AdapJ: An Adaptive Extended Jacobian Controller for Soft Manipulators

Zixi Chen, *Member, IEEE*, Xuyang Ren, Yuya Hamamatsu, *Graduate Student Member, IEEE*, Gastone Ciuti, *Senior Member, IEEE*, Donato Romano, and Cesare Stefanini, *Member, IEEE*

***Abstract*—The nonlinearity and hysteresis of soft robot motions present challenges for control. To solve these issues, the Jacobian controller has been applied to approximate the nonlinear behaviors in a linear format. Accurate controllers like neural networks (NNs) can handle delayed and nonlinear motions, but they require large datasets and exhibit low adaptability. Based on a novel analysis on these controllers, we propose an adaptive extended Jacobian controller, *AdapJ*, for soft manipulators. This controller retains the concise format of the Jacobian controller but introduces independent parameters. Similar to NNs, its initialization and updating mechanism leverages the inverse model without building the corresponding forward model. In the experiments, we first compare the performance of the Jacobian controller, model predictive controller (MPC), NN controller, iterative feedback controller (IFC), and AdapJ in simulation. We further analyze how AdapJ parameters adapt in response to the physical property change. Then, real-world experiments have validated that AdapJ outperforms the NN controller, MPC, and IFC with fewer training samples and adapts robustly to varying conditions, including different control frequencies, material softness, and external disturbances. Future work may include online adjustment of the controller format and adaptability validation in more scenarios.**

***Index Terms*—Adaptive control, Jacobian controller, soft robots.**

I. Introduction

LEVERAGING the soft materials and flexible structures, soft robots offer higher degrees of freedom than traditional rigid robots. As a result, soft robot manipulators have been utilized in many applications, such as minimally invasive surgery [1] and exploration in confined spaces [2]. Their inherent compliance also reduces the risk of damaging objects, the surrounding environment, or human collaborators during motion and manipulation. For example, soft manipulators can be applied for human–robot interaction and demonstrations [3], and some can even be deployed as assistive robots for elderly individuals during showering activities [4]. Complex soft manipulators, like modular soft robots, can perform challenging tasks like shape control [5] and high-level manipulation [6] utilizing dedicated control strategies. Overall, soft manipulators represent a critical component of soft robotics, offering distinct advantages such as safety and adaptability, and have garnered significant attention from researchers across a wide range of disciplines.

It is vital to propose suitable controllers for soft manipulators to achieve challenging tasks. These controllers should be able to cope with the nonlinearity and hysteresis of soft robotic systems, and plenty of controllers have been introduced to address these challenges [10]. For instance, Jacobian approaches are intuitive and straightforward to implement. Although the linear and transient nature does not inherently accommodate the nonlinearity and hysteresis of soft robot motions, high-frequency online updates of the Jacobian matrix are applied to mitigate this issue. Controllers based on physical models like Piecewise Constant Curvature (PCC) [11] and Cosserat rods [12] are explainable, but they are sophisticated and rely on theoretical assumptions that may not hold in practice. Discrepancies between the actual behavior of soft robots and their idealized models caused by complex material properties and unpredictable deformations can lead to control inaccuracies. Statistical controllers, including the Gaussian mixture model [13] and the Gaussian process regression [14], rely on statistical models, which require a moderate amount of training data and yield moderate performance. Neural networks (NNs), especially recurrent neural networks (RNNs) [15], have gained popularity in soft robot control due

Received 29 May 2025; revised 27 July 2025; accepted 17 September 2025. Recommended by Technical Editor W. He and Senior Editor K. Oldham. This work was supported in part by the European Union by the Next Generation EU project under Grant ECS00000017 "Ecosistema dell'Innovazione" Tuscany Health Ecosystem (THE, PNRR, Spoke 9: Robotics and Automation for Health) and in part by the Italian Space Agency (ASI) under Grant DC-DSR-UVS-2022-375 Project "pRomoting pEdogenesis throuGh lunar sOil-terrestriaL organIsms interaction For moon Fertilization – REGOLIFE" [ASI N.: 2024-7-U.0; CUP: J83C24000310005], and BRIEF "Biorobotics Research and Innovation Engineering Facilities" (project identification code IR0000036), project funded under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 3.1 of Italian Ministry of University and Research funded by the European Union – NextGenerationEU. *(Corresponding author: Xuyang Ren.)*

Zixi Chen, Gastone Ciuti, and Donato Romano are with the Biorobotics Institute and the Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, 56127 Pisa, Italy (e-mail: zixi.chen@santannapisa.it; gastone.ciuti@santannapisa.it; donato.romano@santannapisa.it).

Xuyang Ren is with the Multi-scale Medical Robotics Centre and Chow Yuk Ho Technology Centre for Innovative Medicine, Chinese University of Hong Kong, Hong Kong (e-mail: xuyang.ren.cn@gmail.com).

Yuya Hamamatsu is with the Department of computer systems, Tallinn University of Technology, 19086 Tallinn, Estonia (e-mail: yuya.hamamatsu@taltech.ee).

Cesare Stefanini is with the Department of Robotics, Mohamed bin Zayed University of Artificial Intelligence, Masdar City, Abu Dhabi, UAE (e-mail: Cesare.Stefanini@mbzuai.ac.ae).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TMECH.2025.3612503.

Digital Object Identifier 10.1109/TMECH.2025.3612503

to their nonlinear activation functions and effective network structures. Trained NNs can serve as global controllers across the entire workspace; however, their large parameter sets make real-time online updates challenging, limiting their adaptability.

Among all these controllers targeted at soft robotics mentioned above, the Jacobian controller is one of the simplest controllers for soft robotics [2], [10]. Researchers were inspired by the Jacobian controller in rigid robot control and employed it in soft robots. Prior to actuation, the initial Jacobian matrix is required, and Yip and Camarillo in [2] estimate it by actuating each actuator independently by an incremental amount. Then, the actuation is determined by either optimization employing the Jacobian matrix as the forward model [16] or calculation using the inverse Jacobian matrix [17]. Following each actuation step, the Jacobian matrix updates according to the previous robot states and actions, allowing it to adapt to local dynamics. Once the update finishes, the control loop proceeds with the next iteration.

Several modifications have been introduced to employ this controller in real-world applications and achieve better performance. For instance, robot states with a larger update interval are utilized in [18] for the Jacobian matrix update. In this case, the disturbance can be rejected, and such a controller can be applied for complex cardiac ablation tasks. In [19], the Jacobian update process is smoothed by averaging the previous Jacobian matrix with the newly estimated one, resulting in more stable motion control. To perform segmented soft robot control, Liu et al. [20] bridge the gap between robot end position and actuation via segment orientations instead of directly mapping between end position and actuation. In this way, complicated nonlinear mapping functions are decomposed to fit the Jacobian matrix.

While the adaptations above have extended the Jacobian controller's capabilities to support complex tasks, they also reflect a fundamental limitation: soft robots often deviate from the original Jacobian model and its underlying assumptions. Specifically, the Jacobian approaches suppose that the robot state change $\triangle s$ is linear to the actuation change $\triangle a$, expressed as $\triangle s = J\triangle a$. However, considering the time delay in soft robot motion, the robot state may continue to change ($\triangle s \neq 0$) when the actuation remains unchanged ($\triangle a = 0$). Due to the unsatisfactory performance caused by misalignment, the Jacobian matrix is primarily applied for motion description in principle, even with the adaptations mentioned above, and other data-driven controllers (NN in most cases) replace Jacobian controllers in practice. For instance, the Jacobian matrix is referenced in [21] for the theoretical soft robot modeling, but a NN is applied for control to replace the Jacobian approach.

Compared to straightforward yet limited Jacobian controllers, NNs have emerged as the most widely adopted approach in soft manipulator control [10]. Their suitability stems from the inherent nonlinearity and delay of soft robotic motion, which aligns well with the nonlinear activation functions and complex architectures of NNs. Moreover, the rich network structure and the large number of independent trainable parameters are suitable to cope with the intricate soft robot motion. Basic NNs like multilayer perceptron (MLP) can be applied to handle the delayed soft robot motion [22]. Furthermore, RNNs are composed of recurrent structures developed specifically for sequential problems, such as hysteresis [23] and segment sequence [24] in soft robot control.

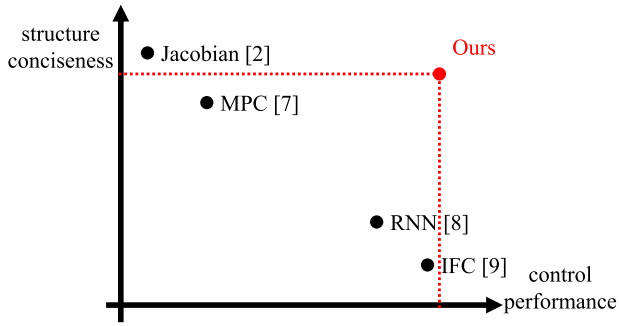


Fig. 1. Qualitative comparison of soft robot controllers. The proposed controller maintains a low structural complexity and computational cost, only slightly greater than that of the Jacobian controller, while outperforming the Jacobian controller [2], MPC [7], RNN controller [8], and IFC [9] in terms of control performance considering accuracy and adaptability.

In this work, based on the novel analysis of the compact Jacobian controller and accurate RNN controller, we introduce an adaptive extended Jacobian controller, *AdapJ*, specifically for soft manipulators. We retain the compact structure of the Jacobian controller while relaxing the parameter constraints. Also, we update the controller directly instead of the forward model and detail the matrix initialization and update approach learned from NN and model predictive controller (MPC). As illustrated in Fig. 1, AdapJ requires significantly less training data than most existing soft manipulator controllers owing to its concise architecture. Despite its simplicity, AdapJ demonstrates superior performance in the simulation and real-world experiments, outperforming classical and state-of-the-art (SOTA) controllers such as the Jacobian controller [2], MPC [7], RNN-based controllers [8], and the iterative feedback controller (IFC) [9]. In addition, its capacity for online updates endows it with adaptability to different frequencies, physical parameters, and even external disturbances.

The contributions of this article are as follows.

1) Based on the insightful analysis of the Jacobian controller and RNN controller, we propose an adaptive extended Jacobian controller for soft manipulators. Inspired by RNN, it relaxes the parameter constraints inherent in traditional Jacobian methods to better address hysteresis.
2) We leverage the inverse dynamics, rather than the corresponding forward model in the Jacobian controller, for initialization and online updating. Inspired by RNN and MPC, we apply motor babbling and the Gauss–Newton algorithm for initialization and online updating.
3) We conduct simulation experiments to compare AdapJ with existing controllers such as the Jacobian controller, MPC, RNN-based controllers, and IFC. Also, we analyze the controller parameter adaptation in response to variations in robot stiffness and damping properties. Real-world experiments on a soft robot manipulator further demonstrate the superior performance of our controller over IFC, MPC, and RNN baselines. We also validate

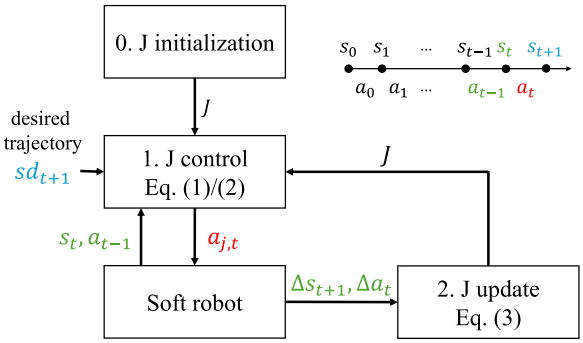


Fig. 2. Diagram of the standard Jacobian controller. Following initialization of the Jacobian matrix, the Jacobian controller computes the actuation $a_{j,t}$ (red) based on the previous robot state, actuation s_t, a_{t-1} (green), and desired robot state sd_{t+1} (blue). Then the Jacobian matrix updates according to the difference in robot state and actuation $\triangle s_{t+1}, \triangle a_t$ (green).

the controller's adaptability under different frequencies, physical properties, and external disturbances.

The rest of this article is organized as follows: Section II introduces AdapJ, beginning with a novel analysis of the limitations and advantages of the Jacobian and RNN controllers. It also details the initialization and updating principles of AdapJ. Section III describes the simulation and real-world experimental setup, including robots, working spaces, and devices. Section IV compares our controller with some existing controllers and analyzes the controller parameter adaptation. Then, we demonstrate the effectiveness of our controller through real-world experiments. The adaptability is also validated across various scenarios. Finally, Section V concludes this article.

II. Controller Design

In this section, we begin by analyzing the original Jacobian controller and RNN controller in Section II-A. Based on the insightful analysis, we propose our adaptive extended Jacobian controller for soft manipulators in Section II-B, followed by the proposed initialization and update strategy in Section II-C.

A. Existing Controller Analysis

1) Jacobian Controller: The Jacobian controller in [2] is shown in Fig. 2. First, the Jacobian controller is initialized by moving each actuator independently by an incremental amount while measuring the motion. In addition to this data collection method, some researchers also initialize the Jacobian matrix via a physical kinematics model [19]. After initializing the Jacobian matrix $J \in R^{d_s \times d_a}$, where d_s and d_a denote the dimensions of state and actuation, the following optimal control strategy is implemented. Combining different strategies in [2], [16], [25], the actuation $a_{j,t}$ is determined by

$$\begin{aligned} \min_{a_{j,t}} \ & \alpha_1 \|a_{j,t}\|_2 + \alpha_2 \|\triangle a_{j,t}\|_2 \\ \text{s.t.} \ & \triangle sd_{t+1} = J \triangle a_{j,t} \\ & \triangle sd_{t+1} = sd_{t+1} - p_t \\ & \triangle a_{j,t} = a_{j,t} - a_{t-1} \end{aligned} \tag{1}$$

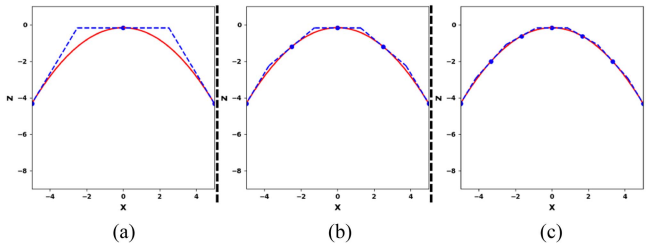


Fig. 3. Diagram of the Jacobian principle. The nonlinear function $z = -(x^2+1)/6$ is approximated by (a) 3, (b) 5, and (c) 7 linear functions. The blue dotted lines represent the linear approximation functions, and the red lines represent the nonlinear function.

where $s_t \in R^{d_s}, a_{t-1} \in R^{d_a}$ represent previous robot state and actuation at tth and $t-1$th timestep, and the robot state s is robot end position in most works [2] and bending angle in some others [7]. sd_{t+1} denotes the desired robot state from the desired trajectory. α_* represent cost weights, where $\alpha_1 = 1, \alpha_2 = 0$ in [2] and $\alpha_1 = 0, \alpha_2 = 1$ in [25].

In addition to this optimal control strategy, the action can also be decided by using the inverse Jacobian matrix $J^{-1} \in R^{d_a \times d_s}$ [17], which can be denoted as

$$\begin{aligned} a_{j,t} &= J^{-1}(sd_{t+1} - s_t) + a_{t-1} \\ &= J^{-1} sd_{t+1} + (-J^{-1}) s_t + I a_{t-1}. \end{aligned} \tag{2}$$

After the action decision, the Jacobian matrix will be updated according to the feedback. The strategy combined from [2], [16], [26] can be denoted as

$$\begin{aligned} \min_{\triangle J} \ & \beta_1 \|\triangle s_t - (J + \triangle J)\triangle a_{t-1}\|_2 + \beta_2 \|\triangle J\|_2 \\ \text{s.t.} \ & \triangle s_t = s_t - s_{t-1} \\ & \triangle a_{t-1} = a_{t-1} - a_{t-2} \end{aligned} \tag{3}$$

where $J + \triangle J$ represent the new Jacobian matrix. β_* represent cost weights, where $\beta_1 = 0, \beta_2 = 1$ in [2], $\beta_1 = 1, \beta_2 = 0$ in [16], and $\beta_1 = \beta_2 = 1$ in [26] for robust Jacobian estimation.

In addition to describing the controller, we aim to illustrate the core principle of the Jacobian approach—approximating nonlinear functions using multiple local linearizations—in Fig. 3. Specifically, we approximate the nonlinear function $z = -(x^2+1)/6$ with multiple linear functions at local sampling points. As the number of sampling points increases from Fig. 3(a) to (c), the Jacobian strategy achieves better approximation performance, aligning with the high control frequency (1 kHz) in the classical Jacobian controller work [2].

After introducing the Jacobian controller and its core principle, we analyze its shortcomings. First, the Jacobian assumption $\triangle s = J \triangle a$ does not suit soft robot motion because the robot may move ($\triangle s \neq 0$) when the actuation keeps constant ($\triangle a = 0$), considering the delay of soft manipulator motion. In other words, the change in state is not solely determined by the actuation difference within a single timestep. In addition, there are strong constraints on the partial derivatives in (2), which are

$$\frac{\partial a_{j,t}}{\partial a_{t-1}} = I$$

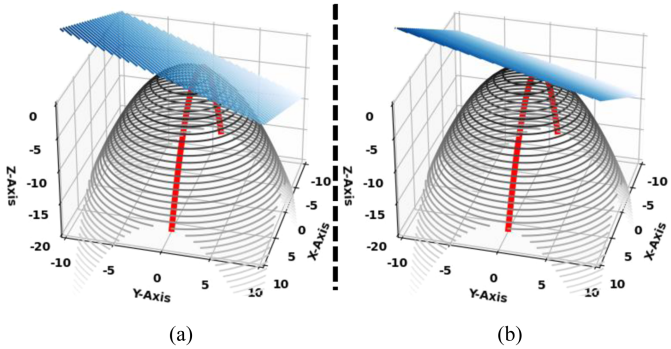


Fig. 4. Diagrams of approximating a multivariable nonlinear function using linear approximation with (a) coupling and (b) independent variables. The grey curved surfaces represent the multivariable nonlinear function $z = -(x^2 + y^2)/6$, the red lines represent the nonlinear function $z = -(x^2 + 1)/6$, and the blue planes represent the linear approximation functions.

$$\frac{\partial a_{j,t}}{\partial sd_{t+1}} = -\frac{\partial a_{j,t}}{\partial s_t} = J^{-1}. \tag{4}$$

The parameter coupling may impair the approximation of soft robot behaviors.

2) RNN Controller: We denote the RNN controller as

$$a_{\text{RNN},t} = f_{\text{RNN}}(sd_{t+1}, s_t \sim s_{t-n+2}, a_{t-1} \sim a_{t-n}) \tag{5}$$

where n denotes the timestep of the recurrent layers. $a_{\text{RNN},t}$ is the actuation determined by the RNN controller and produced by a fully connected layer following the recurrent layers. Unlike the Jacobian controller, the RNN parameters are not subject to strong structural coupling constraints imposed by the Jacobian formulation, as described in (4). Based on the observed differences in parameter coupling and the corresponding impact on control performance, we hypothesize that parameter independence may also influence the effectiveness of soft robot controllers. To validate this, we propose a simple mathematical model.

Given the delayed response in soft robot motion, both modeling and control can be formulated as multivariable nonlinear functions that depend on states and actions over multiple timesteps. We analyze how the parameter independence influences the approximation of multivariable nonlinear functions and furthermore address hysteresis, as shown in Fig. 4. While the effect of sampling density (or control update frequency) on approximation accuracy has been discussed previously, here we focus on comparing approximation quality at a single point. To illustrate this, we approximate the multivariable nonlinear function $z = -(x^2 + y^2)/6$ at the point $(-2, 1, -\frac{5}{6})$ using linear functions with and without parameter coupling. If we couple two variables by imposing the condition $\frac{\partial z}{\partial y} = -\frac{\partial z}{\partial x}$, analogous to the second Jacobian constraint in (4), the linear approximation becomes $z = \frac{1}{2}x - \frac{1}{2}y + \frac{2}{3}$, shown in Fig. 4(a). If we do not impose any coupling between $\frac{\partial z}{\partial y}$ and $\frac{\partial z}{\partial x}$, the linear approximation is $z = \frac{2}{3}x - \frac{1}{3}y + \frac{5}{6}$, as illustrated in Fig. 4(b).

To quantitatively evaluate approximation performance, we compute the deviation between the linear approximation and the true nonlinear surface at points sharing the same x and y coordinates. Specifically, we sample points with $x \in [-2.4, -2.2, -2, -1.8, -1.6]$, $y \in [0.6, 0.8, 1, 0.2, 1.4]$, chosen for their proximity to the approximation point $(-2, 1)$. The average approximation errors for each linearization strategy are presented in Table I. The results indicate that the independent variable case outperforms the coupling cases, demonstrating that independent parameters improve the accuracy of linear approximations for multivariable nonlinear functions and soft robotics modeling and control. Motivated by this insight, we propose the extended adaptive Jacobian controller, which builds upon the original Jacobian controller format but relaxes parameter constraints.

TABLE I
MULTIVARIABLE NONLINEAR FUNCTION APPROXIMATION ERRORS (1E-3)

variable	independent	coupling
error	**26.67**	58.67

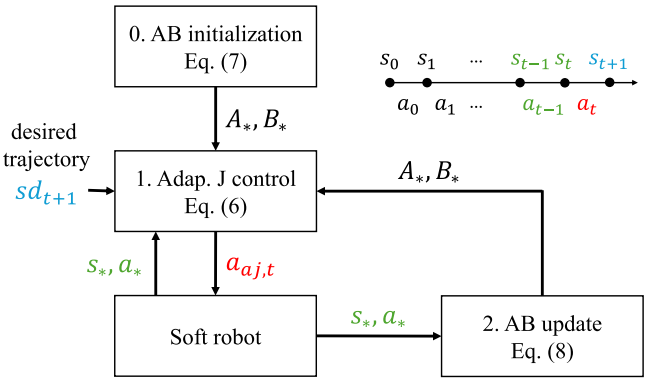


Fig. 5. Diagram of our adaptive extended Jacobian controller. The extended inverse Jacobian matrices A_*, B_* are initialized with motor babbling and batch optimization. Then, the actuation $a_{aj,t}$ (red) is computed based on these matrices, previous robot states, actuations s_*, a_* (green), and the desired robot state sd_{t+1} (blue). After execution, the matrices A_*, B_* are updated, and the control loop proceeds to the next iteration.

B. Adaptive Extended Jacobian Controller-AdapJ

Based on the insights from the existing controller discussion, we propose an adaptive extended Jacobian controller, AdapJ, as shown in Fig. 5. It can be denoted as

$$a_{aj,t} = A_0 sd_{t+1} + A_1 s_t + B_0 a_{t-1} \tag{6}$$

where $A_* \in R^{d_a \times d_s}, B_* \in R^{d_a \times d_a}$ represent independent extended inverse Jacobian matrices and $a_{aj,t}$ represent the tth timestep actuation decided by our controller. Different from the conventional Jacobian controller, the matrices $\frac{\partial a_{aj,t}}{\partial sd_{t+1}}, \frac{\partial a_{aj,t}}{\partial s_t}, \frac{\partial a_{aj,t}}{\partial a_{t-1}}$ are independent and do not coupled following the Jacobian assumption in (4). This adaptation is inspired by the independent parameters in RNN, and initially validated in the multivariable nonlinear function example in Fig. 4. The proposed controller can be seen as an extended Jacobian controller because it will degrade to the inverse Jacobian controller in (2) when $A_0 = -A_1 = J^{-1}, B_0 = \boldsymbol{I}$.

C. Matrix Initialization and Update

Considering the parameter independence, it is impossible to follow the initialization strategy in [2] by actuating each actuator and directly calculating the matrices. Therefore, we collect data from soft robots to initialize extended inverse Jacobian matrices A_*, B_* by utilizing the motor babbling strategy like RNN controllers. Due to the proposed controller conciseness, it requires far fewer samples (only 100 in our cases) than RNN controllers (at least 5000), which means we collect $a_0 \sim a_{99}$ and $s_0 \sim s_{99}$ utilizing motor babbling strategy. The Jacobian control approach in [2] initializes the forward Jacobian matrix. Similarly, MPC in [7] leverages a forward model and optimization for actuation determination. However, we omit the forward model and directly initialize the inverse dynamics. The extended inverse Jacobian matrices A_*, B_* can be initialized as

$$\begin{aligned} &\min_{A_*,B_*} \ \|a_{aj,t} - a_t\| \\ &s.t.\ a_{aj,t} = A_0 s_{t+1} + A_1 s_t + B_0 s_{t-1} \end{aligned} \tag{7}$$

where $t \in [1, 98]$. For instance, when $t = 1$, we estimate $a_{aj,1}$ based on the robot states and actuations s_2, s_1, a_0. When $t = 98$, we estimate $a_{aj,98}$ based on the robot states and actuations s_{99}, s_{98}, a_{97}. We learn from RNN and employ batch optimization to stabilize the initialization process. By updating the matrices online, AdapJ can approximate the real inverse dynamics with the nominal one as an initial guess step by step, endowing it with adaptability to fixed physical property changes such as stiffness and damping changes.

Following the MPC updating principle in [7], the Gauss–Newton method is applied for a short computational time, and the updating strategy can be denoted as

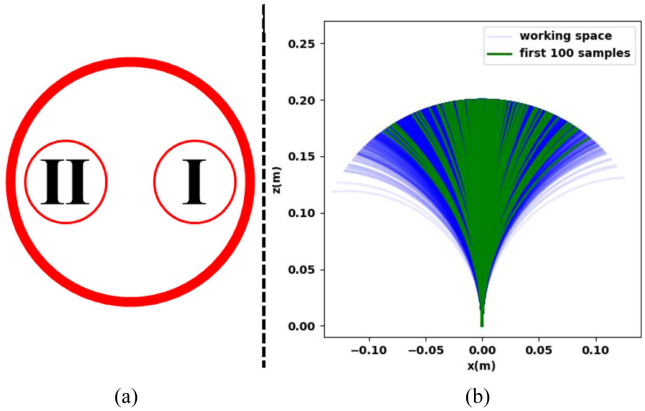
$$\omega = \begin{cases} \omega + \triangle\omega, & \|\triangle\omega\| \le \triangle\omega^{\max} \\ \omega + \frac{\triangle\omega}{\|\triangle\omega\|}\triangle\omega^{\max}, & \|\triangle\omega\| > \triangle\omega^{\max} \end{cases} \tag{8}$$

where

$$\begin{aligned} \triangle\omega &= \rho(\phi\phi^T)^{-1}\phi E \\ \omega &= [A_0\ A_1\ B_0]^T \in R^{(2d_s+d_a)\times d_a} \\ \phi &= [s_{t+1}\ s_t\ a_{t-1}]^T \in R^{(2d_s+d_a)\times 1} \\ E &= [a_t - a_{aj,t}]^T \in R^{1\times d_a} \end{aligned} \tag{9}$$

and $\triangle\omega^{\max}$ denotes the maximum parameter change allowed at each update step for smooth update. The constraint endows AdapJ with robustness to dynamic disturbance, which only affects the manipulator in several timesteps. After a_t is executed and s_{t+1} is measured, the controller can be seen as an online actuation estimator, and we can update matrices A_*, B_* following (8) to fit the current local situation. The model parameter convergence utilizing the Gauss–Newton method for updating has been proven in [7].

In our controller design, nonlinearity is addressed through online updates, as illustrated in Fig. 3, while hysteresis is mitigated by including previous state and actuation as input and relaxing parameter coupling constraints, as shown in Fig. 4. Although AdapJ is fundamentally a linear controller, it effectively approximates the nonlinear and delayed dynamics of soft manipulator behavior as validated in the following experiments. Compared to Jacobian controllers, AdapJ removes parameter coupling to improve multivariable nonlinear function approximation. Compared to RNN controllers, AdapJ employs significantly fewer parameters, enabling online updates and adaptability to fixed physical property changes and dynamic disturbances. Compared to MPC, AdapJ directly approximates the inverse dynamics instead of the forward modeling, eliminating the necessity for actuation decision through time-consuming optimization in [7]. Collectively, these design improvements result in an adaptive and precise controller well-suited for soft manipulator applications.



(a) (b)

Fig. 6. (a) Two actuation units are placed symmetrically in the robot. (b) The working space in the simulation. The first 100 samples (green) are applied for the initialization of the matrices A_*, B_*. The 5000 samples (light blue) collected by the motor babbling strategy basically show the robot working space and are applied to the other controller initialization.

III. Experiment Setup

In this section, we introduce the experiment setup, devices, and the corresponding working space in simulation and the real world in Sections III-A and III-B, respectively.

A. Simulation Experimental Setup

In simulation, we aim to compare controllers preliminarily and perform an intuitive parameter analysis. Therefore, we built a simple soft robot finger based on the pseudorigid model in [11], whose length is 200 mm. The bending stiffness is 0.4 Nm^2 and the damping coefficient is 1 Nms. As shown in Fig. 6(a), two actuation units are placed symmetrically in the robot and actuated by $a_h \in R$, where

$$\begin{aligned} a_I &= \max\{0, a_h\} \\ a_{II} &= \max\{0, -a_h\}. \end{aligned} \tag{10}$$

We randomly actuate the robot and collect 5000 samples, shown in Fig. 6(b), and only the first 100 samples are applied for our controller initialization. The x position of the finger end is applied as the robot state $s \in R$, whose range is $[-131.9,$

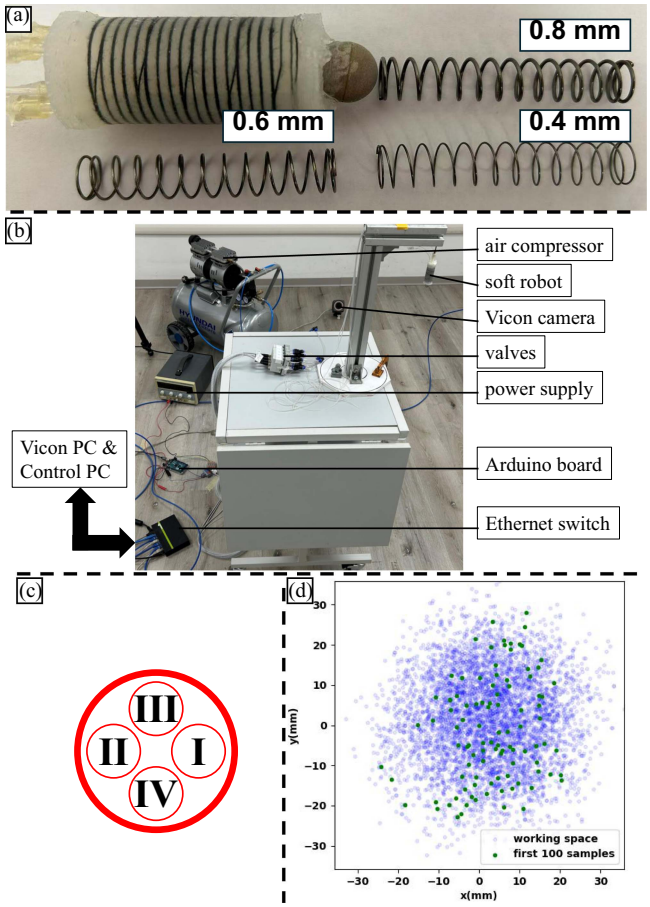


Fig. 7. (a) Pneumatic soft manipulator in real experiments. One manipulator is composed of four chambers, and one springer is inserted inside the center for stiffness adjustment. Three kinds of springs (wire diameter 0.4, 0.6, and 0.8 mm) are applied in our experiments. (b) The experimental setup. The manipulator is actuated by an air compressor and controlled by valves and an Arduino board. The optical tracking system collects the motion. (c) Robot actuation diagram. The actuation values in four actuation units $a_I, a_{II}, a_{III}, a_{IV}$ are controlled by the actuation $a = [a_h, a_v]$. (d) The working space in the real experiment. The first 100 samples (green) are applied for A_*, B_* initialization, and 5000 samples (blue) are collected for the others.

131.9] mm. The actuation a_h and the state s are rescaled to $[-1, 1]$ for RNN.

B. Real Experimental Setup

To validate our controller in the real world, we manufacture a pneumatic silicone soft manipulator shown in Fig. 7(a). The robot is made of Ecoflex 00-30 (Smooth-On, Macungie, PA), and the length of each module is about 45 mm. There are four symmetrical chambers placed at an interval angle of 90° along the circumferential direction inside the manipulator as shown in Fig. 7(c). The cotton line wraps the robot to constrain the radial expansion. Springs (wire diameter 0.6 mm) are put inside the robot center to adjust the stiffness, and we name it Robot 1. The robot structure design can be found in [27]. To validate the adaptability of our controller, we change the inner spring (named Robot 2 for the wire diameter of 0.4 mm and Robot 3 for the wire diameter of 0.8 mm) to change the robot stiffness. The state $s \in R^2$ is the robot end position and actuation $a = [a_h, a_v] \in R^2$ actuates four chambers following:

$$\begin{aligned} a_I &= \max\{0, a_h\} \\ a_{II} &= \max\{0, -a_h\} \\ a_{III} &= \max\{0, a_v\} \\ a_{IV} &= \max\{0, -a_v\}. \end{aligned} \tag{11}$$

Similar to simulation, they are rescaled to $[-1, 1]$ for RNN.

The manipulator is actuated in the setup shown in Fig. 7(b). An optical tracker ball is fixed at the end of the robot. Six optical tracking cameras (VICON Bonita) and one data collection computer named Vicon PC are used to track the robot motion. An air compressor connects chambers, whose pressures are controlled by valves (Camozzi K8P-0-E522-0). We control chamber pressure via a control PC (Ubuntu 20.04, CPU i5-12500H, and RTX 3050) and an Arduino MEGA board. Two computers and six cameras communicate through an Ethernet switch. Pytorch is applied for optimization and NN training. The data collection and control frequency is 10 Hz without a special statement. Similar to the simulation, we collect 5000 samples to train the RNN controller and show the working space of Robot 1 in Fig. 7(d). The working space area is about 71.69 mm × 71.69 mm. The first 100 samples are employed for A_*, B_* matrices initialization.

IV. Experiment Results

In this section, we first compare the original Jacobian controller [2], MPC [7], RNN [8], IFC [9], and the proposed controller in simulation in Section IV-A. The parameter adaptation in response to the robot's physical properties is also discussed. Then we compare the proposed controller, MPC, IFC, and the RNN controller in the real-world experiments in Section IV-B. We validate the adaptability of our proposed controller in different frequencies, physical properties, and external disturbances.

A. Preliminary Controller Analysis in Simulation

In this section, we aim to compare the original Jacobian controller in (2), the MPC in [7], the RNN controller in (5) and [8], IFC in [9], and our controller in (6). IFC is based on RNN and leverages the discrepancy between the target and actual states from previous steps for online compensation. In addition, we train one RNN with only the first 100 samples and name this controller "RNN_100" to validate the RNN data requirement. Then, we test them in various scenarios and analyze the parameter change of our controller under different stiffness and damping factors. The experiments are carried out in the simulation environment mentioned in Section III-A.

1) Controller Comparison: First, we evaluate the controllers on a task involving sinusoidal tracking followed by stabilization at selected target angles, as shown in Fig. 8(a). This task includes dynamic following, abrupt transition, and steady-state maintenance. The tracking errors, computational times per step, number of parameters, and training times are reported in Table II. All the errors included in this work are mean average

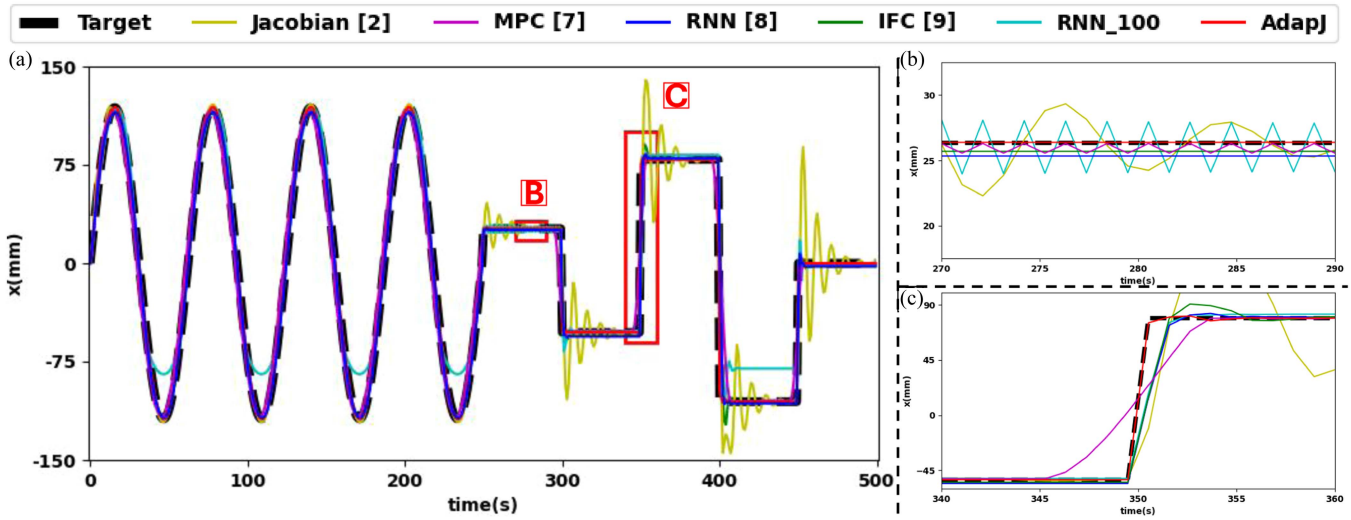


Fig. 8. (a) The target trajectory (dotted black) and the robot motion under the control of the original Jacobian controller [2] (yellow), MPC [7] (magenta), RNN [8] (blue), IFC [9] (green), RNN_100 (cyan), and our controller (red). (b) and (c) show zoomed-in views of the regions highlighted by red boxes in (a), illustrating detailed behavior on (b) maintaining a static state at 26 mm and (c) abrupt target transition from -52 mm to 78 mm, respectively.

TABLE II
AVERAGE ERRORS, STANDARD DEVIATIONS, COMPUTATIONAL TIMES PER STEP, PARAMETER NUMBERS, AND TRAINING TIMES IN SIMULATION

	Error (mm)	Computational Time (ms)	Parameter Number	Traning Time (ms)
Jacobian [2]	7.31 ± 13.31	$\mathbf{0.01}$	$\mathbf{1}$	$\mathbf{1.36}$
MPC [7]	7.48 ± 11.02	1.89	4	2.62
RNN [8]	2.34 ± 6.81	0.25	13089	219.17
IFC [9]	1.99 ± 6.87	0.26	13090	219.17
RNN_100	8.48 ± 10.84	0.25	13089	78.29
Ours	$\mathbf{0.30 \pm 0.56}$	0.04	3	2.02

errors. Compared to RNN, RNN_100 exhibits reduced accuracy in both sine wave tracking and -105 mm maintenance tasks, as shown in Fig. 8(a), demonstrating the necessity of a large amount of data for RNN. However, with the same amount of training data (100), AdapJ obviously outperforms any other controllers on all the various tasks, leveraging a concise linear format and online updating. The initial parameters are $A_0 = 2.89, A_1 = -1.86, B_0 = -0.55$. This result indicates that soft robot motions do not follow the Jacobian principle, which is $A_0 = -A_1 = J^{-1}, B_0 = 1$.

Fig. 8(b) and (c) illustrates the controller performance under steady-state maintenance and abrupt target transition with large deformation, respectively. As shown in Fig. 8(b), the Jacobian controller oscillates while gradually approaching the steady-state target, whereas MPC shares a similar behavior with reduced error. Both RNN and IFC stabilize around 26 mm, with IFC achieving a lower error due to its online error compensation mechanism. However, RNN_100 exhibits persistent oscillations, indicating that RNN fails to capture inverse kinematics with limited data. The degraded performance of RNN_100 underscores a key limitation of offline data-driven approaches: their strong dependence on the quantity and coverage of training data, and insufficient data results in a significant loss of control accuracy. Meanwhile, our AdapJ successfully accomplishes the steady-state maintenance task with the lowest error.

In the abrupt target transition shown by Fig. 8(c), the Jacobian controller still exhibits severe oscillations during the transition phase, and hence, a high deviation. Under MPC control, the robot leaves the original target (-52 mm) before the target transition due to the predictive nature of MPC. MPC, RNN, RNN_100, and IFC reach the new target (78 mm) with noticeable delay, exhibiting overdamped responses and the soft robotics hysteresis. Meanwhile, our proposed controller, trained with the same limited dataset, closely follows the target with minimal overshoot and fast convergence, demonstrating its superior convergence speed, high tracking precision, and strong data efficiency.

Considering both online computational cost and initialization time shown in Table II, the Jacobian approach is the fastest due to its simple structure, proving that it is one of the simplest controllers in soft robotics. In the Jacobian controller, only a single matrix J is required to be initialized and updated, whereas our controller updates three matrices A_*, B_* sharing comparable size. Although RNN, RNN_100, and IFC contain significantly more parameters than MPC and require longer initialization times (>70 ms), they do not perform online updates. In contrast, the model update and actuation decision via optimization in MPC [7] results in a longer time than direct actuation decision using RNN in (5). Meanwhile, our controller achieves a shorter online computational time than all other controllers except the Jacobian controller, aligning with the statement in Fig. 1, and requires a short initialization time similar to MPC (<3 ms). The computational time is the average obtained from the entire motion simulation process, which lasts 500 s in the simulation environments and is performed on the PC described above.

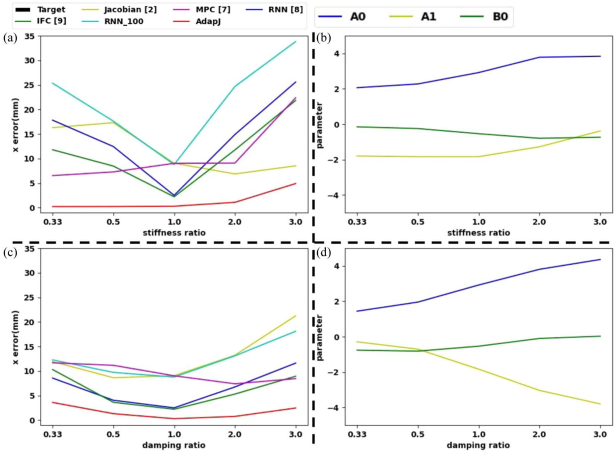


Fig. 9. Robot motion error under (a) stiffness and (c) damping change. The A_*, B_* adjustment in response to the (b) stiffness and (d) damping change. The ratio 1 represents the original scenario, and a bigger ratio represents a higher stiffness/ damping coefficient.

2) Adaptability Validation: After controller comparison in the default situation, we evaluate controller adaptability under different scenarios. Specifically, we modify the stiffness and damping based on the stiffness and damping ratios, and the average errors are depicted in Fig. 9(a) and (c). The ratio 1 represents the default physical parameter, and a higher ratio represents a higher parameter. RNN and RNN_100, as offline controllers, induce higher tracking errors under different physical properties, especially with the stiffness change. The online updating Jacobian controller fails under some extreme situations, reflecting its limited adaptability. Leveraging the online error compensation component, IFC outperforms the original RNN in most adaptability tests, and MPC also exhibits some adaptability to a certain extent, as indicated by the robust errors. However, our controller still obviously outperforms the other controllers with low errors in all cases (<5 mm).

We further analyze the adaptation of the parameters A_*, B_* in response to the physical property change, as shown in Fig. 9(b), (d). A simple soft robot finger is applied because its 1-D parameter space enables direct visualization. The parameter A_0 corresponds to the feedforward term, and higher actuation is required in systems with higher stiffness or damping. Consequently, A_0 roughly increases as stiffness and damping coefficients rise. For the discussion of A_1, we consider the simplified control expression $a_{aj,t} = A_0 0 + A_1 s_t + B_0 0$, where the robot is expected to return to the original state 0 from the current state s_t. A high-stiffness system tends to return to the equilibrium (0), hence $\|A_1\|$ decreases. Meanwhile, a high-damping system tends to resist motion and maintain the current state (s_t), and $\|A_1\|$ should be larger. Regarding B_0, we consider the case $a_{aj,t} = A_0 0 + A_1 0 + B_0 a_{t-1}$, and the feedforward term $B_0 a_{t-1}$ serves as hysteresis compensation. In high-damping systems, motion is heavily constrained, leading to a lower $\|B_0\|$. In contrast, high-stiffness systems store more energy and exhibit stronger memory effects, necessitating a larger $\|B_0\|$ for effective compensation.

TABLE III
COMPUTATIONAL TIME PER STEP (MS), AVERAGE ERRORS AND STANDARD DEVIATIONS (MM) IN THE REAL EXPERIMENTS

	MPC [7]	RNN [8]	IFC [9]	AdapJ
computational time	7.03	1.01	1.15	**0.35**
spiral-10 Hz	1.16 ± 0.78	1.26 ± 0.53	1.13 ± 0.48	$\mathbf{0.79 \pm 0.46}$
triangle-10 Hz	1.38 ± 0.68	1.16 ± 0.61	1.06 ± 0.54	$\mathbf{0.92 \pm 0.53}$
spiral-8 Hz	1.57 ± 0.90	5.24 ± 4.56	5.27 ± 4.39	$\mathbf{0.89 \pm 0.52}$
triangle-8 Hz	1.80 ± 0.95	6.65 ± 4.21	6.43 ± 3.95	$\mathbf{1.09 \pm 0.68}$
spiral-15 Hz	1.28 ± 0.85	1.75 ± 0.64	1.49 ± 0.56	$\mathbf{1.23 \pm 0.89}$
triangle-15 Hz	1.27 ± 0.71	1.76 ± 0.74	1.43 ± 0.62	$\mathbf{1.22 \pm 0.82}$
spiral-Robot 2	1.41 ± 0.86	1.34 ± 0.64	1.17 ± 0.57	$\mathbf{0.97 \pm 0.61}$
triangle-Robot 2	1.26 ± 0.65	1.31 ± 0.58	1.11 ± 0.53	$\mathbf{0.93 \pm 0.53}$
spiral-Robot 3	1.31 ± 0.89	2.39 ± 0.85	2.04 ± 0.75	$\mathbf{0.81 \pm 0.49}$
triangle-Robot 3	1.43 ± 0.76	2.39 ± 1.04	2.01 ± 0.90	$\mathbf{0.98 \pm 0.60}$

B. Real Experiment Results

1) Controller Comparison: Using the real experiment setup in Section III-B, we compare the control accuracy of MPC, RNN, IFC, and AdapJ. We first initialize the controller matrices A_*, B_* following (7) with only 100 samples and the other controllers with 5000 samples, as illustrated in Fig. 7(d). The initial extended inverse Jacobian matrix A_*, B_* are

$$A_0 = \begin{bmatrix} 3.01 & 0.06 \\ -0.03 & 3.00 \end{bmatrix}, \; A_1 = \begin{bmatrix} -1.79 & -0.08 \\ 0.01 & -1.77 \end{bmatrix} \\ B_0 = \begin{bmatrix} -0.49 & 0.00 \\ 0.02 & -0.57 \end{bmatrix} \tag{12}$$

which are much different from the Jacobian controller constraints, which are $A_0 = -A_1 = J^{-1}, B_0 = \boldsymbol{I}$ mentioned in (4). Such a misalignment demonstrates that the original Jacobian controller is not suitable for soft robots.

We evaluate the controllers on trajectory-following tasks using both a spiral and a star-shaped path, as shown in Fig. 10. These trajectories are chosen to assess controller performance across curved segments, straight lines, and sharp turns. The robot motions are shown in Fig. 10(a), and each experiment is repeated for three trials, as in the following experiments. The average tracking errors, standard deviations, and the computational time per step are included in Table III. Our controller is linear in structure and employs only 100 samples for initialization, far fewer than the samples used for RNN training (5000). Despite this, AdapJ achieves lower tracking errors than MPC, RNN, and IFC. Meanwhile, AdapJ achieves the shortest computational time. The lower errors and shorter computational times are consistent with the simulation results and structure simplicity statement in Fig. 1. The computation times in the real-world experiments are longer than in the simulation due to the increased dimensionality of both the state and actuation spaces.

2) Adaptability Validation: The simple structure of our controller makes it possible to update the matrices A_*, B_* online and adjust to situations different from the initial one, which cannot be achieved by offline controllers like RNN. In this case, we include different frequencies (8 and 15 Hz) and different stiffnesses (Robots 2 and 3) to validate the adaptability of our controller. The robot motions under the control of MPC, RNN, IFC, and the proposed controller are shown in Fig. 10(b)–(e). The errors and standard deviations are included in Table III.

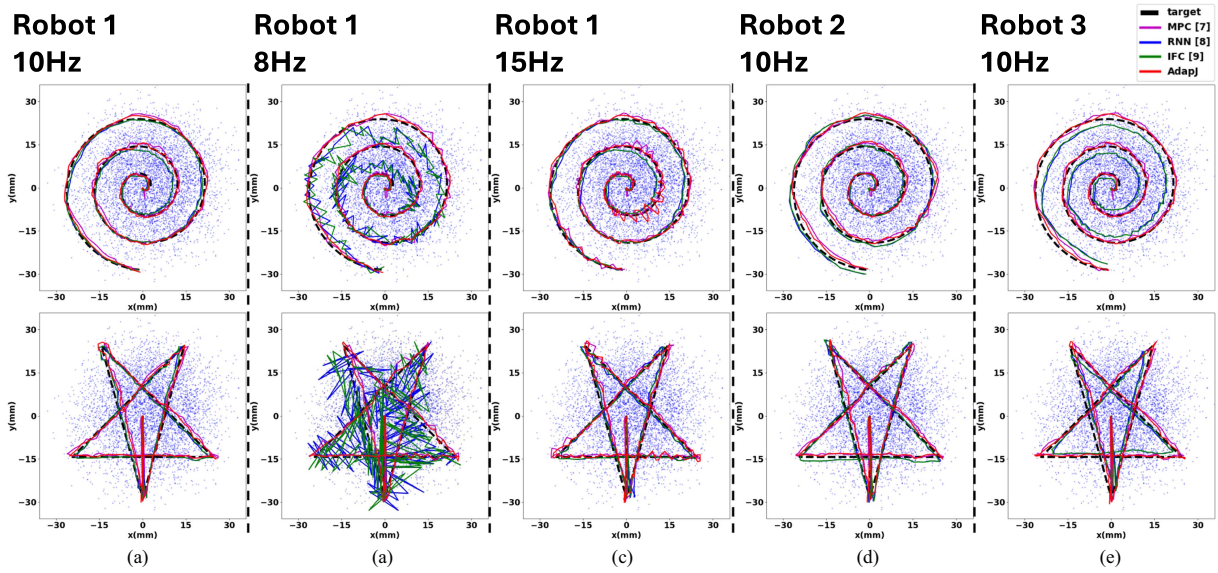


Fig. 10. Target trajectory (black) and measured robot motion of the (a) Robot 1 under 10 Hz, (b) Robot 1 under 8 Hz, (c) Robot 1 under 15 Hz, (d) Robot 2, and (e) Robot 3 under the control of MPC [7] (purple), RNN [8] (blue), IFC [9] (green), and AdapJ (red). The light blue dots show the robot's working space. 4 (controller number) $\times$ 2 (trajectory number) $\times$ 5 (scenario number) $\times$ 3 (trial number for each condition) $=$ 120 experiments are included.

Frequency changes induce misalignment between the training and the test situation. With limited inference ability and a lack of online updating, the RNN controller can roughly follow the trajectories but produces aggressive motions and higher errors, as shown in Fig. 10(b) and (c). IFC, composed of RNN and an online compensation mechanism, achieves adaptability with lower errors than the original RNN. Meanwhile, MPC and our controller can update online and follow the trajectories with different frequencies and lower errors, while AdapJ achieves the lowest tracking errors in these controllers.

Robot 2 is softer than Robot 1, and Robot 3 is harder than Robot 1. Therefore, in the trajectory following tasks, the RNN controller trained on Robot 1 will produce larger real trajectories on Robot 2 and smaller real trajectories on Robot 3, as shown in Fig. 10(d) and (e). Similarly, AdapJ achieves better tracking accuracy than the other controllers in the physical property adaptability experiments. Overall, under these diverse scenarios (different frequencies, different stiffness, and different damping in simulation), our controller can update online and follow trajectories with the lowest errors, demonstrating its adaptability.

3) Disturbance and Obstacle Rejection: In addition to the frequency and physical property adaptability, we also conduct experiments to validate the robustness of our controller under dynamic disturbance and fixed obstacles. In this experiment, the manipulator is controlled to follow a Lissajous trajectory for five rounds, as shown in Fig. 11(a). In the first round, the manipulator can follow the trajectory. In the second round, we continuously and randomly pitch the manipulator. Under the dynamic disturbance, the manipulator can still roughly follow the trajectory and recover the following accuracy in the third round without disturbance. In the fourth round, we fix an obstacle near chamber III, as shown in Fig. 11(b). In this case, AdapJ follows the trajectory by online updating and decides on a larger a_v, as illustrated in Fig. 11(c). After the adaptation to the obstacle, we remove it in the fifth round, and AdapJ can still follow the

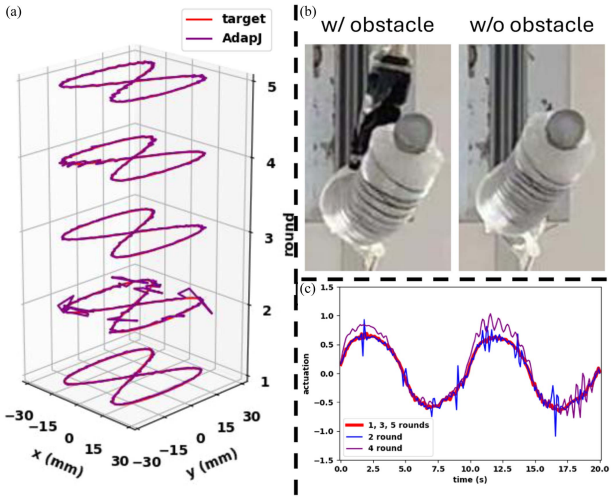


Fig. 11. (a) The target (red) and measured (purple) trajectories under the control of AdapJ in five rounds. (b) The soft manipulator deformation with and without an obstacle. (c) The average a_v in the 1st, 3rd, and 5th rounds without obstacle (red) and a_v in the 2nd round under dynamic disturbance (blue) and in the 4th round with an obstacle (purple).

target trajectory. The experiment results in Fig. 11 demonstrate that the AdapJ is adaptive to dynamic disturbances and fixed obstacles with the help of the extended Jacobian format and the online updating strategy. The experiment video is included in the Supplemental Video.

V. Conclusion

In this work, we return to one of the simplest soft manipulator controllers, the Jacobian controller, and propose an extended Jacobian controller adaptive to different situations. Such a controller is inspired by the linear format of the Jacobian controller and the parameter independence of the RNN controller. We also employ motor babbling and batch optimization as the controller

initialization strategy following the RNN controller and the Gauss–Newton method as the updating strategy following MPC. As mentioned in Fig. 1, this controller is simpler than any controller except the Jacobian controller and outperforms any other controllers on trajectory following tasks. In addition, such a controller demonstrates its adaptability in various scenarios, like different physical properties, frequencies, and even external disturbances. AdapJ achieves internal adaptability defined in [28], demonstrating adaptability across different robots. As a soft manipulator controller, this controller addresses hysteresis by including previous states and actuations as input and relaxing the parameter coupling, and addresses nonlinearity by updating online. As mentioned in [10], soft robot control researchers may blame the Jacobian controller's low efficiency and limited application on the linear format. However, this work demonstrates that a linear controller can achieve satisfying performance on soft robots. All we need is a proper linear format, not the one directly transferred from the rigid robot research.

Although the proposed AdapJ outperforms most controllers in terms of control accuracy, computational time, and adaptability, there are still limitations to this strategy. First, theoretical guarantees remain underdeveloped. The coverage of parameter updating leveraging the Gauss–Newton method has been proven in [7]. However, providing formal proofs of coverage and robustness for data-driven controllers in soft robotics is still an open challenge. A key difficulty lies in establishing an appropriate forward model that can adequately capture the complex dynamics of soft robots. Although some studies have employed the Jacobian forward model for soft robotics [26], such a model is inherently limited due to the simplifying assumption ($\triangle s = J \triangle a$), which does not hold in soft robotic systems as discussed before. Also, data-driven approaches, such as the statistical controllers [13], [14] and RNN controllers [8], [15], always meet challenges with theoretical validation due to the large parameter spaces and reliance on empirical training. Although the parameters in our controller A_*, B_* have been explained roughly in Section IV-A2, providing better explainability than RNN, a rigorous theoretical analysis of their behavior and guarantees is still absent. Enhancing the interpretability and provability of such adaptive data-driven controllers remains a vital direction for future work. Second, this study primarily proposes this concept and initially validates its accuracy and adaptability in soft manipulators. In future work, we would like to extend and adapt this controller to more complex soft robots, such as modular soft robots and mobile soft robotic platforms. The high dynamic behaviors of modular soft robots caused by the modularity may require more previous terms. Also, the nonhomogeneous matrix induced by the higher degrees of freedom may be a challenge.

Acknowledgment

The authors would like to thank Godfried Jansen Van Vuuren for the help during robot setup manufacturing and Qinglin Zhu and Jingyi Sun for the discussion about the applications of neural networks.

References

[1] S. Tully, G. Kantor, M. A. Zenati, and H. Choset, "Shape estimation for image-guided surgery with a highly articulated snake robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2011, pp. 1353–1358.
[2] M. C. Yip and D. B. Camarillo, "Model-less feedback control of continuum manipulators in constrained environments," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 880–889, Aug. 2014.
[3] J. F. Queißer, K. Neumann, M. Rolf, R. F. Reinhart, and J. J. Steil, "An active compliant control mode for interaction with a pneumatic soft robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 573–579.
[4] Z. Tang et al., "Meta-learning-based optimal control for soft robotic manipulators to interact with unknown environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 982–988.
[5] Z. Chen, Q. Guan, J. Hughes, A. Menciassi, and C. Stefanini, "A versatile neural network configuration space planning and control strategy for modular soft robot arms," *IEEE Trans. Robot.*, vol. 41, pp. 4269–4282, 2025.
[6] H. Jiang et al., "Hierarchical control of soft manipulators towards unstructured interactions," *Int. J. Robot. Res.*, vol. 40, no. 1, pp. 411–434, 2021.
[7] Z. Q. Tang, H. L. Heung, K. Y. Tong, and Z. Li, "A novel iterative learning model predictive control method for soft bending actuators," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 4004–4010.
[8] Z. Chen, X. Ren, M. Bernabei, V. Mainardi, G. Ciuti, and C. Stefanini, "A hybrid adaptive controller for soft robot interchangeability," *IEEE Robot. Automat. Lett.*, vol. 9, no. 1, pp. 875–882, Jan. 2024.
[9] X. Li et al., "Disturbance-adaptive tapered soft manipulator with precise motion controller for enhanced task performance," *IEEE Trans. Robot.*, vol. 40, pp. 3581–3601, 2024.
[10] Z. Chen et al., "Data-driven methods applied to soft robot modeling and control: A review," *IEEE Trans. Automat. Sci. Eng.*, vol. 22, pp. 2241–2256, 2025.
[11] C. Della Santina, R. K. Katzschmann, A. Biechi, and D. Rus, "Dynamic control of soft robots interacting with the environment," in *Proc. IEEE Int. Conf. Soft Robot.*, 2018, pp. 46–53.
[12] F. Renda, F. Boyer, J. Dias, and L. Seneviratne, "Discrete cosserat approach for multisection soft manipulator dynamics," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1518–1533, Dec. 2018.
[13] B. Yu, J. d. G. Fernández, and T. Tan, "Probabilistic kinematic model of a robotic catheter for 3D position control," *Soft Robot.*, vol. 6, no. 2, pp. 184–194, 2019.
[14] G. Fang et al., "Vision-based online learning kinematic control for soft robots using local gaussian process regression," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1194–1201, Apr. 2019.
[15] D. Wu et al., "Deep-learning-based compliant motion control of a pneumatically-driven robotic catheter," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 8853–8860, Oct. 2022.
[16] K.-H. Lee et al., "Fem-based soft robotic control framework for intracavitary navigation," in *Proc. IEEE Int. Conf. Real-time Comput. Robot.*, 2017, pp. 11–16.
[17] M. Li, R. Kang, D. T. Branson, and J. S. Dai, "Model-free control for continuum robots based on an adaptive Kalman filter," *IEEE/ASME Trans. Mechatron.*, vol. 23, no. 1, pp. 286–297, Feb. 2018.
[18] M. C. Yip, J. A. Sganga, and D. B. Camarillo, "Autonomous control of continuum robot manipulators for complex cardiac ablation tasks," *J. Med. Robot. Res.*, vol. 2, no. 1, 2017, Art. no. 1750002.
[19] Y. Jin et al., "Model-less feedback control for soft manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2916–2922.
[20] T. Liu et al., "Iterative Jacobian-based inverse kinematics and open-loop control of an MRI-guided magnetically actuated steerable catheter system," *IEEE/ASME Trans. Mechatron.*, vol. 22, no. 4, pp. 1765–1776, Aug. 2017.
[21] T. George Thuruthel, E. Falotico, M. Manti, A. Pratesi, M. Cianchetti, and C. Laschi, "Learning closed loop kinematic controllers for continuum manipulators in unstructured environments," *Soft Robot.*, vol. 4, no. 3, pp. 285–296, 2017.
[22] G. Li et al., "DNN-based predictive model for a batoid-inspired soft robot," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 1024–1031, Apr. 2022.
[23] T. G. Thuruthel, E. Falotico, M. Manti, and C. Laschi, "Stable open loop control of soft robotic manipulators," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 1292–1298, Apr. 2018.
[24] Z. Chen, M. Bernabei, V. Mainardi, X. Ren, G. Ciuti, and C. Stefanini, "A novel and accurate BiLSTM configuration controller for modular soft robots with module number adaptability," 2024, *arXiv:2401.10997*.

[25] Y.-Y. Wu and N. Tan, "Model-less feedback control for soft manipulators with Jacobian adaptation," in *Proc. Int. Symp. Auton. Syst.*, 2020, pp. 217–222.
[26] C. Pan, Z. Deng, C. Zeng, B. He, and J. Zhang, "Optimal visual control of tendon-sheath-driven continuum robots with robust Jacobian estimation in confined environments," *Mechatronics*, vol. 104, 2024, Art. no. 103260.
[27] X. Ren et al., "Design and analytical modeling of a dumbbell-shaped balloon anchoring actuator for safe and efficient locomotion inside gastrointestinal tract," *Soft Robot.*, vol. 12, pp. 374–385, 2024.
[28] Z. Chen et al., "A survey on soft robot adaptability: Implementations, applications, and prospects [survey]," *IEEE Robot. Automat. Mag.*, to be published, doi: 10.1109/MRA.2025.3584346.

Zixi Chen (Member, IEEE) received the M.Sc. degree in control systems from Imperial College, London, U.K., in 2021. He is currently working toward the Ph.D. degree in biorobotics from Scuola Superiore Sant'Anna of Pisa, Pisa, Italy.

His research interests include optical tactile sensors and data-driven approaches in soft robots.

Xuyang Ren received the M.Sc. degree in mechanical engineering from Tianjin University, Tianjin, China, in 2019, and the Ph.D. degree in biorobotics from the BioRobotics Institute of Scuola Superiore Sant'Anna, Pisa, Italy, in 2023.

His main research interests include endoscopic devices, surgical robotics, and soft actuators for medical applications.

Yuya Hamamatsu (Graduate Student Member, IEEE) received the M.Sc. degree in environment from the University of Tokyo, Tokyo, Japan, in 2020. He is currently working toward the Ph.D. degree in computer systems and engineering with the Centre for Biorobotics, Tallinn University of Technology, Estonia.

His research interests are control theory with machine learning techniques on field robotics.

Gastone Ciuti (Senior Member, IEEE) received the master's degree (Hons.) in biomedical engineering from the University of Pisa, Pisa, Italy, in 2008, and the Ph.D. degree (Hons.) in biorobotics from The BioRobotics Institute of Scuola Superiore Sant'Anna, Pisa, Italy, in 2011.

He is currently an Associate Professor of Bioengineering with Scuola Superiore Sant'Anna, leading the Healthcare Mechatronics Laboratory. He has been a Visiting Professor with the Sorbonne University, Paris, France, and Beijing Institute of Technology, Beijing, China, and a Visiting Student with the Vanderbilt University, Nashville, TN, USA, and Imperial College London, London, U.K. He is the coauthor of more than 110 international peer reviewed papers on medical robotics and the inventor of more than 15 patents. His research interests include robot/computer-assisted platforms, such as tele-operated and autonomous magnetic-based robotic platforms for navigation, localization and tracking of smart and innovative devices in guided and targeted minimally invasive surgical and diagnostic applications, e.g. advanced capsule endoscopy.

Prof. Ciuti is a Member of the Technical Committee in BioRobotics of the IEEE Engineering in Medicine and Biology Society (EMBS). He is an Associate Editor of the IEEE JOURNAL OF BIOENGINEERING AND HEALTH INFORMATICS, IEEE TRANSACTION ON BIOMEDICAL ENGINEERING and of the IEEE TRANSACTION ON MEDICAL ROBOTICS AND BIONICS.

Donato Romano received the M.Sc. degree in agriculture, food, and environmental science and technologies (hons.) from the University of Pisa, in 2014, and the Ph.D. degree in bioRobotics (hons.) from Scuola Superiore Sant'Anna, in 2018.

He is currently an Associate Professor with The BioRobotics Institute of Scuola Superiore Sant'Anna of Pisa, Pisa, Italy, where he coordinates the Biorobotic Ecosystems Laboratory. His research interests include the activities on bioinspired and biomimetic robotics, and in particular on animal–robot interaction, biohybrid systems, natural and biohybrid intelligence, ethorobotics, neuroethology. Romano received national and international recognitions for his research.

Prof. Romano also has been visiting Researcher with Khalifa University, Abu Dhabi (UAE). He is Member of the Editorial Board for many International Scientific Journals. Romano is Coordinator, PI, or partner of several national and international research projects.

Cesare Stefanini (Member, IEEE) received the M.Sc. (Hons.) degree in mechanical engineering and the Ph.D. (Hons.) degree in microengineering from Scuola Superiore Sant'Anna (SSSA), Pisa, Italy, in 1997 and 2002, respectively.

He is currently a Professor with the Department of Robotics, Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates. He is the author or coauthor of more than 370 articles in refereed international journals and in international conference proceedings. He is the inventor of more than 15 international patents, nine of which have been industrially exploited by international companies. His research interests include underwater robotics, bioinspired systems, biomechatronics, and micromechatronics for medical and industrial applications.

Prof. Stefanini was the recipient of international recognitions for the development of novel actuators for microrobots, and he has been a Visiting Researcher with Stanford University, Center for Design Research, and the Director of the Healthcare Engineering Innovation Center, Khalifa University, Abu Dhabi, UAE. He was a Professor and the Director with the BioRobotics Institute, SSSA, where he was also the Head of the Creative Engineering Lab. He was the recipient of the "Intuitive Surgical Research Award." He is a Member of the Academy of Scientists of the UAE and of the IEEE Societies RAS (Robotics and Automation) and EMBS (Engineering in Medicine and Biology).