Control of a Soft Robotic Arm Using a Piecewise Universal Joint Model

Zhanchi Wang, Gaotian Wang, Xiaoping Chen, and Nikolaos M. Freris

Abstract—The ‘infinite’ passive degrees of freedom of soft robotic arms render their control especially challenging. In this paper, we leverage a previously developed model, which drawing equivalence of the soft arm to a series of universal joints, to design two closed-loop controllers: a configuration space controller for trajectory tracking and a task space controller for position control of the end effector. Extensive experiments and simulations on a four-segment soft arm attest to substantial improvement in terms of: a) superior tracking accuracy of the configuration space controller and b) reduced settling time and steady-state error of the task space controller. The task space controller is also verified to be effective in the presence of interactions between the soft arm and the environment.

I. INTRODUCTION

Due to the passive compliance and continuous deformation characteristics, soft-bodied robots have demonstrated advantages over their rigid-bodied counterparts in a variety of applications [1], such as soft surgical robots [2], soft arms in interactive tasks [3], and soft artificial limbs [4]. The design of responsive, reliable, and stable controllers is a vital premise for the proliferation of soft robots in real-life applications. However, due to ‘infinite’ underactuated and compliant degrees of freedom (DoFs), controlling a soft arm is an exigent endeavor [5].

With the consideration of the universal approximation ability of robot learning, machine learning methods are often used to solve the modeling and control problems of soft arms with continuous deformation and nonlinear characteristics [6], [7]. However, such methods require collecting large volumes of data for training and are difficult to provide stability guarantees.

There has also been work on developing models and controllers such as finite element method (FEM) [8], discrete Cosserat rod method [9], and discrete Euler–Bernoulli beam method [10]. These methods aim to capture structural properties of the arm, but their applicability is limited to soft robotic systems with simple structures in view of computational cost.

The most widely used simplified model for controlling soft robotic arms is the Piecewise Constant Curvature (PCC) model [11], [12], based on which kinematic and dynamical controllers were developed [13], [14], [15]. More closely related to our approach are methods that draw analogies of soft arms to rigid robots based on the PCC model. The authors in [16] devised an analogue of a planar soft arm under PCC to a discrete joint robot, and resorted to methods for rigid robot control. The method was extended to 3D [17], by modeling a single segment of the arm as a rigid robot with ten joints. Another method of drawing equivalence of a 3D soft arm to a rigid robot and realizing dynamical control under the PCC assumption was demonstrated in [18]. Nevertheless, there are discontinuities and singularities in the commonly employed state representation (direction/angle of bending) of PCC, which may cause instability of PCC-based controllers [19], [20]. To illustrate, when the arm moves continuously in the 3D space, a discontinuity in the state representation will incur inaccurate state estimation which, in turn, will propagate to undesirable changes in actuation. In effect, the PCC model assumes that each segment of the arm is deformed along a circular arc, which is not realistic in practical 3D applications. In our previous work [21], we proposed a simplified modeling method for soft robot arms which capitalizes on a Piecewise Universal Joint (PUJ) principle: it regards each segment of the soft arm as a universal joint, under the less restrictive assumption that it deforms on a plane without twisting. In brief, we established that the PUJ model describes the soft arm more accurately than the PCC model in the presence of external load.

In this paper, we develop two closed-loop controllers for soft robotic arms based on the PUJ model; 1) a controller for trajectory tracking in the configuration space and 2) a controller for position control of the arm tip in the task space. We evaluate the proposed controllers through a series of numerical simulations and experiments on a 3D four-segment soft robotic arm under various scenarios; our experiments corroborate enhanced tracking capability for the former controller and faster response time alongside lower steady-state
error for the latter, compared to baseline methods.

The main contributions of this paper enlist:

- We verify the viability of the PUJ modeling method in the context of dynamical control of the multi-segment soft arm.
- We develop a configuration space controller for a multi-segment soft arm for dynamically tracking desired trajectories; its mean tracking error was measured to be 70\% lower compared to a baseline model-free PID controller.
- We develop a task space controller for driving the end effector to a desired position; its settling time in free space was reduced by 51\% compared to a previously employed simplified model-based controller on the same robot, while the steady-state error under external load was reduced by 58\%.

II. Piecewise Universal Joint Model

In our previous work [21], we cast an analogy of the motion characteristics of a multi-segment soft arm to a series of rigid link mechanisms, namely, universal joints. We presented a Piecewise Universal Joint model which captures the kinematics and dynamics of a soft robotic arm as a serial interconnection of universal joints (a favorable attribute being that a single universal joint was proven sufficient to model one segment of the robot). We proceed to briefly recap the dynamical model developed in [21] for completeness.

For an \( n \)-segment soft arm, we use the coordinate systems at the base of the arm and the ends of each segment, namely \( \{S_0\}, \ldots, \{S_n\} \), to represent the state (see Figure 2(a)). Each segment of the soft arm is represented by a universal joint.

The coordinate system \( \{S_i\} \) is obtained from \( \{S_{i-1}\} \) via four sequential operations in the corresponding coordinate frames: taking the moving frame \( \{S_i\} \) and frame \( \{S_{i-1}\} \) to be initially coincidental, we 1) translate along the z-axis by \( d_i \), 2) rotate around the translated y-axis by \( \theta_i \), 3) rotate around the translated/rotated x-axis by \( \theta_i \), and finally, 4) translate along the translated and twice rotated z-axis by \( d_i \) (see Figure 2(b)).

The parameters \( d_i, \theta_1, \theta_2, \) and \( d_i \) represent the configuration variables for the \( i \)-th segment and the homogeneous transformation matrix, which transforms vectors from coordinate frame \( \{S_i\} \) to coordinate frame \( \{S_{i-1}\} \), is given by [21]:

\[
T_{i-1} = \begin{bmatrix}
    c_{\theta_i} & s_{\theta_i} & 0 & d_i \\
    -s_{\theta_i} & c_{\theta_i} & 0 & 0 \\
    0 & 0 & 1 & 0 \\
\end{bmatrix}, \tag{1}
\]

where \( c_{\theta_i}, s_{\theta_i} \) respectively abbreviate \( \cos \theta_i, \sin \theta_i \). The transformation matrix is usually measured from sensors such as Motion Capture Systems (MCS), and can be used to extract the configuration variables as detailed in [21].

The kinematic structure of a four-segment soft arm is shown in Figure 2(c): four universal joints are connected as an open chain. The \( i \)-th segment is described by a sequence of four joints (PRRP), where \( R \) stands for revolute and \( P \) for prismatic. The kinematics and dynamics of rigid body open chains can be expressed using screw theory; we omit the details and refer the reader to [22, Chapter 3]. For soft arms with ignorable extensibility, one can approximate the prismatic joints in the PUJ model as links with fixed lengths, so that only two revolute joint variables \( \theta_1, \theta_2 \) are needed to represent the state of a segment as universal joint. We adopt this assumption in what follows, as it has been shown effective in our experiments.

The dynamical model for the soft arm was derived from a rigid-body analogue [21]. We take the viscoelasticity of the soft arm into consideration by adding two linear terms \( \mathbf{K} \mathbf{q} \) and \( \mathbf{D} \mathbf{q} \) in the dynamics:

\[
\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{K}\mathbf{q} + \mathbf{D}\dot{\mathbf{q}} = \mathbf{\tau}, \tag{2}
\]

where \( \mathbf{q} := (\theta_1, \theta_2) \in \mathbb{R}^{2n} \) is the state of the robot, \( \mathbf{M}(\mathbf{q}) \in \mathbb{R}^{2n\times2n}, \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{2n\times2n}, \mathbf{G}(\mathbf{q}) \in \mathbb{R}^{2n} \) respectively capture inertia, Coriolis/centrifugal, and gravity, \( \mathbf{K}, \mathbf{D} \in \mathbb{R}^{2n\times2n} \) are diagonal stiffness and damping matrices, and \( \mathbf{\tau} \in \mathbb{R}^{2n} \) is is the joint torque vector. The procedure to estimate \( \mathbf{K} \) and \( \mathbf{D} \) is detailed in [21].

In this paper, we invoke the Newton-Euler recursive method with screw representation to learn the dynamical model [22], due to its computational efficiency that makes it suitable for a soft robotic arm with many degrees of freedom.
multiple segments). The Newton-Euler recursive method can be used for the forward and inverse dynamics of rigid body open chains. The method operates by iteratively calculating the rigid body motion in space through the transmission of speed, acceleration, and force between adjacent rigid links. We provide a brief introduction of this process in the next section and refer the reader to [22, Chapter 8] for more details.

III. CONTROL

In this section, we introduce two closed-loop controllers. The first targets trajectory tracking in the configuration space, while the second serves to control the position in the task space.

A. Configuration Space Controller

The vector \( \mathbf{q} = (\theta_1^1, \theta_2^1, \ldots, \theta_n^1, \theta_1^2, \ldots, \theta_n^2)^\top \in \mathbb{R}^{2n} \) is defined as the state in the configuration space; vectors \( \dot{\mathbf{q}}, \ddot{\mathbf{q}} \in \mathbb{R}^{2n} \) respectively correspond to angular velocity and angular acceleration. Given a reference signal \( \dot{\mathbf{q}}_r, \ddot{\mathbf{q}}_r \in \mathbb{R}^{2n} \), the closed-loop dynamic tracking controller in the configuration space takes the form

\[
\mathbf{u} = f_2(\dot{\mathbf{q}}_r, \ddot{\mathbf{q}}_r, \dot{\mathbf{q}}, \ddot{\mathbf{q}}),
\]

in which \( \dot{\mathbf{q}}, \ddot{\mathbf{q}} \) are estimates of the state and its derivative obtained by a state estimation module, as presented in Section IV.

The control dynamics for trajectory tracking in the configuration space are given by:

\[
\mathbf{\tau} = K_P \mathbf{e} + K_D \mathbf{e} + K_I \int \mathbf{e} + M(\dot{\mathbf{q}})\ddot{\mathbf{q}} + C(\mathbf{\dot{q}}, \mathbf{\dot{\dot{q}}}) + G(\mathbf{\dot{q}}) + K_G \mathbf{q} + \mathbf{D} \dddot{\mathbf{q}},
\]

where \( \mathbf{e} := \dddot{\mathbf{q}} - \dddot{\mathbf{q}}_r \) is the tracking error, and \( K_P, K_D, K_I \) are feedback proportional–integral–derivative (PID) gains. The gains are tuned through experiments and set equal for all eight joints of the robot. The proposed model-based closed-loop controller contains a mixture of feedback and feedforward terms: the terms \( K_G \mathbf{q} + \mathbf{D} \dddot{\mathbf{q}} \) are feedforward, \( G(\mathbf{\dot{q}}) \) is a feedback term for compensating gravity, while the other terms contain both feedforward and feedback components. The integral term is used to eliminate the influence of modeling errors. The block diagram of the proposed configuration space controller is shown in Figure 3.

Calculating the matrix mappings \( \mathbf{M}, \mathbf{C}, \) and \( \mathbf{G} \) in (3) involves solving the inverse dynamics via the Newton-Euler recursive method (Algorithm 1) [22], which in our case also incorporates the elastic and damping terms (step 8). The method consists of two iterative processes, namely forward and backward iterations. The forward iteration calculates the position (step 2), velocity (step 3), and acceleration (step 4) of each link from 1 to \( n \), and the backward iteration calculates the force/torque on each link (step 7), and the force/torque for each joint (step 8), from joint \( n \) to 1. In Algorithm 1, \( \mathcal{A}_i \) is the screw axis for joint \( i \) expressed in the \( i \)-th link frame, \( \mathbf{H}_{i-1} \) is the homogenous pose of the \( i \)-th link expressed in the \((i-1)\)-th link frame, and \( \mathcal{G}_i \) is the spatial inertia matrix of link \( i \). The twist and wrench are represented by \( \mathcal{V}_i, \mathcal{F}_i \), respectively, while \( \mathbf{A} \) denotes the adjoint representation of a homogeneous transformation matrix and \( \mathbf{d} \) denotes the Lie bracket of a twist. We refer the reader to [22, Chapter 8] for more details. After computing \( \mathbf{\tau} \) using Algorithm 1, we proceed to compute the matrices \( \mathbf{M}, \mathbf{C}, \mathbf{G} \); we illustrate for \( \mathbf{M} \), the other cases being analogous. The matrix is built column by column by calling Algorithm 1 \( 2n \) times. In the \( i \)-th call, \( \dot{\mathbf{q}}_i, \mathbf{K}_i, \mathbf{D} \), and the contribution of gravity are set to zero, while \( \dddot{\mathbf{q}}_i \) is a vector with all entries equal to zero except for the \( i \)-th entry equal to one [22].

B. Task Space Controller

Motion control in the task space is indispensable for the soft arm to effectively perform tasks. Given a target point \( \mathbf{x} \), the closed-loop controller in the task space takes the form

\[
\mathbf{u} = f_3(\mathbf{x}, \mathbf{\dddot{x}}),
\]

where \( \mathbf{F}_{ip} = \mathbf{K}_T \left( K_T' \mathbf{e}' + K_T'' \mathbf{e}'' + K_T''' \int \mathbf{e}''' \right), \)

where \( \mathbf{J} \) denotes the Jacobian, \( \mathbf{K}_T \) is a diagonal stiffness matrix, and \( \mathbf{e}' := \dddot{\mathbf{x}} - \dddot{\mathbf{x}} \). This design corresponds to virtually placing a spring with a stiffness of \( \mathbf{K}_T \) between the current position \( \mathbf{x} \) of the end effector and the target position \( \dddot{\mathbf{x}} \). We choose a value of 5Nm for each diagonal entry of \( \mathbf{K}_T \). The block diagram

![Fig. 3. Block diagram of the proposed trajectory tracking controller in the configuration space. \( \mathbf{K} \) and \( \mathbf{D} \) are feedforward terms. \( \mathbf{M}, \mathbf{C}, \) and \( \mathbf{G} \) contain model information. \( \mathbf{A} \) is a conversion mapping between the torque of the PUM model and the air pressure of the soft arm. The state of the soft arm is estimated using an MCS.](image-url)
of the proposed task space controller is shown in Figure 4. The conversion of virtual force to torque, i.e., $J^\top F_{np}$, is implicitly calculated using the inverse dynamics (Algorithm 2) with $\dot{q}, \ddot{q}$ equal to zero. The PID gains are tuned through experiments in the task space.

IV. STATE ESTIMATION

To implement the closed-loop controller in the configuration space proposed in Section III.A, we need to obtain an estimate of the state of the arm $\mathbf{q}$ and its derivative $\dot{\mathbf{q}}$. This can be accomplished by measuring the posture of the reference coordinate systems $\{S_0\}, \ldots, \{S_n\}$ (Figure 2) which is typically performed by cameras. Angles $\theta_1$ and $\theta_2$ can be derived by converting the measured rotation matrix to the form of Z-Y-X Euler angles $(\alpha, \beta, \gamma)^\top$ whence, for a universal joint, $\alpha = 0$, $\beta = \theta_1$, and $\gamma = \theta_2$. This procedure can be done with existing software packages (e.g., rotm2eul in Matlab) and is not detailed here. The velocity vector $\mathbf{\dot{q}}$ is simply calculated via the discrete derivative $\ddot{\mathbf{q}} = \frac{\mathbf{q}(t) - \mathbf{q}(t-dt)}{dt}$ where $dt = 0.005s$ was chosen for simulations and $dt = 0.05s$ for experiments on the real arm. The time granularity of discretization in the real experiments was chosen according to the actual response frequency of the pneumatic soft arm. We use the position of the end effector expressed in the base coordinate system $\{S_0\}$ as the estimation of $\dot{x}$ for the task space controller proposed in Section III.B.

V. EXPERIMENTAL SETUP

A. SOFT ROBOTIC ARM

The proposed dynamical model and control algorithms are validated on a 3D soft arm with four segments. In this section, we synthesize the pneumatic soft robotic arm system and provide the results of system identification.

The design, fabrication, and simulation analysis methods of the arm are detailed in [24], and a block diagram of the hardware system is demonstrated in Figure 5. The air pump generates air at a maximum pressure of 0.7 MPa, which is regulated to 0.4 MPa by the air preparation equipment (AC30C, SMC), and is then provided to the proportional valve array (ITV0030, SMC). The soft robotic arm consists of four segments and is actuated by a total of 16 proportional valves. Control signals generated by the computer are transferred to proportional valves via an analog output card (PCI-1724U, Advantech). We placed reflective markers on both ends of each segment, and use an MCS (Prime 13, OptiTrack) to obtain the pose of the markers. The design parameters of the arm are listed in Table I.

In order to apply the proposed controllers (3) and (4) to the pneumatic soft robotic arm, we devise a conversion mapping between the torque of the equivalent model and the air pressure of the soft arm. We use the conversion proposed in our previous work [21], where the air pressure and the area of the airbags are used to calculate the force and torque with a simple law of $F = ps$, $\tau = Fl$, where $p$ represents pressure, $s$ area, and $l$ length.

B. SYSTEM IDENTIFICATION

The stiffness matrix $\mathbf{K}$ and the damping matrix $\mathbf{D}$ are obtained from experimentally collected data by using linear regression as described in [21]; equation (2) is rewritten as

$$\mathbf{Kq} + \mathbf{Dq} = \mathbf{r} - \mathbf{M(q)q} - \mathbf{C(q)q} - \mathbf{G(q)},$$

where the left side contains the terms to be identified and the right side can be computed from sensor data, along with the previously calculated matrices $\mathbf{M}, \mathbf{C}, \mathbf{G}$. For the soft arm used in this paper, the identification results yield:

$$\mathbf{K} = \text{diag}(3.90, 3.58, 3.08, 2.71, 2.25, 2.20, 0.81, 0.88)$$

$$\mathbf{D} = \text{diag}(0.25, 0.19, 0.16, 0.10, 0.06, 0.06, 0.03, 0.02).$$

VI. EXPERIMENTS

In order to evaluate the performance of the two proposed controllers, we perform two groups of experiments pertaining to trajectory tracking in the configuration space and position control in the task space.

A. TRAJECTORY TRACKING

We test the performance of the proposed configuration space controller for sinusoidal reference trajectories of dif-
different frequencies:
\[ \dot{\theta}_1^i, \dot{\theta}_2^i, \dot{\theta}_3^i, \dot{\theta}_4^i := 0.2 \sin \left( \frac{\pi}{10} t \right), \]
\[ \dot{\theta}_1^4, \dot{\theta}_2^4, \dot{\theta}_3^4, \dot{\theta}_4^4 := 0.3 \sin \left( \frac{\pi}{20} t \right), \]
where \( \theta_1^i, \theta_2^i, \theta_3^i, \theta_4^i \) are the state variables of the first revolute joint of each universal joint and \( \theta_1^4, \theta_2^4, \theta_3^4, \theta_4^4 \) of the second. The reference trajectories in the configuration space makes the arm move in the task space in the manner shown in Figure 6(e). In this experiment, we also compare the performance of the controller with a (model-free) PID controller:

\[
\tau = K_p^c e^c + K_d^c e^c + K_i^c \int e^c dt,
\]
where \( K_p^c, K_d^c, K_i^c \) are pre-tuned PID gains. In addition to experiments on the soft arm, we have also conducted the simulations in Matlab and used the model corresponding to the arm with parameters learned as explained in Section V.

The results of trajectory tracking experiments are shown in Figure 6 and Table II. Our proposed model-based controller outperforms the baseline model-free PID controller in terms of both higher accuracy and lower response delay. In the simulation experiments, our proposed controller achieves a negligible average joint error of 0.0003 rad, while the error of the baseline controller is 0.0112 rad. In the real experiments, our proposed controller achieves an average joint error of 0.0244 rad, versus an 0.0807 rad of the baseline controller, for a reduction of 70%. Note that the real experiments consider a much lower discretization rate (20 Hz) compared to simulations (200 Hz). In summary, our experiments unravel that the model-free PID controller fails to effectively track the given trajectory. We observed that the closer the segment to the root of the arm, the greater the tracking error. On the contrary, the controller proposed in the paper utilizes the information of the model to achieve lower tracking errors for all segments of the arm.

**TABLE II**

| Segment | Simulation error [rad] | Experiment error [rad] |
|---------|------------------------|------------------------|
| 1 (base) | 0.0182 | 0.0002 | 0.1160 | 0.0433 |
| 2 | 0.0133 | 0.0003 | 0.9998 | 0.0354 |
| 3 | 0.0082 | 0.0003 | 0.0697 | 0.0143 |
| 4 (tip) | 0.0051 | 0.0003 | 0.0371 | 0.0044 |

Mean absolute error along the trajectory

**B. Position Control**

We demonstrate the performance of the proposed task space controller by a set of position control experiments in the 3D free space. We randomly chose ten target positions in the workspace of the soft arm, and use the controller to drive the arm to reach the targets from a fixed starting point (resting position). The simplified Jacobian model presented in our previous work [3] is used to establish a baseline method for comparisons.

The task space controller proposed in this article can realize the position control of the soft arm in the 3D space much faster than the baseline method, as shown in Figure 7(a). Table III gives the mean settling time and standard deviation over ten trials. Compared with our previous work, the controller proposed in this paper can reach the target at the same accuracy using much less time (over 50%). Taking the error drop to 10% of the initial error as the performance indicator (10% settling time), the mean settling time of the proposed controller is 3.34 s versus 6.88 s, see Table III. At the same time, our experimental results illustrate that the proposed controller achieves similar performance for all points chosen randomly with initial position errors ranging 300 - 500 mm (outer layer of the workspace). This is not
the case for the baseline controller (the farther the target is the larger the settling time, see Figure 7(a)). This attests to faster response of our proposed controller, which constitutes a favorable feature for further applications of the soft arm.

TABLE III

| Controllers | Mean settling time [s] | Standard deviation |
|-------------|------------------------|--------------------|
| Ours        | 3.34                   | 0.76               |
| Baseline    | 6.88                   | 1.50               |

10% settling time

In order to demonstrate the ability of the task space controller proposed in this paper to deal with external forces, we conducted position control experiments with load. We randomly select a target in the 3D space, and used the controller to drive the arm towards the target from a fixed starting point (resting position) with different loads of 50g, 100g, 200g, 500g on the tip. The controller proposed in [3] was again used as baseline.

Our experiments illustrate a substantial improvement in response time compared to the baseline controller, as shown in Figure 7(b). The mean steady-state error of the proposed controller is 6.75mm, compared to 16.07mm for the baseline (Table IV). It is noteworthy that the performance is robust with the load weight, which further corroborates the applicability of the proposed controller in interactive tasks.

We further demonstrate the capability of the proposed controller in a simple practical task (Figure 8). We manually specify the position of a drawer handle so that the arm can reach and grab it. Then, we select a target point and drive the arm to reach the target and output a force that can open the drawer.

VII. CONCLUSIONS

We proposed two controllers for 3D multi-segment soft robotic arms: one for trajectory tracking in the configuration space and one for position control in the task space. Both controllers are implemented based on the PUJ model proposed in our previous work [21]; our experiments support the viability of the model in the context of dynamical control of a soft arm. Both controllers have been experimentally affirmed to be more responsive and accurate than baseline methods. The task space controller is stiffness-based: in addition to motion control in the task space, our preliminary results support that it is also expected to be effective in the presence of interactions between the soft arm and the environment. In future work, we will further explore the performance of the PUJ-based controllers for interactive tasks.

ACKNOWLEDGMENT

The authors would like to thank Hao Jiang, Yusong Jin, and Yinghao Gan for fruitful discussions.
