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*, Wanyu Ma, Peng Zhou, *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' non-linear properties and high-dimensional geometric configuration. In this paper, we propose an efficient shape servoing framework to manipulate elastic objects through real-time visual feedback automatically. The proposed parameterized regression features are used to construct a compact (low-dimensional) feature vector (Bézier and NURBS) that quantifies the object's shape, thus enabling the establishment of an explicit shape servo-loop. To automatically manipulate the object into a desired configuration, our adaptive controller can iteratively estimate the sensorimotor model that relates the robot's motion and shape changes. This valuable capability enables the effective deformation of objects with unknown mechanical models. An auto-tuning algorithm is introduced to adjust the controller's gain and, thus, modulate the shaping motions based on optimal performance criteria. To validate the proposed framework, a detailed experimental study with vision-guided robot 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 soft 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 traditionally 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 deformable objects might be delicate and thus easy to damage).

In shape servoing tasks, feature extraction is performed to describe the object's configuration with low-dimensional feedback coordinates [7]. Traditional forms to represent deformations include: centroids, distance, angles, curvature, etc [8]. 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 [9] to characterize contours, but with a limited real-time performance as its complicated calculations. Principal component analysis was used to project raw Fast Point Feature Histograms into a new space with higher variance and a lower dimension [10]. A catenary-based feature descriptor was developed for tethered robots in [11], however, it is only for specific shapes. A Fourier-based approach was developed in [7] to characterize contours with low-dimensional features. Bézier and Non-Uniform Rational Basis Splines (NURBS) are exciting options to describe complex shapes. Yet, they have not been sufficiently studied in the literature to establish an explicit shape servo-loop. Contour moments were used in [12] to represent the object by removing the moment invariance; however, this method was limited to a 2D environment. Other types of representation methods rely on machine learning, e.g., [13]; These approaches typically require large data sets to generalize to different situations (which may be difficult to guarantee in many applications). A computationally efficient feature extraction algorithm is a critical problem in shape servoing.

To perform shape servoing tasks, a feedback controller requires a model that captures the relationship between the robot's motions and the produced object's deformations [14]. A diminishing rigidity model was used in [15] to calculate the deformation Jacobian matrix. This method required prior (coarse) knowledge of the object's properties to specify the decreasing rate correctly. Function approximation technique (FAT) was developed in [16] to estimate the sensorimotor model, yet it depends on the choice of the activation function. There are other approaches that do not require prior knowledge of the structure, e.g., in [17], [18] a Broyden-like method was used to online estimate this deformation matrix. Although these types 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, [19]

This work is supported by the Research Grants Council of Hong Kong under grant number 15212721 and grant 15231023.

Jiaming Qi and Guangtao Ran are with Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China. (e-mail: qijm_hit@163.com; 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).

Wanyu Ma and David Navarro-Alarcon are with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong. (e-mail: wanyu.ma@connect.polyu.hk, dnavar@polyu.edu.hk).

P. Zhou is with the Department of Computer Science, University of Hong Kong, Pok Fu Lam, Hong Kong. (e-mail: jeffzhou@hku.hk)

constructed a state-space model with the elements of Jacobian matrix and used KF to 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 behaviors than rigid counterparts, therefore, it is advantageous to control their transient behavior. This can be achieved through performance adjustment methods, which have been used in motor speed regulation problems [20] to achieve accurate tracking while considering various performance criteria (e.g., overshoot, rising, settling time) [21]. 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 overstretching and over-compression of an object [22]. 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 vision-based framework to deform elastic objects into desired 2D shapes automatically. To quantify the configuration of these DLOs, a general parametric-based extraction framework is proposed using linear regression, which enables representing shapes with a compact feedback-like vector. To the best of the authors' knowledge, this is the first time that a shape controller uses regression features to establish an *explicit* shape servo-loop (Bézier/NURBS were used in [23], but only to project a contour into an image plane, which vastly differs from our approach). An unscented KF is used to iteratively estimate the unknown nonlinearity between the robot and the object, which obtains the estimation sensorimotor model to guide the robot. The controller's parameters are adjusted online using the optimization algorithm, which modulates the transient response.

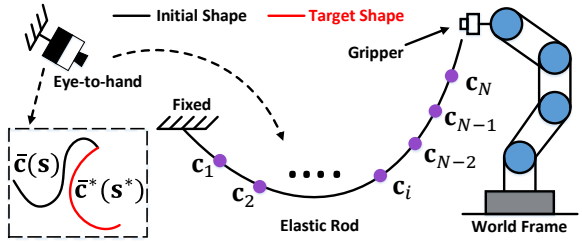


Fig. 1. Representation of active manipulation of the elastic rod, where a vision sensor continuously measures the feature vector $\mathbf{s}$, which should be accurately deformed into target feature $\mathbf{s}^*$ within the controller.

II. Preliminaries

Notation. Column vectors are denoted with bold small letters $\mathbf{m}$ and matrices with bold capital letters $\mathbf{M}$. 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.

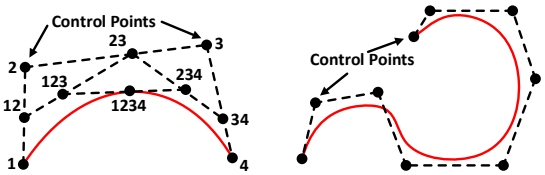
To derive our method, some conditions are now introduced:

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

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

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 $\Delta\mathbf{r}_k \in \mathbb{R}^q$ 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 choice of $\mathbf{r}_k$ depends on the task requirement (joint or 2D/3D pose).
- 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.



(a) Bézier (b) NURBS

Fig. 2. Graphical representation of the Bézier and NURBS curves.

III. Methods

A. Feedback Shape Parameters

The *naïve approach* to shape servoing is to synthesize 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 modeled 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, yet, 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 [24]. It has local shape description properties, thus, the number of control points is independent of the degree of curves. This model can describe 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 [25]. 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 [7].

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

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

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:

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

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.

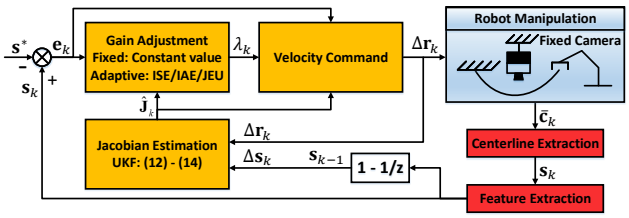


Fig. 3. The block diagram showing the overall control workflow.

Remark 1. Although only four regression functions are given, there are many other curve descriptors (e.g., B-spline, rational approximation, etc.) that can be expressed as $\bar{\mathbf{c}} = \mathbf{B}\mathbf{s}$ to represent shapes with varying complexity. Furthermore, the regression feature (8) can also be applied to 3D configurations.

B. UKF-Based Approximation of the Deformation Model

Since we consider regular (i.e. mechanically well-behaved without snapping) elastic objects, it is reasonable to assume that small motions $\Delta\mathbf{r}_k$ will produce small changes $\Delta\bar{\mathbf{c}}_k$. We locally model this situation (around the 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:

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

We use a modified UKF [26] (which has good dynamic performance and robustness to external disturbances) 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 $\widehat{\mathbf{x}}_{k-1}$ and covariance $\widehat{\mathbf{P}}_{k-1}$ is estimated by sigma points defined by

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

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. The positive-definite matrix $\Delta\mathbf{U}_k$ is introduced to improve the approximation stability [26]. Sigma points pass through the process model to generate transformed points:

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

where $\varpi_0 = 1 - \frac{1}{a^2}, \varpi_i = \frac{1}{2pqa^2}, i = 1, \ldots, 2pq$, are scalar weights and $\sum_{i=0}^{2pq} \varpi_i = 1$.

Step 3: Update. The measurement residual and Kalman gain are defined as $\tilde{\mathbf{y}}_k = \mathbf{y}_k - \widehat{\mathbf{y}}_k$ and $\mathbf{K}_k = \widehat{\mathbf{P}}_{xy,k}\widehat{\mathbf{P}}_{yy,k}^{-1}$, respectively. Classical KF can be used to update the measurement signal as follows:

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

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

After completing an iteration, we de-vectorize $\widehat{\mathbf{x}}_k$ to get an estimation $\widehat{\mathbf{J}}_k$ of the unknown Jacobian matrix.

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 [26] (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:

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

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 $\Delta\mathbf{r}_k$:

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

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 result in slow motions, while decreasing λ may cause overshoots (due to the excessive error). Therefore, it is beneficial to automatically adjust λ to achieve an optimal dynamic response. This is done by using gradient-descent parameter optimization 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:

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

which is used for computing the parameter tuning criterion:

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

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:

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

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]:

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

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

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

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).

Theorem 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 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]:

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

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

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

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:

$$-\mathbf{e}_{k-1}^\intercal \mathbf{L}_k \mathbf{e}_{k-1} + \mathbf{e}_{k-1}^\intercal \tilde{\mathbf{J}}_k \Delta \mathbf{r}_k + \frac{1}{2}\Delta \mathbf{r}_k^\intercal \mathbf{J}_k^\intercal \mathbf{J}_k \Delta \mathbf{r}_k \leq 0 \tag{26}$$

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:

$$-\mathbf{e}_{k-1}^\intercal \tilde{\mathbf{J}}_k \Delta \mathbf{r}_k \leq \frac{1}{2}\|\mathbf{e}_{k-1}\|^2 + \frac{1}{2}\epsilon_J^2 \epsilon_r^2 \tag{27}$$

Substituting (27) into (26), and after some algebraic and bounds operations, the following relations can be derived:

$$\epsilon_r^2 \bar{\beta}\left(\mathbf{J}_k^\intercal \mathbf{J}_k\right) - \epsilon_J^2 \epsilon_r^2 \leq \left(2\bar{\beta}\left(\mathbf{L}_k\right) + 1\right)\|\mathbf{e}_{k-1}\|^2 \tag{28}$$

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

$$\Omega = \left\{ \mathbf{e}_k \;\middle|\; 2V_k = \|\mathbf{e}_k\|^2 \leq \frac{\epsilon_r^2\left(\bar{\beta}\left(\mathbf{J}_k^\intercal \mathbf{J}_k\right) - \epsilon_J^2\right)}{2\bar{\beta}\left(\mathbf{L}_k\right) + 1} \right\} \tag{29}$$

It is assumed that UKF has been sufficiently iterated to provide a small enough ϵ_J that ensures that $\bar{\beta}\left(\mathbf{J}_k^\intercal \mathbf{J}_k\right) - \epsilon_J^2 > 0$ (see Remark 2). From (29), 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. From the practical view, condition (29) implies that $\mathbf{e}_k$ doesn't entirely translate into convergence; it provides a notion that $\mathbf{e}_k$ is not too big, which reflects that the final shape of the object is "close" to the target shape within an allowable range. This means that the manipulation results meet the task requirements.

Remark 6. 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 7. The proposed framework cannot guarantee to deform all target shapes. As $\mathbf{L}_k$ is never full rank, for 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 [32].

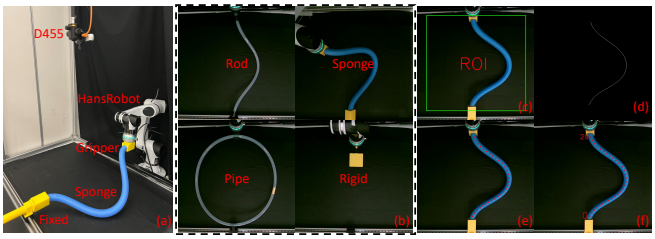


Fig. 4. Experimental setup and image processing steps.

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 various objects (rod, sponge, pipe, and rigid, shown in Fig. 4b) with one fixed end, a HansRobot arm constrained to move over the horizontal "xy" plane and rotate around z-axis, and a D455 camera (eye-to-hand) to observe the scene. Visual feedback is processed with OpenCV on 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.

Firstly, the region of interest (ROI) that contains the elastic rod is chosen according to the mission requirement (see Fig. 4c). Then, the *OpenCV/Open* function is 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. 4d). Subsequently, farthest point sampling extracts a fixed number of centerline points (see Fig. 4e). Finally, the closest point to the fixed end is chosen as the starting point of the centerline parametric curve (see Fig. 4f).

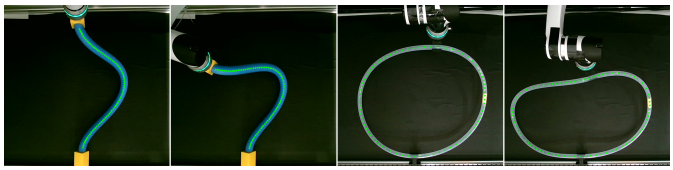


Fig. 5. Shapes of various configurations manipulated by the robot.

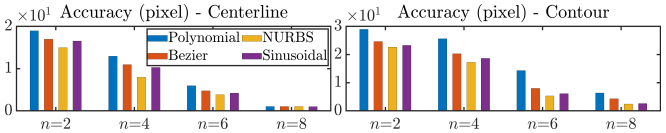


Fig. 6. Comparison results among four regression methods among 20000 shape sets. For NURBS, we set $\omega_j = 0.327, j \in [0, n]$ as the optimal value. The abscissa is the fitting order n.

B. Comparison of the Feature Extraction Methods

In this section, 20,000 sets of centerline and contour observations (with $N = 80$, 10,000 for each) are collected by commanding the robot to continuously deform the elastic object, which is then used to evaluate the proposed parameterized regression features. The shape sets cover a more

extensive range, which can better verify the generalization of the regression feature. Such shaping actions are shown in Fig. 5 and visualized in the accompanying multimedia attachment. For that, the average error between the feedback shape $\bar{\mathbf{c}}$ and the reconstructed shape $\mathbf{Bs}$ is calculated as $\frac{1}{1e4}\sum_{i=1}^{1e4}\|\bar{\mathbf{c}}_i - \mathbf{Bs}_i\|$, and the computation time required for each method is also given.

Fig. 6 shows that as n increases, the extraction works better, further helping to represent the object's shape. As for the same n, NURBS performs best; Sinusoidal and Bézier are similar, and Polynomial is prone to overfitting with higher n. Theoretically, the performance of NURBS can be improved by adjusting ω_j. The sinusoidal performance is most obvious as n increases, and when n is large enough, the sinusoidal is similar to NURBS. This illustrates the effectiveness of sinusoidal series in approximating the function. The features generated by Bézier and NURBS are physically interpretable, i.e., fitting points, which may be suitable for some special occasions. Four regression methods meet the real-time requirement; the slowest is NURBS (0.026s) with the most iterative calculations, yet it still enables the implementation of 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.

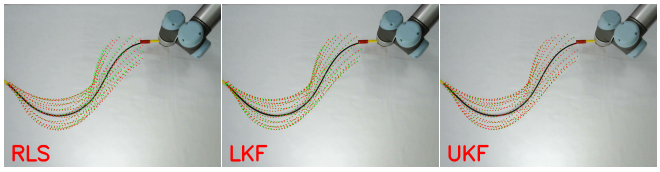


Fig. 7. Comparison between the visually measured shape (dashed green line) and its approximation shape with NURBS series (dashed red line).

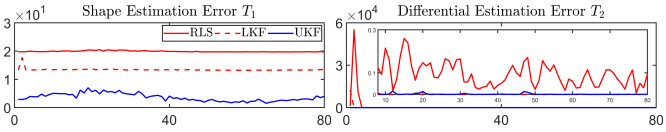


Fig. 8. Profiles of T_1 and T_2 that are computed along the circular trajectory. The abscissa is the step size.

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 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) [33], and linear Kalman filter (LKF) [19]. Fig. 7 depicts three measured shapes (dashed green line) and their corresponding approximated shapes (dashed red line), all computed with NURBS. The following two error metrics are used to evaluate the estimation algorithms:

$$T_1 = \|\widehat{\bar{\mathbf{c}}}_k - \bar{\mathbf{c}}_k\|, \quad T_2 = \|\Delta\mathbf{s}_k - \widehat{\mathbf{J}}_k\Delta\mathbf{r}_k\| \tag{30}$$

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. 8 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 Various Objects

In this section, four shape control experiments (rod, sponge, pipe, and rigid) with a variety of initial and desired configurations are conducted with the proposed controller (17) with a fixed gain ($\lambda = 2.8$), and comparing UKF with FAT from [16] and GML from [1]. Exp1-Exp3 are for deformation tasks, while Exp4 is for the positioning task with the rigid object; Exp3 and Exp4 use contour as the object outline. The target shapes are obtained by driving the robot to an arbitrary configuration and recording the corresponding feature vector $\mathbf{s}^*$ (this ensures the target's reachability). No other information (e.g., the corresponding robot pose $\mathbf{r}$) is used in these control experiments, where the robot automatically deforms/drives the objects towards the targets by using visual feedback only. In these tests, the linear speed is limited to 0.01m/s and the joint speed to 5°/s to meet the assumptions proposed in Sec. II and ensure the validity of the local Jacobian estimation.

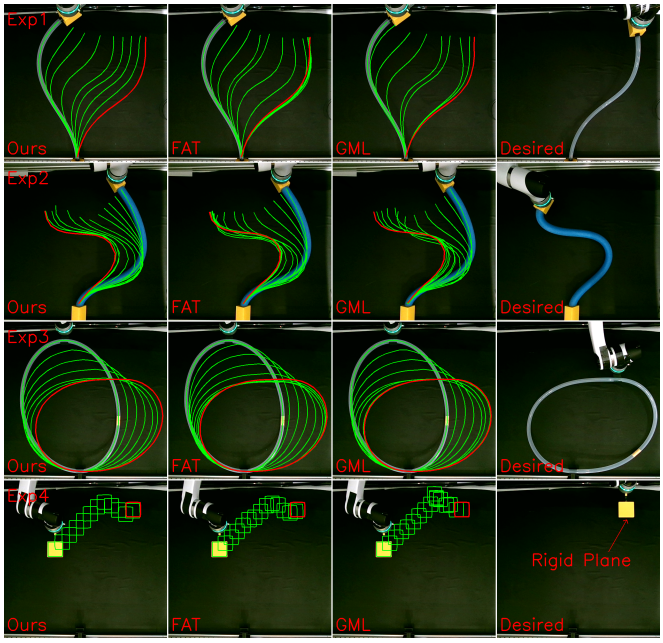


Fig. 9. Initial, transition (solid green line), and target (solid red line) configurations in the four shape manipulation experiments which have a variety of initial and target shapes. These experiments are conducted with the motion controller (17) and UKF, compared with FAT [16], and GML [1].

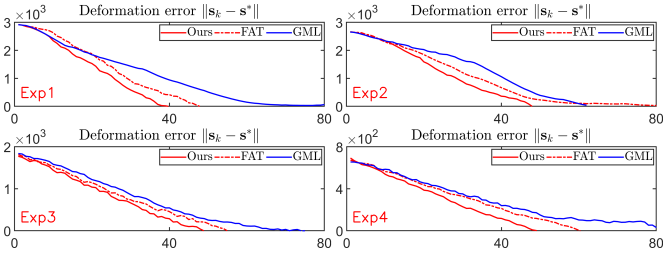


Fig. 10. Profiles of $\|\mathbf{e}_k\|$ from Ours, FAT [16], and GML [1] with four shape manipulation experiments. The abscissa is the step size.

Fig. 9 depicts the active shaping motions (solid green lines) of the object towards the desired shape (solid red

line) of the three manipulation methods; the fourth column in each row represents the desired shape. These visually-guided motions demonstrate that the proposed framework (representation, approximation, and control) can automatically drive the object to different shape configurations and have the versatility to be used with different materials. The results also show that NURBS can adequately approximate the object's centerline and contour. This is particularly crucial for Exp4, whose goal is to evaluate the method combined with contour shapes to complete visual servoing tasks, which can reflect the parametric generability of the proposed controller.

Fig. 10 shows the error trajectories $\mathbf{e}_k$ that were computed during the experiments, which shows that our method provides the best results, with FAT performing better than GML. The results demonstrate that 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 $\|\mathbf{e}_k\|$. The successful motion tasks (Exp1 - Exp4) illustrate the versatility of the presented framework for shape servoing and positioning tasks. It can guide the robot to manipulate the objects into multiple configurations without damaging them (i.e., over-stretching or over-compressing).

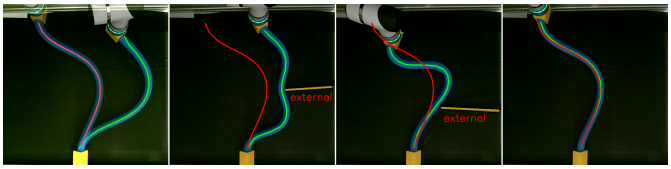


Fig. 11. Manipulation within the human interference.

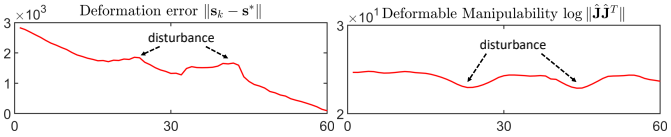


Fig. 12. Profiles of $\|\mathbf{e}_k\|$ and DM within human interference. The abscissa is the step size.

Fig. 11 shows manipulation results obtained in the presence of external disturbances. The solid red line is the desired shape; The yellow stick marked "external" is the human-introduced disturbance, which modifies the object's shape during the task. Fig. 12 gives the profiles of $\mathbf{e}_k$ and deformable manipulability (DM) [16] (it evaluates the feasibility of changing the object's shape under the current object-robot configuration). The smaller the DM, the more singular estimated $\hat{\mathbf{J}}_k$ is. The results show that UKF can overcome the influence of nonlinear disturbance to a certain extent (the DM gets out from singularity after descending), and can quickly respond to sudden system deviations and control the robot back to normal ($\|\mathbf{e}_k\|$ drops after being disturbed), showing strong adaptability and robustness.

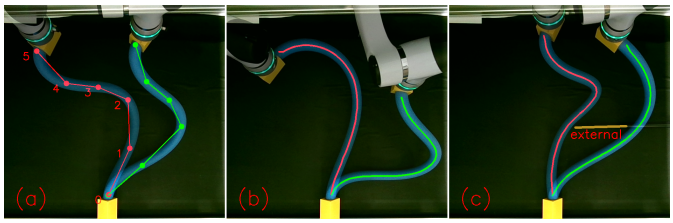


Fig. 13. Fault shape deformation experiments display.

To thoroughly evaluate the proposed framework, we test the following three fault conditions in which the robot cannot perform the shape servoing task (shown in Fig. 13): a) If the number of collected points is small ($N \leq n+1$), then $\mathbf{s}$ is not enough to represent the object, and it is easy to cause errors in subsequent Jacobian calculations and the final manipulation; b) If the estimated Jacobian matrix has not been 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; c) 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 exactly drive the object towards it, and it will typically reach a local minimum.

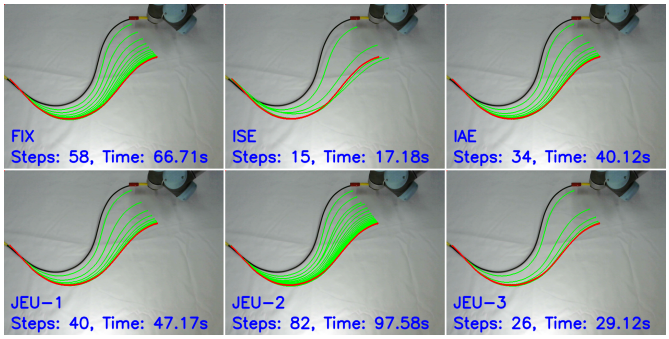


Fig. 14. Initial (solid black line), transition (solid green line), and target (solid red line) configurations among FIX, ISE, IAE, JEU-1 ($\omega_1 = 0.5, \omega_2 = 0.5$), JEU-2 ($\omega_1 = 0.1, \omega_2 = 0.9$), and JEU-3 ($\omega_1 = 0.9, \omega_2 = 0.1$). The comparisons are conducted within LKF [19].

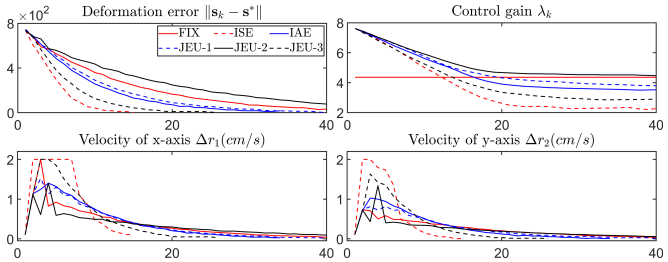


Fig. 15. Profiles of $\|\mathbf{e}_k\|$, λ_k, and $\Delta\mathbf{r}_k$ among FIX, ISE, IAE, JEU-1, JEU-2, and JEU-3. The abscissa is the step size.

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 $\lambda = 4.36$ 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 $\lambda_0 = 7.6$ and is updated with a learning rate gain $d = 0.017$. 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.

Fig. 14 presents manipulation results obtained with different parameter optimization methods. The results show that 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. 15 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 various profiles of λ_k show that it continuously adjusts the system performance as $\mathbf{e}_k$ changes within optimization regulation. The above analysis demonstrates that the proposed parameter optimization criteria can effectively adjust λ according to different *quantitative* task requirements.

V. Conclusion

This paper presents a shape servoing framework for automatically manipulating the elastic objects to desired configurations. A general representation framework based on parametric features is presented to characterize the object's shape with a compact feedback-like vector on which an explicit servo-loop is established. Different curve presentations (sinusoidal, polynomial, Bézier, and NURBS) are computed using linear regression. A UKF-based online estimator is proposed to coordinate the driving motions of the robot with the produced visual measurements and overcome external disturbances during the manipulation. An adaptive velocity controller is derived considering various performance criteria, which tune the system's performance according to the task requirement. The stability of the system is rigorously analyzed using Lyapunov theory. A detailed experimental study is presented to validate the manipulation framework's effectiveness.

The proposed framework has some limitations. First, it is only suitable to manipulate predominantly elastic materials (i.e., those whose shape is mainly determined by the 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 complete (i.e., un-occluded) visual observation of the object at all times and is limited to controlling 2D shapes and conducting 2D plane motions. Future work includes the design of similar control schemes that can handle occlusions of the object. Our team is developing a neural network that can predict the object's shape based on partial observations.

References

[1] M. Yu, K. Lv, H. Zhong, S. Song, and X. Li, "Global model learning for large deformation control of elastic deformable linear objects," *IEEE TRO*, vol. 39, no. 1, pp. 417–436, 2022.

[2] J. Huang and K. S. Au, "Task-oriented grasping position selection in deformable object manipulation," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 776–783, 2022.

[3] A. Cherubini, V. Ortenzi, A. Cosgun, R. Lee, and P. Corke, "Model-free vision-based shaping of deformable plastic materials," *The International Journal of Robotics Research*, vol. 39, no. 14, pp. 1739–1759, 2020.

[4] F. Zhang and Y. Demiris, "Visual-tactile learning of garment unfolding for robot-assisted dressing," *IEEE RAL*, 2023.

[5] Y. Qin, A. Escande *et al.*, "Dual-arm mobile manipulation planning of a long deformable object in industrial installation," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 3039–3046, 2023.

[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," *IJRR*, vol. 37, no. 7, pp. 688–716, 2018.

[7] 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 TRO*, vol. 34, no. 1, pp. 272–279, 2018.

[8] 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.

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

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

[11] M. Laranjeira, C. Dune, and V. Hugel, "Catenary-based visual servoing for tether shape control between underwater vehicles," *Ocean Engineering*, vol. 200, p. 107018, 2020.

[12] J. Qi, G. Ma, J. Zhu, P. Zhou, Y. Lyu, H. Zhang, and D. Navarro-Alarcon, "Contour moments based manipulation of composite rigid-deformable objects with finite time model estimation and shape/position control," *IEEE TMECH*, vol. 27, no. 5, pp. 2985–2996, 2021.

[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 ICRA*, 2017, pp. 2146–2153.

[14] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, p. eabd8803, 2021.

[15] K. Almaghout and A. Klimchik, "Planar shape control of deformable linear objects," *IFAC*, vol. 55, no. 10, pp. 2469–2474, 2022.

[16] L. Han, H. Wang, Z. Liu, W. Chen, and X. Zhang, "Visual tracking control of deformable objects with a fat-based controller," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 2, pp. 1673–1681, 2021.

[17] 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 RAL*, vol. 4, no. 2, pp. 254–261, 2018.

[18] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," *IEEE RAL*, vol. 5, no. 4, pp. 5252–5259, 2020.

[19] 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.

[20] 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.

[21] 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.

[22] 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 RAL*, vol. 6, no. 2, pp. 2878–2885, 2021.

[23] 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.

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

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

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

[27] J. Liu, J. Gao, W. Yan, Y. Chen, and B. Yang, "Image-based visual servoing of underwater vehicles for tracking a moving target using model predictive control with motion estimation," *International Journal of Vehicle Design*, vol. 91, no. 1-3, pp. 46–66, 2023.

[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 TNNLS*, vol. 26, no. 7, pp. 1532–1538, 2014.

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

[33] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," *IEEE RAL*, vol. 5, no. 4, pp. 5252–5259, 2020.