Adaptive Dynamic Sliding Mode Control of Soft Continuum Manipulators

Amirhossein Kazemipour\textsuperscript{1,2}, Oliver Fischer\textsuperscript{1}, Yasunori Toshimitsu\textsuperscript{1,3}, Ki Wan Wong\textsuperscript{1}, Robert K. Katzschmann\textsuperscript{1}

\textbf{Abstract}—Soft robots are made of compliant materials and perform tasks that are challenging for rigid robots. However, their continuum nature makes it difficult to develop model-based control strategies. This work presents a robust model-based control scheme for soft continuum robots. Our dynamic model is based on the Euler-Lagrange approach, but it uses a more accurate description of the robot’s inertia and does not include oversimplified assumptions. Based on this model, we introduce an adaptive sliding mode control scheme, which is robust against model parameter uncertainties and unknown input disturbances. We perform a series of experiments with a physical soft continuum arm to evaluate the effectiveness of our controller at tracking task-space trajectory under different payloads. The tracking performance of the controller is around 38\% more accurate than that of a state-of-the-art controller, i.e., the inverse dynamics method. Moreover, the proposed model-based control design is flexible and can be generalized to any continuum robotic arm with an arbitrary number of segments. With this control strategy, soft robotic object manipulation can become more accurate while remaining robust to disturbances.

I. INTRODUCTION

Soft robotics is a rapidly growing sub-field of robotics. Soft robots are fabricated with compliant and deformable materials, and they can perform tasks that would be extremely challenging for conventional rigid robots [1]. The inherent compliance of soft manipulators distinguishes them from other types of robots, making them more suitable for interacting with humans and their environment [2]. Their continuum properties also enable them to adapt to complex environments in which rigid robots may fail [3]. However, their continuum nature comes at a cost: developing model-based control strategies is complicated.

There are several approaches to developing a dynamic model for soft robotic arms. For example, a dynamic model can be based on Koopman Operator theory [4], [5], reduced-order finite element models [6], polynomial curvature fitting [7], or discrete Cosserat rod models [8]. In [9]–[11], the dynamic model for the soft continuum arms is based on an approach called Augmented Rigid Body formulation. In this approach, the soft robot’s motion is approximated with a classic rigid link manipulator, and then it is transformed back into a Piecewise Constant Curvature (PCC) formulation for control. However, as the number of Constant Curvature (CC) segments increases, the auxiliary rigid states in the model significantly increase the computational burden in the controller loop. As an alternative, a dynamic model of a soft continuum arm can be derived from the integral Lagrangian approach [12] if we assume that the mass distribution along the arm is continuous. This eliminates the need for augmented rigid states in the model. However, when this approach is used to model a continuum arm that has multiple sections, computing the dynamic terms in real-time implementation is computationally inefficient. To address this issue, the mass of each section can be approximated discretely by lumping it into a single mass point along the arm [13], [14]. However, the lumped-mass models in [13], [14] assume that the mass of each segment is located at the tip of each segment, and this leads to an inaccurate description of the dynamics, particularly for segments with a high length-to-diameter ratio. Thus, previous models tend to suffer from inefficiencies in the computation of the dynamic parameters or model oversimplifications, resulting in poor performance of model-based controllers.

Prior works on closed-loop control strategies for a soft robotic arm that has to perform dynamic tasks include proportional-derivative (PD) control with dynamic compensation [9]–[11] and observer-based dynamic control with a reduced-order finite element model [6]. However, these controllers assume perfect knowledge of the model and its parameters. To make the robot robust to model uncertainties, an adaptive kinematic controller is proposed in [15] and tested on a physical soft robotic arm; however, pure kinematic-based control strategies are not suitable for robots that perform dynamic tasks. To tackle this problem, a popular adaptive control scheme, which was introduced by Li and Slotine [16], is implemented in curvature-space for a sim-

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig1.png}
\caption{Panel (a) shows a dynamically controlled soft robot that is tracking a desired task-space trajectory while carrying a payload with its soft gripper. This robotic arm consists of two segments, and each segment has three chambers that can deform under pressurization. The soft arm is controlled using a model-based adaptive control strategy, as is illustrated in panel (b). This robot is actuated using a proportional valve controller, and the motion capture cameras are used to measure its curvature.}
\end{figure}
be derived from the Lagrangian as \( \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = u_i \), for \( i = 1, \ldots, 2n \). \( u_i \) denotes the generalized force associated with generalized coordinate \( q_i \in \mathbb{R}^{2n} \), and \( n \) denotes the total number of continuum segments.

1) Potential Energy: The elastic and gravitational potential energies can be computed as \( \mathcal{U}_e = \frac{1}{2} \sum_{i=1}^{n} k_{s,i} \theta_i^2 \) and \( \mathcal{U}_g = -\sum_{i=1}^{n} m_i g_0 T_{0,ci} \), respectively. \( k_{s,i} \), \( m_i \), and \( r_{0,ci} \) denote the stiffness coefficient, the mass, and the center of mass (CoM) position of the \( i \)-th segment, respectively. \( g_0 = (0 \ 0 \ -g)^T \) is the gravity acceleration vector in the base reference frame. \( g > 0 \) denotes the gravitational acceleration constant.

2) Center of Mass Position: \( r_{0,ci} \) denotes the i-th segment’s CoM position with respect to the robot’s base frame \( \{S_0\} \). It can be computed as:
\[
\begin{pmatrix}
0 \\
T_1(\phi_1, \theta_1) \\
\vdots \\
T_{i-1}(\phi_{i-1}, \theta_{i-1}) \\
\end{pmatrix}
\begin{pmatrix}
1 \\
r_{i,ci} \\
\vdots \\
r_{n,ci} \\
\end{pmatrix}
\begin{pmatrix}
\frac{\rho_i \cos \theta_i}{2} - \eta_i \\
0 \\
\vdots \\
0 \\
\end{pmatrix}
\begin{pmatrix}
\frac{\rho_i \sin \theta_i}{2} \\
0 \\
\vdots \\
0 \\
\end{pmatrix}
\end{pmatrix}
\]
where \( \rho_i \) is the curvature radius of the i-th segment, and \( \eta_i = \frac{2\rho_i \sin \theta_i}{\eta} \) is the distance between the CoM and the center of curvature.

Remark 1. In contrast to previous models [13], [14], which assume that the mass is located at the segment’s tip, eq. (2) considers that the mass is located at each segment’s centroid, leading to a more realistic representation of the CoM position.

3) Kinetic Energy: The total kinetic energy \( \mathcal{T} \) can be derived from each segment’s individual energy terms as \( \mathcal{T} = \sum_{i=1}^{n} \frac{1}{2} m_i \dot{q}_i^2 \), where \( m_i \) denotes the mass. The linear velocity of each segment can be computed as \( \dot{q}_i = \frac{\partial r_{0,ci}}{\partial \dot{q}_i} \). \( r_{0,ci} \) is obtained in the same way as in eq. (1).

We note that the kinetic energy neglects the contributions of rotational energies, as they are much lower than the translational energies [14].

4) Dynamic Terms: The dynamic terms can be obtained from the system’s potential and kinetic energies [21]. The dynamics model in compact form can be written as:
\[
M(q) \ddot{q} + C(q, \dot{q}) \dot{q} + D(q) \dot{q} + g(q) + k(q) = A(q)p + d,
\]
where \( p \in \mathbb{R}^{c} \) indicates the air pressure in the robot’s fluidic chambers, and the superscript \( c \) specifies the total number of chambers. \( M \in \mathbb{R}^{2n \times 2n} \), \( C \in \mathbb{R}^{2n \times 2n} \), \( g \in \mathbb{R}^{2n \times 1} \), \( k \in \mathbb{R}^{2n \times c} \), and \( A \in \mathbb{R}^{2n \times c} \) are the generalized inertia matrix, Coriolis/centrifugal matrix, gravity force vector, elastic force vector, and actuator mapping matrix, respectively. \( D \in \mathbb{R}^{2n \times 2n} \) is the damping matrix, which is described in [9]. \( d \in \mathbb{R}^{2n} \) denotes the unknown input disturbances.

C. Linear Parameterization

Neglecting the unknown input disturbances, the dynamic equations in eq. (3) can be re-written in a linear form as:
\[
Y(q, \dot{q}, \ddot{q}) = A(q)p,
\]
where \( a \in \mathbb{R}^r \) contains the \( r \) number of dynamic coefficients in the robot. \( Y \in \mathbb{R}^{2n \times r} \) is called the regression matrix,
which is a known time-varying matrix that depends only on \( q, \dot{q}, \) and \( \ddot{q} \). Vector \( a \) combines the robot’s physical parameters (i.e., the length, mass, stiffness, and damping coefficients of the segments).

III. CONTROL SYNTHESIS

In this section, we present the proposed adaptive control scheme for soft continuum manipulators in both the curvature space and the task space.

A. Adaptive Control: Curvature Space

We first begin by modifying the reference trajectories that are presented in [19]. These trajectories allow us to define a terminal sliding manifold. The proposed reference trajectories at velocity and acceleration levels are as follows:

\[
\begin{align*}
\dot{q}_r &= q_d + \Lambda \text{sign}^\alpha(q_d - q), \\
\ddot{q}_r &= \ddot{q}_d + \alpha \Lambda |q_d - q|^{\alpha - 1}(q_d - \dot{q}),
\end{align*}
\]

where \( \text{sign}^\alpha(x) \triangleq |x|^\alpha \text{sign}(x) \). \( q_d \) is the desired trajectory in the configuration space, \( \Lambda \) is a constant diagonal matrix with positive diagonal entries, and the exponent \( \alpha \in (0.5, 1) \) is a constant scalar. Accordingly, the nonlinear terminal sliding manifold \( s \) can be defined as \( s \triangleq q - q_r = \dot{e}_q + \Lambda \text{sign}^\alpha(e_q) \), where \( e_q \triangleq q - q_d \) and \( \dot{e}_q \triangleq \dot{q} - \dot{q}_d \).

The control law is presented as follows:

\[
p = A^1(q) \left( M(q) \ddot{q}_r + C(q, \dot{q}) \dot{q}_r + D(q) \dot{q}_r + \ddot{q} + \dot{\Theta}(q) \right) + \dot{\Theta}(q) \left( K_D s - \dot{b} \text{sign}(s) \right),
\]

where \( K_D > 0 \) is a diagonal matrix, and \( A^1(q) \) is the pseudo-inverse of the mapping matrix. The estimated feedback-linearization terms in the control law can be computed using the regressed mapping matrix \( Y \) introduced in eq. (4) as:

\[
M\ddot{q}_r + C\dot{q}_r + \dot{D}q + \dot{\Theta} = Y(q, \dot{q}, \ddot{q}, \dot{q}_r, \dot{\Theta}).
\]

Accordingly, the control law in eq. (6) can be rewritten as:

\[
p = A^1(q) \left( Y(q, q_r, \dot{q}_r, \ddot{q}_r) \dot{q} - \dot{K}_D s - \dot{b} \text{sign}(s) \right).
\]

In the control synthesis, we have considered two adaptation laws:

\[
\dot{\Theta} = -\Gamma Y^T (q, \dot{q}, \ddot{q}, \dot{q}_r, \ddot{q}_r) s, \quad \dot{b} = \Psi |s|.
\]

The first adaptation law is used to estimate the uncertain dynamic coefficients in vector \( a \), where \( \dot{\Theta} \) is the estimation of dynamic coefficients \( a \). The second adaptation law is designed to make the control law robust to unknown external disturbances, where \( \dot{b} \) is the estimation of the upper bound of disturbances \( b \). Note that \( \Psi > 0 \) and \( \Gamma > 0 \) are the diagonal constant gain matrices.

For convenience in the stability proof, we first give the following lemma and assumption:

**Lemma 1.** If the Coriolis matrix is defined using Christoffel symbols, the matrix \((M - 2C)\) is skew-symmetric [21].

**Assumption 1.** The input disturbances \( d \) are bound by \(|d_i| \leq b_i, \forall i = 1, \ldots, n\), where \( b_i \in \mathbb{R} \) is an unknown positive scalar.

The stability of the control scheme can be proved by considering the following Lyapunov function:

\[
V(t) = \frac{1}{2} \dot{q}^T M \ddot{q} + \frac{1}{2} \ddot{\Theta} \Gamma^\dagger \ddot{\Theta} + \frac{1}{2} \dot{b}^T \Psi \dot{b},
\]

where \( \ddot{\Theta} = \ddot{\Theta} - \dot{\Theta} \) and \( \dot{b} = \dot{b} - \dot{b} \).

By replacing the control law in eq. (6) and using eq. (7), we have:

\[
\dot{V} = \frac{1}{2} \dot{q}^T M \ddot{q} + \frac{1}{2} \ddot{\Theta} \Gamma^\dagger \ddot{\Theta} + \frac{1}{2} \dot{b}^T \Psi \dot{b}.
\]

Replacing the adaptation laws in eq. (9) yields

\[
\dot{V} = \frac{1}{2} \dot{q}^T (M - 2C) s - \dot{b}^T Y \ddot{q} - \dot{b}^T K_D s + \dot{b}^T \dot{b}.
\]

(12)

Considering the skew-symmetric property in property [1] and the relation \( s^T d \leq |s|^2 \dot{b} \) followed by assumption [2], we obtain \( V \leq -s^T K_D s \), with \( K_D > 0 \). Therefore, the proposed control law in eq. (6) with the adaptation laws in eq. (9) force the trajectories to reach the sliding manifold \( s = 0 \). When the sliding condition \( s = 0 \) is reached, the trajectories are determined by the following differential equation:

\[
\dot{e}_q = -\Lambda \text{sign}^\alpha(e_q).
\]

(14)

By directly integrating eq. (14), it can be shown that \( |e_q(0)| \neq 0 \), the trajectories will reach \( e_q = 0 \) in a finite time. This can be determined by \( t_{f,1} = \frac{1}{\Lambda} (1 - \alpha) \text{sign}^\alpha(e_q(0)) \). Therefore, \( e_q = 0 \) is a terminal attractor (i.e., the tracking errors converge to zero in a finite time).

To avoid overestimating the dynamic coefficients, we use the boundary layer technique in [23], [24] by defining a new variable as \( s_{\Delta_i} = s_{i} - \phi_i \text{sat}_{\phi_i}(s_{i}) \), for \( i = 1, \ldots, n \), where \( \phi_i > 0 \) is the boundary layer thickness. The saturation function is defined as:

\[
\text{sat}_{\phi_i}(s) = \begin{cases} 
\text{sign}(s) & \text{if } |s| \geq \phi_i \\
\frac{s}{\phi_i} & \text{if } |s| < \phi_i
\end{cases}
\]

(15)

Inside the boundary layer \( |s| < \phi_i \), the new variable \( s_{\Delta,1} = s_i - \phi_i \) is defined and outside the boundary layer \( |s_i| \geq \phi_i \), the relation \( s_{\Delta,1} = s_i \) is satisfied. Accordingly, the adaptation laws are modified as:

\[
\dot{\dot{s}}_{\Delta,1} = -\Gamma Y^T (q, \dot{q}, \ddot{q}, \dot{q}_r, \ddot{q}_r) s_{\Delta,1}, \quad \dot{b} = \Psi |s_{\Delta,1}|.
\]

(16)

The saturation function defined in eq. (15) can also be used to replace the sign function in eq. (6), thereby eliminating the chattering phenomenon caused by the switching function.

**Remark 2.** Compared to the adaptive controller in [17], the control law presented here has an additional term \( b \text{sign}(s) \). As shown in the stability proof, this term can reject the bounded input disturbances. Moreover, thanks to the second adaptation law in eq. (9), there is no need to have prior knowledge of disturbances; the controller rejects them by adjusting the adaptive gains in the switching term.

**Remark 3.** By choosing the reference trajectories as in eq. (5), the resulting sliding surface \( s \) becomes a Terminal Sliding Manifold (TSM). Note that, for \( \alpha = 1 \), \( a \) is equivalent to the conventional linear sliding surface, and eq. (14) becomes \( \dot{e}_q = -\Lambda e_q \). TSM is widely used in various
applications [25]–[27] because it guarantees the convergence of tracking errors on the sliding manifold in finite-time [27]. This is faster than the asymptotic convergence of the classic linear sliding surface, which is used in Slotine-Li’s adaptive method [19].

B. Experimental Validation Results

The experimental setup, including the soft robotic arm, is shown in fig. 3. It consists of a soft arm with a gripper, a proportional valve manifold, and a motion capture system [28]. The arm’s length is 27 cm and it weighs 276 g in total. The arm has three inflatable chambers in each of its two segments. It is made of silicone elastomer and reinforced with fibers to reduce bloating and increase bending under pressure. The soft gripper and each of the six chambers were actuated independently through an array of proportional valves. The robot configurations were measured in real-time with a motion capture system that consisted of eight infrared cameras. These cameras were mounted around the arm and connected to a laptop that was running a motion capture software. The reflective markers were attached to the robot base and around the tip of each segment. The valve system software. The reflective markers were attached to the robot and connected to a laptop that was running a motion capture with a motion capture system that consisted of eight infrared valves. The soft gripper and each of the six chambers were actuated independently through an array of proportional pressure. The soft gripper and each of the six chambers were actuated independently through an array of proportional valves. The proportional valve array independently pressurizes the chambers. Motion capture cameras measure the curvature of the segments.

we carried out a data acquisition procedure to estimate the dynamic coefficients in \( a_2 \). We collected identification data for nine experiments. In each experiment, we injected a sinusoidal input under pressure into fully pneumatic valves. The amplitudes of the sinusoidal inputs were 0.4 bar, 0.6 bar, and 0.8 bar. The periods were 8 s, 16 s, and 24 s. Given that the factorization of the regressor matrix is \( (Y_1 \ Y_2) (a_1^T \ a_2^T)^T = Ap \), the regression problem can be defined as \( Y_2 a_2 = (A p - Y_1 a_1) \). This can be solved by the pseudo-inverse approach [29]. The estimated values for the stiffness and damping coefficients, along with the mass and length of each segment, are reported in table I.

TABLE I. The physical characteristics of the arm.

| Segment # | \( m \) [g] | \( L \) [cm] | \( k_s \) [N m] | \( k_d \) [N m s] |
|-----------|-------------|-------------|--------------|--------------|
| 1         | 154         | 13.5        | 0.124        | 0.011        |
| 2         | 122         | 13.5        | 0.083        | 0.009        |

After estimating the numerical values for the dynamic coefficients, we validated our model by actuating the robot through pre-defined feed-forward pressures. We then compared the results of the simulated robot with the results of the real robot. To do this, the chambers of each segment were actuated as \( p_i(t) = A \sin^2 \left( \frac{2 \pi t}{T} + i \frac{2 \pi t}{T} \right) \), \( i = 0, 1, 2 \), where \( i \) denotes the index of the chambers for each segment. \( A = 0.4 \) bar and \( T = 16 \) s. As is shown in fig. 4, the evolutions of \( \phi_i \) and \( \theta_i \) over time for the simulated robot closely match with the experimental results. To verify the efficacy of our model, we compared the step response of the dynamic model in which the center of mass is considered to be at the centroid of each segment (as is described in eq. (2)) with the step response of dynamic model in which the center of mass is located at the tip of each segment (as is described in [13], [14]). The amplitude of the step input under pressure was \( p = (0.4 \ 0.3 \ 0.1 \ 1.1 \ 0.8 \ 0.0)^T \) bar. Figure 5 shows the resulting trajectories of the robot’s end-effector and the position error, which is the Euclidean distance between the simulated and measured position. The simulated response of our model closely matches the experimental data, while the dynamic model in [13], [14] has a steady-state error of around 3 cm.

2) Task-space Control: The effectiveness of the proposed model-based control scheme in the task-space was demonstrated on a physical soft robotic arm. To implement our...
controllers did not have information about the payload mass a priori.

A comparison of our controller performance against the benchmark is shown in a video\(^1\) of the experiments. Figure 6 and Figure 7 show the quantitative results that we obtained for the circular and star-shaped trajectories, respectively. The quantitative performance comparisons of our adaptive controller (AC) and the benchmark controller (ID) for various trajectories are reported in Table II. The error is defined as the Euclidean distance between the desired position and the measured position. In Table II, \(M_e\), \(\mu_e\), and \(\sigma_e\) denote the maximum absolute value, average, and standard deviation of the tracking errors, respectively. ‘C’ and ‘S’ stand for the circular and star-shaped trajectories, respectively. The low and high reference velocities correspond to different velocity/acceleration values used in the timing law. When the robot is loaded with a payload and when faster motions are considered, the performance of the inverse dynamics controller degrades significantly. However, our proposed adaptive controller achieves higher robustness in handling payload mass variations and maintains a relatively similar performance at both slow and fast reference trajectories.

In Fig. 4b, we compare our Lagrangian-based modeling approach and the Augmented Rigid Body modeling technique in terms of computational efficiency; namely, the execution time that was required to update the dynamic terms. The Augmented Rigid Body modeling method was implemented via the Drake C++ library [9]. As is illustrated in Fig. 4b, our Lagrangian-based model updated the dynamic terms around one order of magnitude faster than the Augmented Rigid Body model.

V. CONCLUSION AND FUTURE WORK

Our model-based control strategy enables soft continuum robotic arms to track task-space trajectories in a 3D space while carrying an unknown payload. Many parameters, such as stiffness and damping coefficients, must be identified in soft robotic arm models. Moreover, adding a payload to the robot changes the model’s dynamic parameters. Our generalizable adaptive controller can update these parameters online for accurate soft robot control. We hope to use these characteristics in future works to assist soft manipulators in performing dynamically loaded tasks, such as picking and placing objects with unknown loads. As our adaptive control scheme ignores uncertainties in the actuator mapping matrix or the presence of actuator faults, a fault-tolerant control approach deserves investigation for future work.

---

1\text{Video of the real-world experiments including Cartesian trajectory tracking: https://www.youtube.com/watch?v=oa55u8tpqh8}
controller (red line), both of which carried different payloads. In (a), (b), and (c), the maximum reference velocity and acceleration were set to \( \nu_{\text{max}} = 0.05 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \), respectively. In (d), (e), and (f), we use \( \nu_{\text{max}} = 0.11 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \) in the timing law.

Fig. 6. The experimental tracking results for a circular reference trajectory (dotted black line) using our adaptive controller (blue line) and the benchmark controller (red line), both of which carried different payloads. In (a), (b), and (c), the maximum reference velocity and acceleration were set to \( \nu_{\text{max}} = 0.05 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \), respectively. In (d), (e), and (f), we use \( \nu_{\text{max}} = 0.11 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \) in the timing law.

Fig. 7. The experimental tracking results for a star-shaped reference trajectory (dotted black line) using our adaptive controller (blue line) and the benchmark controller (red line), both of which carried different payloads. In (a), (b), and (c), the maximum reference velocity and acceleration were set to \( \nu_{\text{max}} = 0.05 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \), respectively. In (d), (e), and (f), we use \( \nu_{\text{max}} = 0.11 \text{ m s}^{-1} \) and \( a_{\text{max}} = 0.01 \text{ m s}^{-2} \) in the timing law.
