Shape Servoing of Deformable Objects using Adaptive Deformation Model Estimation

Vrithik Raj Guthikonda * **Ghananeel Rotithor** **
Ashwin P. Dani ***

* *University of Connecticut, Storrs, CT 06269 USA (e-mail: vrithik.guthikonda@uconn.edu)*
** *University of Connecticut, Storrs, CT 06269 USA (e-mail: ghananeel.rotithor@uconn.edu)*
*** *University of Connecticut, Storrs, CT 06269 USA (e-mail: ashwin.dani@uconn.edu)*

Abstract: In this paper, we propose an adaptive shape servoing method to deform a soft object into a desired 3-D shape. The high dimensional representation and the unknown deformation properties of the soft object pose a challenge to actively manipulate its shape. To address this issue, we develop a method to compute the deformation jacobian matrix in real-time. The jacobian is estimated using a set of basis functions and its corresponding parameters to capture the dynamics of the system and relate the applied input motion to changes in the soft object's shape. An integral concurrent learning (ICL) based adaptive update law is derived using Lyapunov analysis to estimate the deformation parameters and prove its convergence. A physics-based simulation is used to validate the proposed method and controller by performing manipulation tasks with different desired configurations. The performance is compared with a standard gradient update law to demonstrate the accuracy and robustness of our approach.

1. INTRODUCTION

Servoing the shape of the deformable or soft objects to a desired shape by externally providing inputs to servo control the shape is termed as Shape Servoing. Shape sevoing is useful in many applications such as robotic automation of food processing, manipulation of soft robots, and manipulation of soft tissues for medical applications. In shape servoing, typically the shape of the deformable object is measured using an external stationary camera sensor and an external input is provided using a robot to change the shape of the object based on current and desired configuration of the object.

For shape control of the deformable objects, point-based representation is commonly used where the object is represented as a grid of points on the object (Aranda et al. (2020)). The shape servoing task then becomes matching the current features on the deformable object to the desired features. Other representations of the object shape such as the object contour measured in the images have also been studied in literature, for example, Navarro-Alarcon and Liu (2017); Wang et al. (2018); Xu et al. (2022). The contours are represented using a truncated Fourier series in Navarro-Alarcon and Liu (2017), Bezier curve in Xu et al. (2022), B-splines in Li et al. (2005), nonuniform rational B-spline (NURBS) in Wang et al. (2018). These methods require contour estimation in the images which is often a challenging task. Other representations of the flexible object shape are also developed in terms of extended objects using a collection of ellipses or image moment features, see, Yao et al. (2020a,b); Yao and Dani (2017).

A deformation model is required to design a feedback controller for servoing the robot such that the desired shape of the deformable object is achieved. The deformation model provides a relationship between the change in the features as a function of the external velocity input provided at a point or in a small region of the deformable object. Estimating a global deformation model of the entire shape is very challenging due to high dimensional representation of soft objects. In Navarro-Alarcon and Liu (2017), a local linear deformation model is learned by solving a least squares type problem for a given object configuration and deformation model recalibration is achieved by iterative learning at various different configurations. In Wang et al. (2018) the relationship between the control points of Bezier curves and NURBS curve and the velocity is computed by adaptive estimation of the parameters in an adaptive controller loop. A multi-layer neural network is used in Hu et al. (2019) to relate the end-effector movement to the object's deformation with an online learning process to improve model accuracy. In Lagneau et al. (2020), the deformation Jacobian is estimated based on a least-squares minimization, which is further updated based on a user specified confidence threshold. Estimating the deformation Jacobian is a common approach to address this problem however, the control of such systems is difficult due to the limited number of inputs used to control the high dimensional object Yu et al. (2022). In Berenson (2013) a diminishing rigidity-based method is used to compute the Jacobian matrix, which is used to drive the deformable object to a desired configuration.

⋆ This work was supported in part by a Space Technology Research Institutes grant (number 80NSSC19K1076) from NASA Space Technology Research Grants Program and in part by NSF grant no. SMA-2134367.

Manuscript 2085 submitted to IFAC World Congress 2023.
Received November 11, 2022.

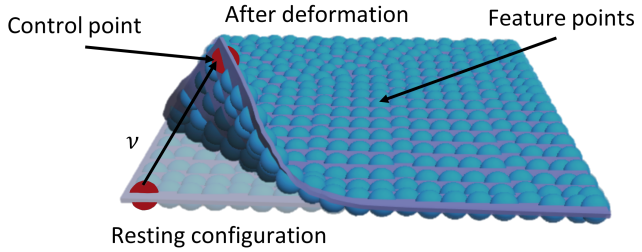


Fig. 1. Shape servoing setup.

A tangent space mapping algorithm is presented in Tang et al. (2016) for the manipulation of deformable objects.

In this paper, a point-based representation of the deformable object such as a cloth is used. It is assumed that the 3D locations of the points on the object are measured by a camera sensor. The Jacobian relationship between the velocity provided to the object and the object deformation measured in terms of the motion of the points is approximated using a Fourier series-based regression Romeres et al. (2020). Since the parameters of the Jacobian are unknown, an adaptive controller is developed, which estimates the parameters of the Jacobian along with regulating the positions of the points to their desired positions. The adaptive law uses integral concurrent learning (ICL) proposed in Parikh et al. (2019), which guarantees the estimation of the parameters along with the regulation of the shape estimation error. The local exponential stability of the adaptive controller can be concluded which provides robustness properties to the controller against the uncertainties in the modeling. The proposed controller is implemented in a simulation task of regulating the shape of a cloth from its current shape to a desired shape. The simulation results of ICL-based adaptive shape regulation controller show good tracking performance against the standard gradient-based adaptive law. The contributions of this paper are as follows:

(1) A new adaptive deformation model approximation is proposed to compute the deformation matrix in real-time without requiring a prior knowledge about the model.
(2) An ICL-based adaptive update law is designed to eliminate the need of an estimator for the state derivative while also improving the model estimation accuracy by collecting data during manipulation.
(3) The ICL-based update law is used to prove parameter convergence as well as prove stability of the system.

The rest of the paper is structured as follows. The problem formulation and shape representation are discussed in Section 2. The system dynamics, regulation and parameter estimation errors, control objective and control design is stated in Section 3. The Lyapunov stability proofs of the controller is given in Section 4, followed by a validation of the controller in Section 5 which shows the validation results of the proposed controller using a simulator platform.

2. SHAPE REPRESENTATION AND PROBLEM DEFINITION

Let us define a deformable object such as a cloth using feature points, which are used to describe the shape and position of the cloth in 3D; and control points, which are used to control the cloth and carry out the regulation task. As shown in Fig. 1, a resting configuration represents the initial state of the soft object before any input is applied. In this paper, the object deformation is defined as the change in feature points on the soft object as a result of an applied velocity input to the control point. A visual illustration of the model is shown in Fig. 1 and a block diagram of the shape servoing system is shown in Fig. 2.

Assumption 1. *The control points are selected such that the desired configuration is reachable.*

3. SHAPE SERVOING CONTROL DEVELOPMENT

3.1 System Dynamics

Consider the following system representing the evolution of the points on the deformable object as a function of the velocity applied to a point on the object. The system model can be written as

$$\dot{s} = J(s)v \tag{1}$$

where $s(t) = [s_1^T, ..., s_n^T]^T \in \mathbb{R}^{3n}$ denotes the set of features used to describe the deformable object with $s_i(t) = [x_i, y_i, z_i]^T$ denoting the i^{th} feature point's position with respect to a fixed frame, $v(t) \in \mathbb{R}^m$ represents the input velocity vector, where m denotes the number of DOF. The deformation Jacobian matrix $J(s) \in \mathbb{R}^{3n\times m}$ is expressed by

$$J(s) = [\theta_1 Y_1(s), \theta_2 Y_2(s), \ldots, \theta_{3n} Y_{3n}(s)]^T \tag{2}$$

where $Y_i(s) \in \mathbb{R}^{d\times 1}$ represents a vector of basis functions and $\theta_i \in \mathbb{R}^{m\times d}$ represents its corresponding unknown parameter matrix. Since deriving the Jacobian matrix analytically is challenging, a Fourier series-based regression is proposed to approximate the deformation Jacobian matrix where each basis function vector $Y_i(s)$ is constructed as follows

$$Y_i(s) = \frac{1}{\sqrt{d}}\left[\cos\left(\frac{\omega_1^T s}{\tau}\right), \ldots, \cos\left(\frac{\omega_d^T s}{\tau}\right) \sin\left(\frac{\omega_1^T s}{\tau}\right) \ldots \sin\left(\frac{\omega_d^T s}{\tau}\right)\right]^T \tag{3}$$

the vector w_k denotes a vector of frequencies with $k \in \{1, ..., d\}$. Since, $d \geq 2$ and even, the number of sines and cosines are same and equal to $d/2$, τ is a constant selected based on the size of the basis vector.

Remark 1. *The trade-off between the model accuracy and computational requirements can be adjusted using the parameter d, see, Williams and Rasmussen (2006).*

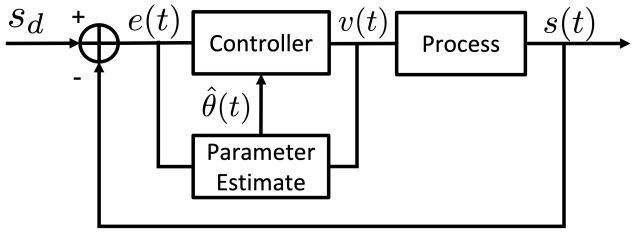


Fig. 2. Block diagram of the adaptive deformation estimation model.

3.2 Controller Objective

The control objective is to regulate the shape of the deformable object given the current measurements of a collection of points $s_i(t)$, $\forall i = 1, \cdots, n$ on the object and the desired collection of points representing the desired shape of the object. Since the Jacobian of the deformation model (1) is unknown, the system dynamics are re-written in the following linearized form

$$\dot{s} = \boldsymbol{Y}^T(s)\boldsymbol{V}^T(v)\bar{\theta} \tag{4}$$

the parameter vector $\bar{\theta} \in \mathbb{R}^{3nmd}$ is constructed by stacking the columns of the unknown parameter matrix θ_i as follows

$$\bar{\theta} = \left[\theta_{11}^T, \theta_{12}^T, \ldots, \theta_{21}^T \ldots \theta_{3nd}^T\right]^T \tag{5}$$

where the vector θ_{ij} represents the i^{th} matrix's j^{th} column. The regressor matrix $\boldsymbol{Y}(s) \in \mathbb{R}^{3nd\times 3n}$ is created such that

$$\boldsymbol{Y}(s) = \begin{bmatrix} Y_1(s) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & Y_{3n}(s) \end{bmatrix} \tag{6}$$

and a matrix $\boldsymbol{V}(v) \in \mathbb{R}^{3nmd\times 3nd}$ is constructed using the control input vector $v(t)$ as follows

$$\boldsymbol{V}(v) = \begin{bmatrix} v & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & v \end{bmatrix} \tag{7}$$

For the control design, the shape regulation error $e(t) \in \mathbb{R}^{3n}$ is defined as

$$e(t) \triangleq s(t) - s_d \tag{8}$$

where $s_d \in \mathbb{R}^{3n}$ are the feature points on the deformable object at the desired configuration and the parameter estimation error $\tilde{\bar{\theta}} \in \mathbb{R}^{3nmd}$ is defined as

$$\tilde{\bar{\theta}} \triangleq \bar{\theta} - \hat{\bar{\theta}}. \tag{9}$$

Taking the time derivative of (8) and using (1) and (4) yields

$$\begin{aligned} \dot{e}(t) &= \dot{s}(t) \\ &= J(s)v = \boldsymbol{Y}^T(s)\boldsymbol{V}^T(v)\bar{\theta} \\ &= \boldsymbol{Y}^T(s)\boldsymbol{V}^T(v)\tilde{\bar{\theta}} + \hat{J}(s)v \end{aligned} \tag{10}$$

where the last equation is obtained by adding and subtracting $\hat{J}(s)v$

3.3 Controller Design

To achieve the desired control objective, the velocity controller is designed as

$$v = -\alpha_v \hat{J}(s)^{\dagger} e \tag{11}$$

where $\hat{J}(s)^{\dagger} = (\hat{J}^T\hat{J})^{-1}\hat{J}^T$ denotes the Moore-Penrose pseudo-inverse of $\hat{J}(s)$ and $\alpha_v \in \mathbb{R}^{m\times m}$ is a constant positive-definite diagonal gain matrix. By substituting the controller into (10), the closed-loop error dynamics can be written as

$$\dot{e} = -\alpha_v \hat{J}(s)\hat{J}(s)^{\dagger} e + \boldsymbol{Y}^T(s)\boldsymbol{V}^T(v)\tilde{\bar{\theta}} \tag{12}$$

The Jacobian $\hat{J}$ is estimated by designing an update law for the parameters $\hat{\bar{\theta}}$ using integral concurrent learning. To simplify the notations let $Q = \boldsymbol{Y}^T\boldsymbol{V}^T \in \mathbb{R}^{3n\times 3nmd}$. The ICL-based parameter update law is designed as

$$\dot{\hat{\bar{\theta}}} = \Gamma_{\theta}^{-1}\{\boldsymbol{V}\boldsymbol{Y}e + K_{cl}\sum_{i=1}^{N}\mathbf{Q}_i^T(s(t_i) - s(t_i - \triangle t) - \mathbf{Q}_i\hat{\bar{\theta}})\} \tag{13}$$

where $\Gamma_{\theta}^{-1} \in \mathbb{R}^{3nmd\times 3nmd}$ and $K_{cl} \in \mathbb{R}^{3nmd\times 3nmd}$ are constant, positive definite diagonal gain matrices and $\Delta t \in \mathbb{R}$ is a positive constant representing the size of the window of integration.

Let us define the integral of Q as

$$\mathbf{Q}(t) = \int_{max\{t-\triangle t, 0\}}^{t} Q(s(\tau), v(\tau), \tau)d\tau \tag{14}$$

Taking the integral on both sides of (4) and substituting $Q = \boldsymbol{Y}^T\boldsymbol{V}^T$ the following expression is obtained

$$\int_{t-\triangle t}^{t} \dot{s}(\tau)d\tau = \int_{t-\triangle t}^{t} Q(s, v, \tau)\bar{\theta}d\tau \tag{15}$$

Using (14) and the fundamental theorem of calculus, (15) can be written as

$$s(t) - s(t - \triangle t) = \mathbf{Q}(t)\bar{\theta} \tag{16}$$

Since $\bar{\theta}$ is a constant, it can be taken out of the integral. Substituting the result from (16) into the ICL-based update law in (13) results in

$$\dot{\hat{\bar{\theta}}} = \Gamma_{\theta}^{-1}\{\boldsymbol{V}\boldsymbol{Y}e + K_{cl}\sum_{i=1}^{N}\mathbf{Q}_i^T\mathbf{Q}_i\tilde{\bar{\theta}}\} \tag{17}$$

Using the parameter estimation error (9), the parameter estimation error dynamics can be written as

$$\dot{\tilde{\bar{\theta}}} = -\Gamma_{\theta}^{-1}\{\boldsymbol{V}\boldsymbol{Y}e + K_{cl}\sum_{i=1}^{N}\mathbf{Q}_i^T\mathbf{Q}_i\tilde{\bar{\theta}}\}. \tag{18}$$

To implement the update law in (17), data collected from the system is stored in a history stack, denoted by $\{s(t_j), s(t_j - \Delta t), \mathbf{Q}_i(t_j)\}_{j=1}^{N}$, which is recorded at the increasing time sequence $\{t_j\}_{j=1}^{N}$. To study the stability of the controller (11) along with the parameter update law (17), consider the combined error vector $\zeta = [e^T \quad \tilde{\bar{\theta}}^T]^T \in \mathbb{R}^{3n(1+md)}$. The error dynamics then can be written using (12) and (18) as

$$\dot{\zeta}(t) = \begin{bmatrix} -\alpha_v\hat{J}\hat{J}^{\dagger} & \boldsymbol{Y}^T\boldsymbol{V}^T \\ -\Gamma_{\theta}^{-1}\boldsymbol{V}\boldsymbol{Y} & -\Gamma_{\theta}^{-1}K_{cl}\sum_{i=1}^{N}\mathbf{Q}_i^T\mathbf{Q}_i \end{bmatrix}\zeta(t). \tag{19}$$

4. STABILITY ANALYSIS

To facilitate the analysis of the controller and parameter estimation law, the following assumption about the history stack is made.

Assumption 2. *The system in (1) is sufficiently exciting over a finite duration of time, which implies that* $\exists\delta > 0$, $\exists T > \Delta t : \forall t \geq T$, $\lambda_{\min}\left\{\sum_{i=1}^{N}\mathbf{Q}_i^T\mathbf{Q}_i\right\} > \delta$, *where* $\lambda_{\min}$ *denotes the minimum eigenvalue of the matrix.*

Remark 2. *Due to the high dimension of* $\hat{\bar{\theta}}$, *the history stack is recorded up to time* $T > 0$, *such that Assumption 2 is satisfied. After time* T *the history stack is either not changed or updated sequentially because updating the*

history stack based on a singular value maximization is computationally expensive.

Remark 3. *Assumption 2 is weaker than the typical persistence of excitation condition required in traditional adaptive control, see, Parikh et al. (2019).*

Consider following candidate Lyapunov function $V : \mathbb{R}^{3n} \times \mathbb{R}^{3nmd} \to \mathbb{R}_+$

$$V\left(e, \tilde{\hat{\theta}}\right) = \frac{1}{2} e^T e + \frac{1}{2} \tilde{\hat{\theta}}^T \Gamma_\theta \tilde{\hat{\theta}} = \zeta^T P \zeta \tag{20}$$

where $P = \text{blkdiag}\, \frac{1}{2} \{\mathbf{I}_{3n}, \Gamma_\theta\}$. The candidate Lyapunov function can be bounded as

$$\underline{c} \|\zeta\|^2 \leq V\left(e, \tilde{\hat{\theta}}\right) \leq \bar{c} \|\zeta\|^2 \tag{21}$$

where $\bar{c} = \lambda_{\max}\{P\} = \frac{1}{2}\max\{1, \lambda_{\max}(\Gamma_\theta)\}$ and $\underline{c} = \lambda_{\min}\{P\} = \frac{1}{2}\min\{1, \lambda_{\min}(\Gamma_\theta)\}$. Also define the ϵ-open ball of dimension $3n$ around the origin as $\mathcal{B}_{3n}(0, \epsilon) = \left\{x \in \mathbb{R}^{3n} \middle| \|x\| < \epsilon\right\}$.

Assumption 3. *The matrix $\hat{J}$ is full column rank that is* $\text{rank}\left(\hat{J}\right) = m$ *and for an appropriate $\epsilon > 0$ if $e \in \mathcal{B}_{3n}(0, \epsilon)$, the feature error $e \notin \text{Null}\left(\hat{J}^T\right)$.*

Remark 4. *Assumption 3 requires the construction of $\hat{J}$ using suitable basis functions and parameter values $\hat{\tilde{\theta}}$ so that the full column rank condition is satisfied. The second condition requires that the feature motion is realizable and no local minima should exist in the local region defined by $\mathcal{B}_{3n}(0, \epsilon)$. This condition can be achieved by shrinking ϵ appropriately as outlined in Chaumette and Hutchinson (2006) and Chaumette (1998).*

Taking the time derivative of (20) and substituting (10) and (17) yields

$$\begin{aligned}
\dot{V} &= e^T \dot{e} + \tilde{\hat{\theta}}^T \Gamma_\theta \dot{\tilde{\hat{\theta}}} \\
&= e^T \hat{J} v + e^T \boldsymbol{Y}^T(s) \boldsymbol{V}^T(v) \tilde{\hat{\theta}} \\
&- \tilde{\hat{\theta}}^T \Gamma_\theta (\Gamma_\theta^{-1} \{\boldsymbol{V}\boldsymbol{Y} e + K_{cl} \sum_{i=1}^{N} \mathbf{Q}_i^T \mathbf{Q}_i \tilde{\hat{\theta}}\}) \\
&= e^T \hat{J} v + e^T \boldsymbol{Y}^T \boldsymbol{V}^T \tilde{\hat{\theta}} - \tilde{\hat{\theta}}^T \boldsymbol{V}\boldsymbol{Y} e - \tilde{\hat{\theta}}^T K_{cl} \sum_{i=1}^{N} \mathbf{Q}_i^T \mathbf{Q}_i \tilde{\hat{\theta}}
\end{aligned} \tag{22}$$

Substituting the controller (11) yields

$$\dot{V} = -\alpha_v e^T \hat{J} \hat{J}^\dagger e - \tilde{\hat{\theta}}^T K_{cl} \sum_{i=1}^{N} \mathbf{Q}_i^T \mathbf{Q}_i \tilde{\hat{\theta}}. \tag{23}$$

In the subsequent theorems the stability of the controller and parameter estimation scheme are proven when Assumption 2 is not satisfied and satisfied.

Theorem 1. *If Assumption 3 is satisfied then for $t \in [0, T)$, the feature and parameter estimation errors generated by the closed loop dynamics in (19) remain bounded.*

Proof. Since Assumption 2 is not satisfied $\sum_{i=1}^{N} \mathbf{Q}_i^T \mathbf{Q}_i \succeq 0$. Then derivative of the Lyapunov function in (20) can be upper bounded as

$$\dot{V} \leq -\alpha_v e^T \hat{J} \hat{J}^\dagger e. \tag{24}$$

Since $e^T \hat{J} \hat{J}^\dagger e \geq 0$, it can be concluded that $\|\zeta(t)\| \leq \sqrt{\frac{\bar{c}}{\underline{c}}} \|\zeta(0)\|$ using Theorem 8.4 of Khalil (2002). □

For the next theorem which analyzes the stability of the error dynamics when the history stack is full, define the set $\mathcal{D} \triangleq \mathcal{B}_{3n(1+md)}(0, r)$ where $r = (\underline{c}\epsilon/\bar{c})$. From the definition of the set $\mathcal{D}$ it can be concluded that if $\zeta(T) \in \mathcal{D}$ implying that $e(T) \in \mathcal{B}_{3n}(0, \epsilon)$

Theorem 2. *If Assumptions 2 and 3 are satisfied, then for $\zeta(T) \in \mathcal{D}$, the system in (19) is exponentially stable.*

Proof. If Assumption 2 is satisfied, then $\lambda_{\min}\left\{\sum_{i=1}^{N} \mathbf{Q}_i^T \mathbf{Q}_i\right\} > \delta$. Let $\lambda_1(t), \cdots, \lambda_m(t)$ be the nonzero eigenvalues of $\hat{J}\hat{J}^\dagger$ in descending order. Since, $e \notin \text{Null}\left(\hat{J}^T\right)$, the derivative of the Lyapunov function in (20) can be upper bounded as

$$\begin{aligned}
\dot{V} &\leq -\lambda_{\min}\{\alpha_v\} \inf_{t \geq 0} \lambda_m \|e\|^2 - \lambda_{\min}\{K_{cl}\}\, \delta \left\|\tilde{\hat{\theta}}\right\|^2 \\
&\leq -\min\left\{\lambda_{\min}\{\alpha_v\} \inf_{t \geq 0} \lambda_m, \lambda_{\min}\{K_{cl}\}\, \delta\right\} \|\zeta\|^2
\end{aligned} \tag{25}$$

Denote $k_1 = \min\{\lambda_{\min}\{\alpha_v\} \inf_{t \geq 0} \lambda_m, \lambda_{\min}\{K_{cl}\}\, \delta\} > 0$, using the bounds on the Lyapunov function, (25) can be further upper bounded as

$$\dot{V} \leq -\frac{k_1}{\bar{c}} V. \tag{26}$$

Using the Comparison Lemma of Khalil (2002), the solution to the differential inequality in (26) can be obtained as

$$V(e(t), \tilde{\hat{\theta}}(t)) \leq V(e(T), \tilde{\hat{\theta}}(T)) e^{-\frac{k_1}{\bar{c}} t}, \ \forall t \in [T, \infty), \tag{27}$$

which leads to the following bound on the error

$$\|\zeta(t)\| \leq \sqrt{\frac{\bar{c}}{\underline{c}}} \|\zeta(T)\| e^{-\frac{k_1}{2\bar{c}} t}, \ \forall t \in [T, \infty). \tag{28}$$

From (28), $\zeta(t) \in \mathcal{L}_\infty$ and the feature and parameter estimation error is exponentially stable using Theorem 4.10 of Khalil (2002). □

In general $\hat{J}\hat{J}^\dagger \succ 0$ is very hard to achieve when the dimension of $s(t)$ is larger than the dimension of $v(t)$. In this case the error dynamics is not GES. Similar to the stability analysis for the IBVS shown in Chaumette and Hutchinson (2006) a local exponential stability of the error dynamics can be concluded.

5. SIMULATION RESULTS

5.1 Simulation Setup

A particle based simulation library from NVIDIA Flex is used to evaluate the proposed method on a Lenovo Intel-i7 laptop with Intel(R) Iris(R) Xe Graphics and 16-GB RAM. A cloth of size 30 x 30 and mass 1 is chosen to model a deformable 3D object as shown in Fig. 3(a). A total of $n = 17$ points are sampled from the cloth to represent its 3D position and configuration in the world frame. To manipulate the shape of the soft object, we chose one point on the corner of the cloth to apply the computed velocity inputs. Flex allows control over each point's position as well as its linear velocities. The vector s_d is collected by storing the positions of each point at a reachable desired configuration. Each iteration, the control input is computed and applied to the control point to drive the cloth to the desired configuration shown in Fig. 3(c).

Manuscript 2085 submitted to IFAC World Congress 2023.
Received November 11, 2022.

5.2 Simulation 1

The parameters used in the simulation are as follows, $m = 3, d = 20, \alpha_v = 0.7\,\mathbf{I}_3, \Gamma_\theta^{-1} = 0.8\,\mathbf{I}_{3nmd}, K_{cl} = 1.5\,\mathbf{I}_{3nmd}, \Delta t = 0.16$ s. The input vector is initialized as $v = [1, 1, 1]^T$ and θ is initialized using a random vector uniformly distributed from $[-1, 1]$. The weights w_k, where $k \in 1, \ldots, d$, are initialized from a normal distribution with zero mean and unit variance. The size of the history stack is computed using the inequality $N \geq \frac{d*3*n*m}{3*n}$ and is selected as $N = 70$. For this simulation, the history stack is filled once according to Assumption 2. The simulator parameters used are $numIterations = 2$, $numSubsteps = 2$. The results from the simulation are summarized in Fig. 4(a) - 4(c) and visually in Fig. 3(a) - 3(c). Fig. 4(a) compares the performance of the ICL based controller with the performance of a gradient-based controller by measuring the error between the object's current shape and the desired shape at each time step. The gradient-based method reaches its lowest error around 25 seconds however, this is caused by an overshoot, resulting in the cloth to eventually converge to a steady state error of 1.3 units. On the other hand, the ICL-based adaptive update law, utilizes the history stack for better adaption of the parameters, which allows it to converge to a smaller steady state error of 0.28 units. Fig. 4(b) shows the computed linear velocities at each time step to drive the cloth to the desired configuration using the gradient-based controller. The volatility in the velocity is caused due to the steady state error of 1.3 units as well as simulation limitations. Note that since this algorithm is implemented on a physics-based simulation, the control point has to over come the simulation gravity as well as bear the weight of other points as it is reaching its desired configuration. In theory, the velocity would go to 0 as the system converges since the end-effector is assumed to be able to overcome the weight of the cloth and maintain a specific pose. Fig. 4(c) shows the computed velocities using the ICL approach and as expected, the velocities are significantly less volatile and converge close to zero due to the small steady state error of 0.28 units.

5.3 Simulation 2

For this simulation, the history stack is updated sequentially as long as Assumption 2 is valid. The ICL-based adaptive controller is implemented and compared with the standard gradient-based adaptive controller for a different desired configuration. The parameters used are $m = 3$, $d = 40$, $\alpha_v = 0.7\,\mathbf{I}_3$, $\Gamma^{-1} = 0.8\,\mathbf{I}_{3nmd}$, $K_{cl} = 1.875\,\mathbf{I}_{3nmd}$ and the input vector is initialized as $v = [0.1, 0.1, 0.1]^T$. The simulator parameters used are $numIterations = 10$, $numSubsteps = 5$. Mostly all the system parameters remained the same with the exception of K_{cl} and d which affect the influence of the history stack and improve the approximation of the model. The results as seen from Fig. 5(a) show a clear advantage of using the ICL-based adaptive update law compared to the standard gradient-based method. The gradient-based controller converges at an error of around 12.5 units whereas the proposed method converges to 0.5 units. The quality of the points stored in the history stack determine the rate of convergence. Unlike simulation 1, where the history stack only contained the first N data points, simulation 2 builds on this approach by sequentially replacing the old points with new incoming points. This allows the stack to evolve and adapt to unexpected disturbances that may be exhibited at a time after the stack is full. The velocity graph in Fig. 5(b) approaches zero as the error reaches a steady state value, however, since the error did not converge to zero, the input continues to compensate for the steady state error. The velocity graph for the ICL-based adaptive update law in Fig. 5(c) shows that the velocity is adjusted based on the evolving stack such that the velocity converges to zero along with the error.

6. CONCLUSION

In this paper, an adaptive controller is designed for controlling the shape of the deformable object. The unknown Jacobian relationship between the external velocity applied to the deformable object and change in positions of points, is approximated using a Fourier series-based regression. An ICL-based parameter update law is designed to estimate the parameters and regulate the shape of the object to a desired shape. Lyapunov analysis is performed to prove stability of the system and prove parameter convergence. The proposed method is implemented using NVIDIA Flex simulator for displacing a cloth to a desired configuration from its current configuration. A comparison between a gradient parameter update law and ICL-based law is provided, which shows better convergence of the shape regulation error. Future work will include performing the deformation task using a robot.

REFERENCES

Aranda, M., Ramon, J.A.C., Mezouar, Y., Bartoli, A., and Özgür, E. (2020). Monocular visual shape tracking and servoing for isometrically deforming objects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 7542–7549.

Berenson, D. (2013). Manipulation of deformable objects without modeling and simulating deformation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4525–4532.

Chaumette, F. (1998). Potential problems of stability and convergence in image-based and position-based visual servoing. In *The Confluence of Vision and Control*, 66–78. Springer.

Chaumette, F. and Hutchinson, S. (2006). Visual servo control. I. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4), 82–90.

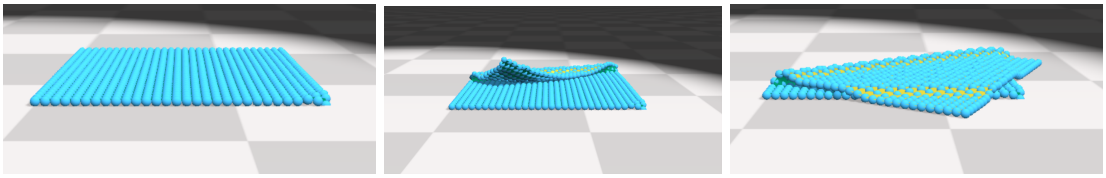
Hu, Z., Han, T., Sun, P., Pan, J., and Manocha, D. (2019). 3-d deformable object manipulation using deep neural networks. *IEEE Robotics and Automation Letters*, 4(4), 4255–4261.

Khalil, H.K. (2002). *Nonlinear Systems*. Prentice Hall, third edition.

Lagneau, R., Krupa, A., and Marchal, M. (2020). Active deformation through visual servoing of soft objects. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 8978–8984.

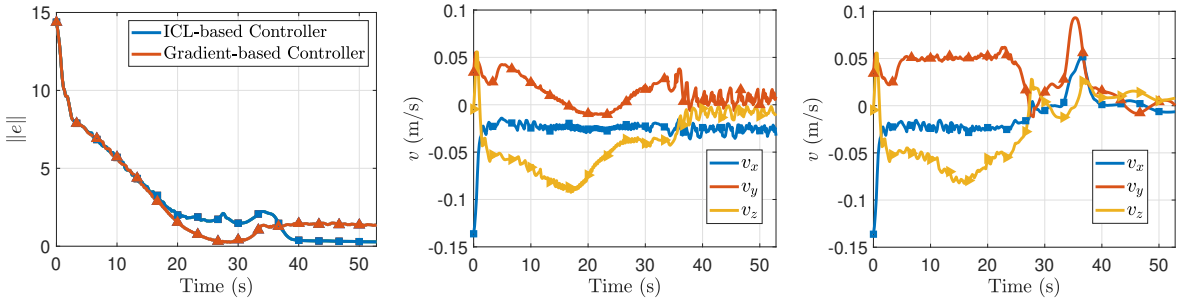
Li, P., Chaumette, F., and Tahri, O. (2005). A shape tracking algorithm for visual servoing. In *IEEE International Conference on Robotics and Automation*, 2847–2852.

Manuscript 2085 submitted to IFAC World Congress 2023.
Received November 11, 2022.



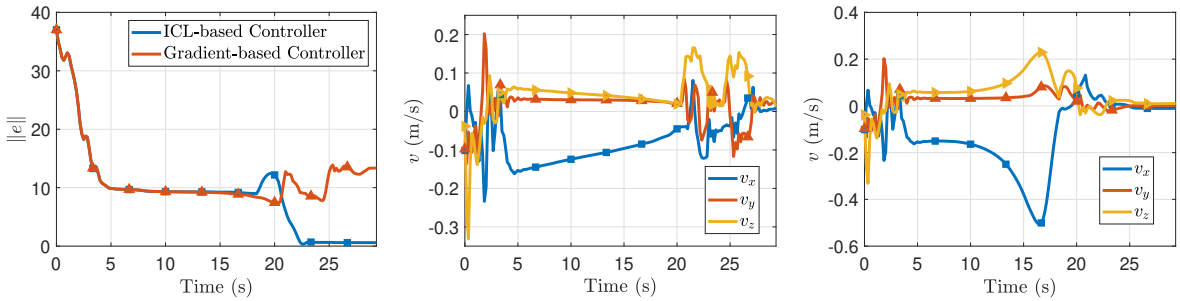
(a) (b) (c)

Fig. 3. Simulation results showing (a) Initial Configuration of the cloth. (b) Intermediate configuration. (c) Desired configuration of the cloth.



(a) (b) (c)

Fig. 4. Results for Simulation 1 (a) Comparison of the shape error $\|e\|$ between gradient-based adaptive controller and the proposed ICL-based adaptive controller. (b) Evolution of the velocity for gradient-based adaptive controller. (c) Evolution of the velocity for ICL-based adaptive controller.



(a) (b) (c)

Fig. 5. Results for Simulation 2 (a) Comparison of the shape error $\|e\|$ between gradient-based adaptive controller and the proposed ICL-based adaptive controller with cyclic update. (b) Evolution of the velocity for gradient-based adaptive controller. (c) Evolution of the velocity for ICL-based adaptive controller.

Navarro-Alarcon, D. and Liu, Y.H. (2017). Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2-D image contours. *IEEE Transactions on Robotics*, 34(1), 272–279.

Parikh, A., Kamalapurkar, R., and Dixon, W.E. (2019). Integral concurrent learning: Adaptive control with parameter convergence using finite excitation. *International Journal of Adaptive Control and Signal Processing*, 33(12), 1775–1787.

Romeres, D., Zorzi, M., Camoriano, R., Traversaro, S., and Chiuso, A. (2020). Derivative-free online learning of inverse dynamics models. *IEEE Transactions on Control Systems Technology*, 28(3), 816–830.

Tang, T., Liu, C., Chen, W., and Tomizuka, M. (2016). Robotic manipulation of deformable objects by tangent space mapping and non-rigid registration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2689–2696.

Wang, H., Yang, B., Wang, J., Liang, X., Chen, W., and Liu, Y.H. (2018). Adaptive visual servoing of contour features. *IEEE/ASME Transactions on Mechatronics*, 23(2), 811–822.

Williams, C.K. and Rasmussen, C.E. (2006). Gaussian processes for machine learning, vol. 2, no. 3.

Xu, F., Zhang, Y., Sun, J., and Wang, H. (2022). Adaptive visual servoing shape control of a soft robot manipulator using bézier curve features. *IEEE/ASME Transactions on Mechatronics*.

Yao, G. and Dani, A. (2017). Image moment-based random hypersurface model for extended object tracking. In *International Conference on Information Fusion*, 1–7.

Yao, G., Saltus, R., and Dani, A. (2020a). Image moment-based extended object tracking for complex motions. *IEEE Sensors Journal*, 20(12), 6560–6572.

Yao, G., Saltus, R., and Dani, A.P. (2020b). Shape estimation for elongated deformable object using b-spline chained multiple random matrices model. *International*

Journal of Intelligent Robotics and Applications, 4(4), 429–440.

Yu, M., Lv, K., Zhong, H., Song, S., and Li, X. (2022). Global model learning for large deformation control of elastic deformable linear objects: An efficient and adaptive approach. *IEEE Transactions on Robotics*.

Manuscript 2085 submitted to IFAC World Congress 2023.
Received November 11, 2022.