\mathsf{T} \widehat{\mathbf{J}}_k \right) \mathbf{\Phi}^{-2} \widehat{\mathbf{J}}_k^\mathsf{T} \mathbf{e}_{k-1}}{\|\mathbf{e}_k\|} \tag{21}$$

iii) JEU (integral of the squared error and control) ISE and IAE only consider error-based performance constraints, thus, cannot adjust other system requirements, e.g., minimum overshoot, peak time, rising time. The JEU criterion contains a weighted sum of both squared error and control terms [29]:

$$H = \frac{1}{2} \sum_{k=1}^{\infty} \left(\omega_1 \mathbf{e}_k^\mathsf{T} \mathbf{e}_k + \omega_2 \Delta \mathbf{r}_k^\mathsf{T} \Delta \mathbf{r}_k \right) \tag{22}$$

for ω_1 and ω_2 as positive scalars that satisfy $\omega_1 + \omega_2 = 1$. The JEU-based parameter tuning criterion is:

$$\nabla_\lambda H = \sum_{k=1}^{\infty} \left(\omega_1 \mathbf{e}_k^\mathsf{T} \widehat{\mathbf{J}}_k + \omega_2 \Delta \mathbf{r}_k^\mathsf{T} \right) \mathbf{\Phi}^{-2} \widehat{\mathbf{J}}_k^\mathsf{T} \mathbf{e}_{k-1} \tag{23}$$

Remark 4. Note that for the above three criteria, we replace the unknown deformation Jacobian matrix $\mathbf{J}_k$ by its numerical approximation $\widehat{\mathbf{J}}_k$.

In our proposed method, the control parameter λ is dynamically updated with the gradient-descent rule as follows:

$$\lambda_k = \lambda_{k-1} - d \cdot \nabla_\lambda H, \quad \text{for } \lambda_k \geq \epsilon_\lambda \tag{24}$$

where d is a positive gain to control the update rate of λ_k, and $\epsilon_\lambda > 0$ is a small scalar. The initial value λ_0 is generally set to a large positive value to prevent control saturation (arising e.g., due to a large initial error $\mathbf{e}_k$). These tuning criteria are only implemented when $\lambda_k \geq \epsilon_\lambda$ to prevent it becoming smaller than ϵ_λ. This adaptive update rule is used to compute λ in the control law (17).

Proposition 1. *Consider the dynamic system* (9) *in closed-loop with the controller* (17)*, with model estimator* (12)–(14) *and gain adjustment* (24)*. For this system, the deformation error* $\mathbf{e}_k$ *converges to a compact set around zero, which is uniformly bounded.*

Proof: Consider the discrete Lyapunov function defined by $V_k = \frac{1}{2}\mathbf{e}_k^\intercal\mathbf{e}_k$, whose finite difference is [30]:

$$\Delta V_k = V_k - V_{k-1} = \frac{1}{2}\mathbf{e}_k^\intercal\mathbf{e}_k - \frac{1}{2}\mathbf{e}_{k-1}^\intercal\mathbf{e}_{k-1} \qquad (25)$$

By substituting (16) and (17) into (25), we obtain:

$$\begin{aligned}
\Delta V_k &= (\mathbf{e}_{k-1} + \tfrac{1}{2}\mathbf{J}_k\Delta\mathbf{r}_k)^\intercal\mathbf{J}_k\Delta\mathbf{r}_k \\
&= \mathbf{e}_{k-1}^\intercal\mathbf{J}_k\Delta\mathbf{r}_k + \frac{1}{2}\Delta\mathbf{r}_k^\intercal\mathbf{J}_k^\intercal\mathbf{J}_k\Delta\mathbf{r}_k \qquad (26) \\
&= \mathbf{e}_{k-1}^\intercal\widehat{\mathbf{J}}_k\Delta\mathbf{r}_k + \mathbf{e}_{k-1}^\intercal\tilde{\mathbf{J}}_k\Delta\mathbf{r}_k + \frac{1}{2}\Delta\mathbf{r}_k^\intercal\mathbf{J}_k^\intercal\mathbf{J}_k\Delta\mathbf{r}_k \\
&= -\mathbf{e}_{k-1}^\intercal\mathbf{L}_k\mathbf{e}_{k-1} + \mathbf{e}_{k-1}^\intercal\tilde{\mathbf{J}}_k\Delta\mathbf{r}_k + \frac{1}{2}\Delta\mathbf{r}_k^\intercal\mathbf{J}_k^\intercal\mathbf{J}_k\Delta\mathbf{r}_k
\end{aligned}$$

for $\mathbf{L}_k = \widehat{\mathbf{J}}_k\mathbf{\Phi}^{-1}\widehat{\mathbf{J}}_k^\intercal \geq 0$ as a positive semi-definite symmetric matrix. With the aim to establish the convergence set Ω where $\Delta V_k \leq 0$, let us analyse the following relation:

$$-\mathbf{e}_{k-1}^\intercal\mathbf{L}_k\mathbf{e}_{k-1} + \mathbf{e}_{k-1}^\intercal\tilde{\mathbf{J}}_k\Delta\mathbf{r}_k + \frac{1}{2}\Delta\mathbf{r}_k^\intercal\mathbf{J}_k^\intercal\mathbf{J}_k\Delta\mathbf{r}_k \leq 0 \quad (27)$$

Considering the Young's inequality and the upper bounds of $\tilde{\mathbf{J}}_k$ and $\Delta\mathbf{r}_k$ (see Sec. II and Remark 2), we can obtain:

$$-\mathbf{e}_{k-1}^\intercal\tilde{\mathbf{J}}_k\Delta\mathbf{r}_k \leq \frac{1}{2}\|\mathbf{e}_{k-1}\|^2 + \frac{1}{2}\epsilon_J^2\epsilon_r^2 \qquad (28)$$

Substituting (28) into (27), and after some algebraic and bounds operations, the following relations can be derived:

$$\Delta\mathbf{r}_k^\intercal\mathbf{J}_k^\intercal\mathbf{J}_k\Delta\mathbf{r}_k \leq \left(2\bar{\beta}(\mathbf{L}_k)+1\right)\|\mathbf{e}_{k-1}\|^2 + \epsilon_J^2\epsilon_r^2 \quad (29)$$

$$\epsilon_r^2\bar{\beta}\left(\mathbf{J}_k^\intercal\mathbf{J}_k\right) - \epsilon_J^2\epsilon_r^2 \leq \left(2\bar{\beta}(\mathbf{L}_k)+1\right)\|\mathbf{e}_{k-1}\|^2 \qquad (30)$$

where $\bar{\beta}(\bullet)$ is the maximum eigenvalue of $\bullet$. Finally, the convergence set Ω of $\mathbf{e}_k$ is calculated as follows:

$$\Omega = \left\{\mathbf{e}_k \ \Big|\ 2V_k = \|\mathbf{e}_k\|^2 \leq \frac{\epsilon_r^2\left(\bar{\beta}\left(\mathbf{J}_k^\intercal\mathbf{J}_k\right) - \epsilon_J^2\right)}{2\bar{\beta}\left(\mathbf{L}_k\right)+1}\right\} \quad (31)$$

It is assumed that UKF has been sufficiently iterated to provide a small enough ϵ_J that ensures that $\bar{\beta}(\mathbf{J}_k^\intercal\mathbf{J}_k) - \epsilon_J^2 > 0$ (see Remark 2). From (31), we can see that $\mathbf{e}_k$ is uniformly bounded [31] with the convergence set Ω. As λ_k remains positive semi-definite, $\mathbf{L}_k$ will always remain positive semi-definite, this ensures that $\bar{\beta}(\mathbf{L}_k)$ is always positive. Thus, the stability of the system is not affected (considering that both the numerator and denominator are greater than zero), i.e., $\|\mathbf{e}_k\|$ will converge into Ω. ∎

Remark 5. For ISE (19), λ_k decreases (as $\nabla_\lambda H \geq 0$) to make the system converge faster with enlarging $\Delta\mathbf{r}_k$. For IAE (21), λ_k is also reduced due to positive $\nabla_\lambda H$. When $\|\mathbf{e}_k\|$ is large, IAE is slower than ISE, but when $\|\mathbf{e}_k\|$ is small (especially for $\|\mathbf{e}_k\| \leq 1$), the acceleration effect of IAE will be more obvious than that of ISE to update λ_k faster. For JEU (23), it is similar to ISE when $\omega_1 = 1$ and $\omega_2 = 0$. While, for the case of $\omega_1 = 0, \omega_2 = 1$, λ_k increases (as $\nabla_\lambda H \leq 0$) to limit $\Delta\mathbf{r}_k$, which can be explained that the elastic rod is deformed in a slower manner. When neither is zero, the elastic rod is deformed with the given weights considered.

Remark 6. The proposed framework cannot guarantee to deform all target shapes. As $\mathbf{L}_k$ is never full rank, for
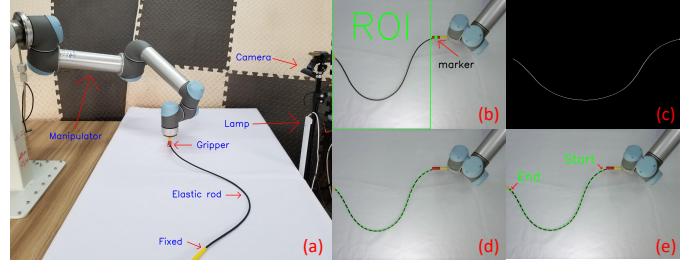

Fig. 4. Experimental setup and image processing steps.

unfeasible targets, the feedback shape can only converge to nearby neighborhoods of $\mathbf{s}^*$. For these types of overdetermined visual servoing controllers (i.e., with more output features than input controls), global asymptotic convergence of $\|\mathbf{e}_k\|$ cannot be guaranteed [21].

IV. RESULTS

A. Setup

This section validates the proposed regression features and the adaptive servoing controller. As shown in Fig. 4a, the experimental platform includes an elastic rod (with one fixed end), a UR5 robot constrained to move over the horizontal "xy" plane, and a static Logitech C270 camera to observe the scene. Visual feedback is processed with the OpenCV libraries in a Linux-based PC at 30 fps. A video with the conducted experiments can be downloaded from https://github.com/q546163199/experiment_video/raw/master/paper1/video.mp4.

The centroid of the gripper's red fiducial marker is highlighted by a green point, and is used to segment the region of interest (ROI) that contains the elastic rod (see Fig. 4b). The *OpenCV/Open* function used to denoise the ROI and obtain a binary image, which is further processed by the *OpenCV/Thinning* function to obtain the unordered centerline of the rod's skeleton (see Fig. 4c). Farthest Point Sampling is used to extract a fixed number of centerline points (see Fig. 4d). The closest point to the gripper's marker is chosen as the starting point of the centerline parametric curve (see Fig. 4e).

B. Comparison of the Feature Extraction Methods

In these experiments, the robot is first commanded to continuously deform the elastic rod to collect 30 sets of centerline observations, which are then used to verify the performance of the proposed parameterized regression features. Table I compares the error magnitude $\|\bar{\mathbf{c}} - \mathbf{Bs}\|$, which quantifies the accuracy between the feedback centerline $\bar{\mathbf{c}}$ and the reconstructed centerline $\mathbf{Bs}$; The results also compare the computation time required for each method. It can be seen that NURBS has the highest accuracy, while the polynomial has the lowest. The sinusoidal is the fastest, while NURBS is the slowest, yet, it still enables to implement servo-loops of well above 30Hz. Hence, throughout the rest of the experiments, we use NURBS (with $n = 8$) to characterize and control the feedback shapes of the manipulated elastic object.

C. Estimation of the Deformation Jacobian Matrix

In this section, we evaluate the performance (and thus, the suitability) of the UKF algorithm to estimate the system's

TABLE I
COMPARISON RESULTS AMONG SINUSOIDAL, POLYNOMIAL, BÉZIER AND NURBS AMONG 30 SHAPE SETS

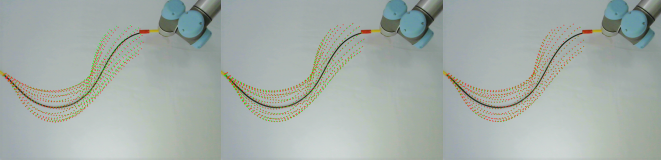|	**B**	**s**	Order	Average time	Average error
Sinusoidal	200x18	18	8	0.002s	8.1965px
Polynomial	200x18	18	8	0.004s	16.5667px
Bézier	200x18	18	8	0.007s	10.1931px
NURBS	200x18	18	8	0.011s	4.5628px

(a) RLS (b) LKF (c) UKF

Fig. 5. Comparison between the visually measured shape (dashed green line) and its approximation shape with NURBS series (dashed red line).

differential model. For that, the robot is commanded to move along a circular trajectory while rigidly grasping the elastic rod; These actions produce feedback shapes that progressively deform into a wide variety of configurations. During these motions, we simultaneously approximate the model (9) with 3 different methods: UKF, recursive least squares (RLS) [32], and linear Kalman filter (LKF) [16]. Fig. 5 depicts three measured shapes (dashed green line) and their corresponding approximated shapes (dashed red line), all computed with NURBS. The following two errors metric are used to evaluate the estimation algorithms:

$$T_1 = \|\widehat{\bar{\mathbf{c}}}_k - \bar{\mathbf{c}}_k\|, \quad T_2 = \|\Delta \mathbf{s}_k - \widehat{\mathbf{J}}_k \Delta \mathbf{r}_k\| \quad (32)$$

where $\widehat{\bar{\mathbf{c}}}_k = \mathbf{B}\widehat{\mathbf{s}}_k$ is the approximated shape, with $\widehat{\mathbf{s}}_k$ calculated as $\widehat{\mathbf{s}}_k = \widehat{\mathbf{s}}_{k-1} + \widehat{\mathbf{J}}_k \Delta \mathbf{r}_k$, with $\widehat{\mathbf{s}}_0 = \mathbf{s}_0$ for $\widehat{\mathbf{s}}_0$ and $\mathbf{s}_0$ as the initial values of $\widehat{\mathbf{s}}_k$ and $\mathbf{s}_k$, respectively. Fig. 6 shows the evolution of T_1 and T_2 during the manipulation motions. Compared with RLS and LKF, we can see that UKF has a higher accuracy in the estimation of the Jacobian matrix, with both errors rapidly converging to a small steady error. The lack of noticeable fluctuations in the UKF-computed plots reflects the adaptability of the algorithm to different local regions.

D. Shape Servoing of Elastic Rods

In this section, six shape control experiments with a variety of initial and desired configurations are conducted with the proposed controller (17) with the fixed gain ($\lambda = 2.8$), combined with RLS [32], LKF [16], and UKF. The target shapes are obtained by driving the robot to an arbitrary position and recording the corresponding feature vector $\mathbf{s}^*$ (this ensures the reachability of the task). No other information (e.g., the corresponding robot pose $\mathbf{r}$) is used in these control experiments where the robot automatically drives the cable to the desired configuration, by using visual feedback only. In these tests, the following safety velocity limits are used for each motion coordinate of the robot: $\|\Delta r_k\| \leq 0.01$ m/s.

Fig. 7 depicts the active shaping motions (solid green lines) of the elastic rod towards the desired shape (solid red line) of each control experiment with the three estimation algorithms. These visually-guided motions demonstrate that the proposed framework (representation, approximation, and control) can automatically drive the object to different shape
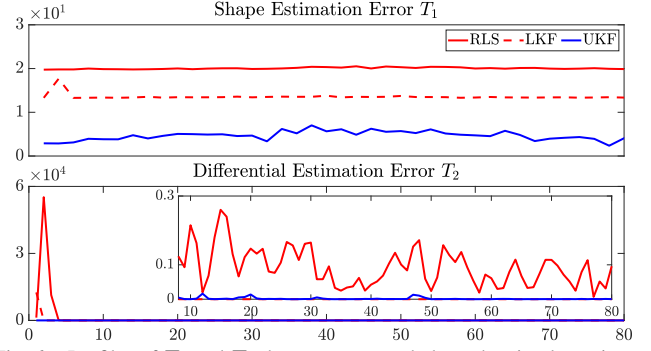

Fig. 6. Profiles of T_1 and T_2 that are computed along the circular trajectory. The abscissa is the step size.

configurations. Fig. 8 shows the profiles of $\mathbf{e}_k$ and $\Delta \mathbf{r}_k$ that were computed during the experiments. The figures show that UKF presents the best performance and RLS the worst. UKF can accurately estimate the Jacobian matrix, which helps to drive the robot $\Delta \mathbf{r}_k$ in the correct directions that minimize the vision error $\|\mathbf{e}_k\|$. Fig. 7 also shows that NURBS can adequately approximate the object's centerline. These results demonstrate the controller can guide the robot to deform the objects into multiples shapes without damaging it (i.e., over-stretching, or over-compressing).

To thoroughly test the performance of the proposed method, we also tested the following two fault conditions were the robot cannot perform the shape servoing task (shown in Fig. 9): 1) If estimated Jacobian matrix has not be properly and sufficiently initialized at the time instant zero $\widehat{\mathbf{J}}_0$ (see Remark 2), the motion controller may drive the robot in the wrong directions. 2) If the target feature vector $\mathbf{s}^*$ represents an unfeasible shape (i.e., a configuration that cannot be achieved with the current object-manipulation setup), the controller cannot drive the object towards it, and it will typically reach a local minimum.

E. Parameter Optimization Comparison

In this section, the performance of the parameter optimization criteria (ISE (18), IAE (20) and JEU (22)) is compared with the traditional fixed gain (FIX) control method (i.e., for the case where $\lambda = 7.2$ is constant). To evaluate the impact of the weights in JEU (22), the following three sets of parameters are used: 1) JEU-1: $\omega_1 = 0.5, \omega_2 = 0.5$, in this case, the effect is similar to IAE; 2) JEU-2: $\omega_1 = 0.1, \omega_2 = 0.9$, which pays more attention to the smoothness of $\Delta \mathbf{r}_k$; 3) JEU-3: $\omega_1 = 0.9, \omega_2 = 0.1$, which improves the convergence of $\mathbf{e}_k$; The adaptive gain is initialized as follows $\lambda_0 = 7.2$ and is updated with a learning rate gain $d = 0.001$. In these experiments, the upper limit of the velocity command for each motion coordinate of the robot is set to $\|\Delta r_k\| \leq 0.02$ m/s. The performed motions for this comparison study are shown in Fig. 10. The profiles of $\mathbf{e}_k$ and $\Delta \mathbf{r}_k$ are given in Fig. 11.

TABLE II
COMPARISONS AMONG FIX, ISE, IAE, JEU-1, JEU-2, AND JEU-3

	FIX	ISE	IAE	JEU-1	JEU-2	JEU-3
Steps	58	15	34	40	82	26
Time	66.71s	17.18s	40.12s	47.17s	97.58s	29.12s

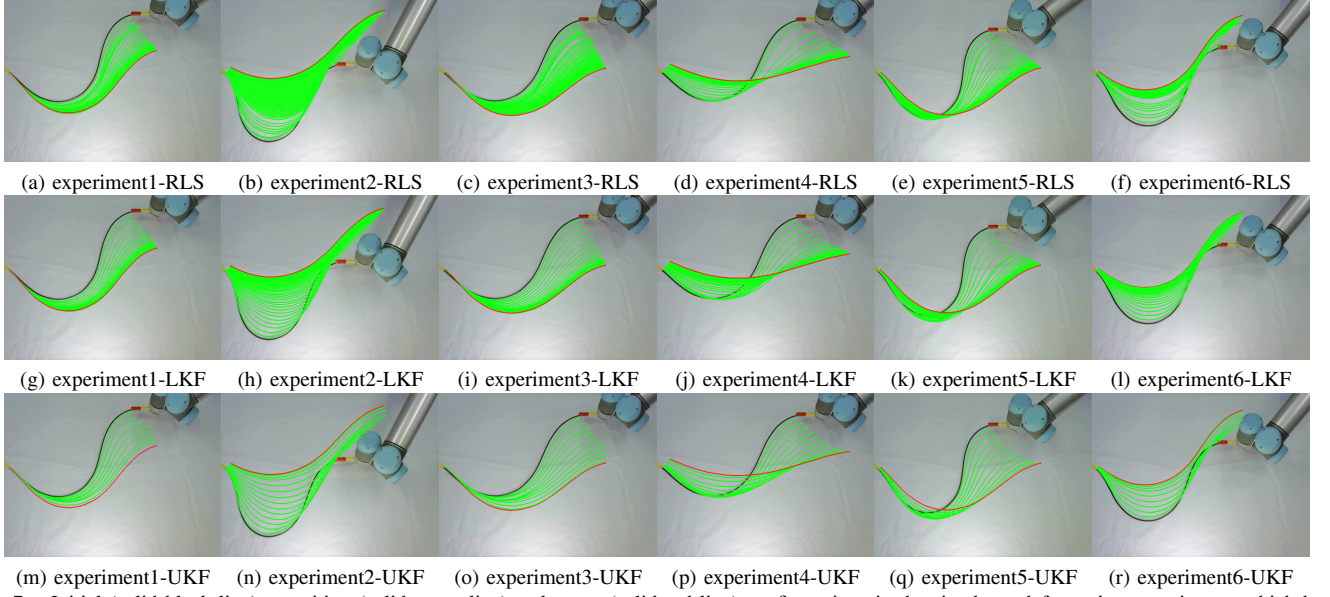Table II presents performance metrics obtained with the different parameter optimization methods. The table shows that

(a) experiment1-RLS   (b) experiment2-RLS   (c) experiment3-RLS   (d) experiment4-RLS   (e) experiment5-RLS   (f) experiment6-RLS

(g) experiment1-LKF   (h) experiment2-LKF   (i) experiment3-LKF   (j) experiment4-LKF   (k) experiment5-LKF   (l) experiment6-LKF

(m) experiment1-UKF   (n) experiment2-UKF   (o) experiment3-UKF   (p) experiment4-UKF   (q) experiment5-UKF   (r) experiment6-UKF

Fig. 7. Initial (solid black line), transition (solid green line) and target (solid red line) configurations in the six shape deformation experiments which have a variety of initial and target shape. These experiments are conducted within the motion controller (17) among RLS [32], LKF [16] and UKF.
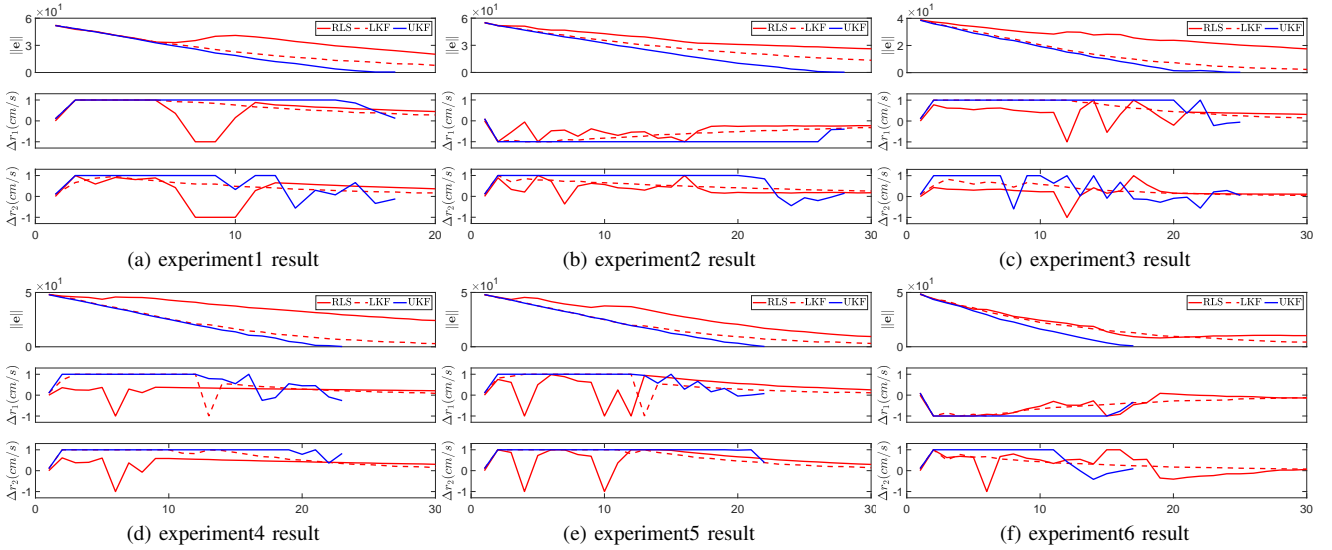

(a) experiment1 result   (b) experiment2 result   (c) experiment3 result

(d) experiment4 result   (e) experiment5 result   (f) experiment6 result

Fig. 8. Profiles of $\|\mathbf{e}_k\|$ and $\Delta\mathbf{r}_k$ among RLS, LKF and UKF within six shape deformation experiments. The abscissa is the step size.

(a) Fault case 1   (b) Fault case 2
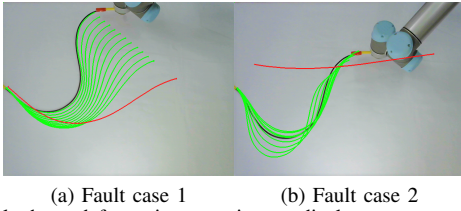
Fig. 9. Fault shape deformation experiments display.

(a) FIX   (b) ISE   (c) IAE
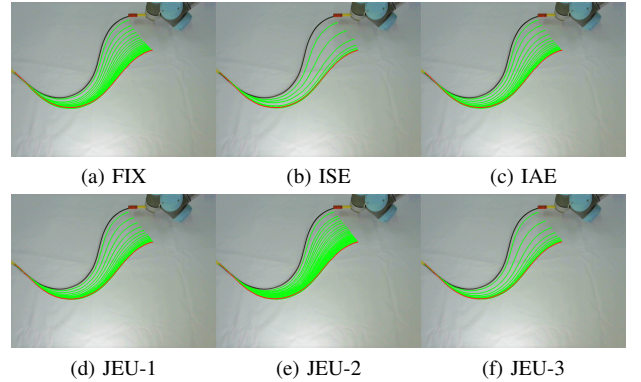
(d) JEU-1   (e) JEU-2   (f) JEU-3

Fig. 10. Initial (solid black line), transition (solid green line) and target (solid red line) configurations among FIX, ISE, IAE, JEU-1 ($\omega_1 = 0.5, \omega_2 = 0.5$), JEU-2 ($\omega_1 = 0.1, \omega_2 = 0.9$), and JEU-3 ($\omega_1 = 0.9, \omega_2 = 0.1$). The comparisons are conducted within LKF [16].

ISE has the shortest convergence time, followed by JEU-3, while JEU-2 is the slowest. As ISE focuses on compensating errors, the resulting motion command $\Delta\mathbf{r}_k$ presents larger fluctuations. Fig. 11 shows that JEU-2 provides slower convergence with smoother profiles $\Delta\mathbf{r}_k$ than JEU-3; The controls $\Delta\mathbf{r}_k$ computed with JEU-3 are relatively larger and close to saturation. This shows that by varying the weights of JEU, we can tune the system to match various target performances. The above analyses demonstrate that the proposed parameter optimization criteria can effectively adjust λ according to different task requirements.

V. CONCLUSION

This paper presents a shape servoing framework for automatically manipulating an elastic rod to desired configurations.
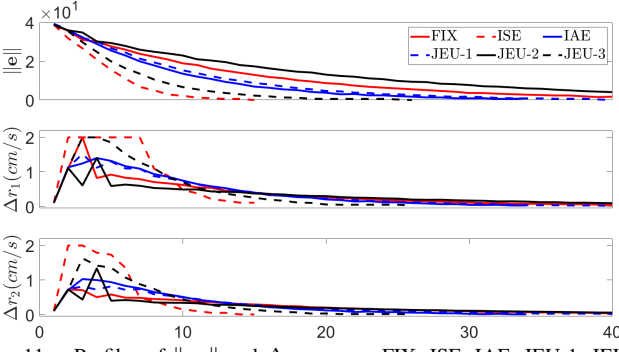
Fig. 11. Profiles of $\|\mathbf{e}_k\|$ and $\Delta\mathbf{r}_k$ among FIX, ISE, IAE, JEU-1, JEU-2, and JEU-3. The abscissa is the step size.

A general method based on parametric features (e.g. sinusoisal, polynomial, Bézier, and NURBS) is presented to characterize the shape of the object with a compact feedback-like vector (on which an explicit servo-loop is established upon). A UKF-based online estimator is proposed to coordinate the driving motions of the robot with the produced visual measurements. An adaptive velocity controller is derived considering various performance criteria. The stability of the system is rigorously analyzed using Lyapunov theory. A detailed experimental study is presented to validate the effectiveness of the new control method.

The proposed method has some limitations. First, the proposed framework is only suitable to manipulate mostly elastic materials (i.e., those whose shape is mainly determined by its potential energy); The method does not consider inelastic or non-homogeneous materials. Second, the method cannot judge if the desired configuration is reachable, which can lead to failure during the task execution. Third, the method requires a full (i.e., un-occluded) visual observation of the object at all times. Future work includes the design of similar control schemes which can handle occlusions of the object; Our team is currently developing a neural network which can predict the shape of the object based on partial observations.

REFERENCES

[1] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, X. Li, J. Pan, and W. Yuan, "Challenges and outlook in robotic manipulation of deformable objects," *IEEE Robot. Autom. Magazine (in press)*, pp. 1–11, 2022.

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

[3] S. Tokumoto and S. Hirai, "Deformation control of rheological food dough using a forming process model," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1457–1464.

[4] X. Li, X. Su, and Y.-H. Liu, "Vision-based robotic manipulation of flexible PCBs," *IEEE/ASME Trans. on Mechatronics*, vol. 23, no. 6, pp. 2739–2749, 2018.

[5] H. Wakamatsu, E. Arai, and S. Hirai, "Knotting/unknotting manipulation of deformable linear objects," *The International Journal of Robotics Research*, vol. 25, no. 4, pp. 371–395, 2006.

[6] 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," *Int. J. Robot. Res.*, vol. 37, no. 7, pp. 688–716, 2018.

[7] D. Navarro-Alarcon, Y.-h. Liu, J. G. Romero, and P. Li, "On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments," *The International Journal of Robotics Research*, vol. 33, no. 11, pp. 1462–1480, 2014.

[8] F. Chaumette, "Image moments: a general and useful set of features for visual servoing," *IEEE Transactions on Robotics*, vol. 20, no. 4, pp. 713–723, 2004.

[9] Z. Hu, T. Han, P. Sun, J. Pan, and D. Manocha, "3-d deformable object manipulation using deep neural networks," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, 2019.

[10] M. Laranjeira, C. Dune, and V. Hugel, "Catenary-based visual servoing for tethered robots," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 732–738.

[11] ——, "Catenary-based visual servoing for tether shape control between underwater vehicles," *Ocean Engineering*, vol. 200, p. 107018, 2020.

[12] 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, 2018.

[13] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2146–2153.

[14] D. Navarro-Alarcon, Y.-H. Liu, J. G. Romero, and P. Li, "Model-free visually servoed deformation control of elastic objects by robot manipulators," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1457–1468, 2013.

[15] F. Alambeigi, Z. Wang, R. Hegeman, Y.-H. Liu, and M. Armand, "Autonomous data-driven manipulation of unknown anisotropic deformable tissues using unmodelled continuum manipulators," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 254–261, 2018.

[16] J. Qian and J. Su, "Online estimation of image jacobian matrix by kalman-bucy filter for uncalibrated stereo vision feedback," in *IEEE International Conference on Robotics Automation*, 2002.

[17] M. A. Ibrahim, A. K. Mahmood, and N. S. Sultan, "Optimal pid controller of a brushless dc motor using genetic algorithm," *Int J Pow Elec & Dri Syst ISSN*, vol. 2088, no. 8694, p. 8694, 2019.

[18] M. M. Kamal, L. Mathew, and S. Chatterji, "Speed control of brushless dc motor using intelligent controllers," in *2014 Students Conference on Engineering and Systems*. IEEE, 2014, pp. 1–5.

[19] J. Huang, Y. Cai, X. Chu, R. H. Taylor, and K. S. Au, "Non-fixed contact manipulation control framework for deformable objects with active contact adjustment," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2878–2885, 2021.

[20] H. Wang, B. Yang, J. Wang, X. Liang, W. Chen, and Y. Liu, "Adaptive visual servoing of contour features," *IEEE/ASME Trans. on Mechatronics*, vol. 23, no. 2, pp. 811–822, 2018.

[21] F. Chaumette and S. Hutchinson, "Visual servo control. Part I: Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, 2006.

[22] B. Siciliano, "Kinematic control of redundant robot manipulators: A tutorial," *Journal of Intelligent and Robotic Systems*, vol. 3, no. 3, pp. 201–212, 1990.

[23] L. Piegl and W. Tiller, *The NURBS book*. Springer Science & Business Media, 2012.

[24] M. J. D. Powell *et al.*, *Approximation theory and methods*. Cambridge university press, 1981.

[25] K. Xiong, H. Zhang, and C. Chan, "Performance evaluation of ukf-based nonlinear filtering," *Automatica*, vol. 42, no. 2, pp. 261–270, 2006.

[26] C. K. Chui, G. Chen *et al.*, *Kalman filtering*. Springer, 2017.

[27] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on automatic control*, vol. 45, no. 3, pp. 477–482, 2000.

[28] M. Zhuang and D. P. Atherton, "Automatic tuning of optimum pid controllers," *Control Theory Applications Iee Proceedings D*, vol. 140, no. 3, pp. 216–224, 1993.

[29] W. Xiong, B. Xu, and Q. Zhou, "Study on optimization of pid parameter based on improved pso," *Computer Engineering*, vol. 31, no. 24, pp. 41–43, 2005.

[30] S. Sarpturk, Y. Istefanopulos, and O. Kaynak, "On the stability of discrete-time sliding mode control systems," *IEEE Transactions on Automatic Control*, vol. 32, no. 10, pp. 930–932, 1987.

[31] J. Ma, S. S. Ge, Z. Zheng, and D. Hu, "Adaptive nn control of a class of nonlinear systems with asymmetric saturation actuators," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 7, pp. 1532–1538, 2014.

[32] K. Hosoda and M. Asada, "Versatile visual servoing without knowledge of true jacobian," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, vol. 1. IEEE, 1994, pp. 186–193.