Decentralized Estimation And Control Of A Soft-robotic Arm Using Linearized Beam Model

Sachin Shivakumar, Daniel Aukes, Spring Berman, Ximin He, Rebecca E. Fisher, Hamidreza Marvi and Matthew M. Peet

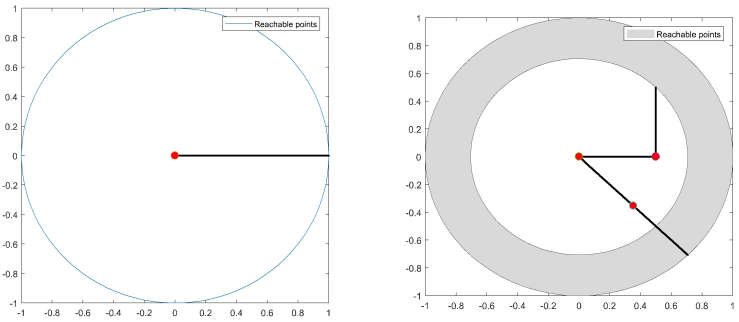
Abstract In this chapter, we use PDE models to design decentralized estimation and control laws for a segmented octopus arm. The dynamics of the soft-robot arm are formulated as a nonlinear PDE, which are then linearized about setpoints to obtain a linear PDE representation similar to linear Euler-Bernoulli beam equations. We use this linearized PDE model to design infinite-dimensional control and estimation laws. The optimal controllers and observers are then discretized during the implementation phase, to perform operations such as shape tracking. We show that the discretized observer or controller can be implemented in a manner that allows decentralized operation in the robot arm.

1 Introduction

Consider a rigid robotic manipulator with a single segment of fixed length r allowed to rotate about the origin in a 2D plane. The tip of the segment can reach the points on the circle centered at the origin with radius r as shown in Fig. 1a. Note that there does not exist any configuration of the segment in which the tip can be inside or outside this circle. As the segment is rigid, the reachable set of the tip of the segment is limited. However, if the robot was made of two segments of length $\frac{1}{2}r$ each connected by a pin joint, as shown in Fig. 1b, with a constraint that the joint angle be greater than 90° then the tip of the robot can reach every point whose distance is between $\frac{1}{\sqrt{2}}r$ and r. Increasing the number of segments further leads to an increase in the configuration space and the points which the tip can reach. A continuum robotic arm is an arm with an infinite number of infinitesimally small segments. The tip of a

Sachin Shivakumar
Arizona State University, e-mail: sshivak8@asu.edu

Matthew M. Peet
Arizona State University, e-mail: mpeet@asu.edu



(a) Robot with one segment

(b) Robot with two segments

Fig. 1: Figure on the left shows the locations reachable by the tip of a robot arm with one segment allowed to rotate about the origin. On the right, the robot arm is made of two segments and can reach all the points in the shaded region.

soft continuum robot can reach every point in the circle of radius r. In addition, the added degrees of freedom in a continuum robot arm allows the robot to take more difficult geometric configurations. Conventionally, robots built with highly flexible components are called soft-robots. Soft robots can be used to perform tasks that require complex motions and high precision, for instance, soft robots can be used in medical surgeries [3], autonomous underwater exploration [6] and manufacturing [16]. To control or observe a soft-robot arm, we will obtain a linear model for the dynamics. Next, we will use a newly developed computational framework to design H_∞-optimal controllers and observers. Finally, we will show how the controller or observer can be implemented in a decentralized manner.

Modeling of a soft-robot Several control-oriented models have been developed for soft-robots which approximate the soft-robot as a series of segments governed by the kinematics of rigid or deformable objects [9, 5]. However, approximating a soft-robot as a series of rigid arm-segments requires us to use a large number of segments to accurately represent large deformations. Furthermore, control of these models in a decentralized manner is computationally intractable. On the other hand, modeling the soft-robot as a single continuously deformable body with large deformations leads to nonlinear stress-strain relationships which depend on spatial derivatives of the displacement of the arm - implying that the dynamics of the system are governed by a nonlinear function of the spatial derivatives of the distributed state, i.e. a nonlinear PDE. Unfortunately, most control of nonlinear PDEs is suboptimal and ad hoc. For this reason, we propose to linearize the nonlinear dynamics about a collection of setpoints. A setpoint, in case of PDEs, can be any pre-specified state which satisfies the PDE and the boundary conditions.

There is no general method that can be used to linearize PDEs that have nonlinear terms. However, in [1] a method was given for the special case when the non-linear

terms are polynomials in the states and their spatial derivatives. In this case, the original nonlinear PDE system is approximated by a system of linear PDEs about a collection of setpoints. This is the approach taken in section 2.2 of this manuscript.

Observation and control of a linear soft-robot model Optimal Control of linear PDEs with discretization is challenging. For this problem, we turn to the recently developed Partial Integral Equation (PIE) framework. Because our PDE only has a single spatial dimension, it can be represented in this framework as

$$\mathcal{T}\dot{\mathbf{x}}(t) = \mathcal{A}\mathbf{x}(t) + \mathcal{B}_1 w(t) + \mathcal{B}_2 u(t)$$

where $\mathcal{T}, \mathcal{A}, \mathcal{B}_i$ are Partial Integral (PI) operators of the form

$$\left(\mathcal{P}_{\{N_i\}}\mathbf{y}\right)(s) := N_0(s)\mathbf{y}(s) + \int_a^s N_1(s,\theta)\mathbf{y}(\theta)d\theta + \int_s^b N_2(s,\theta)\mathbf{y}(\theta)d\theta.$$

Optimal control and estimation of PIEs can be performed using a generalization of Linear Matrix Inequalities (LMIs). We refer to this framework as the Linear PI Inequality (LPI) framework and once our Linearized PDEs have been converted to PIEs, in section 3.2, the PIETOOLS toolbox is used to find optimal continuum estimators and feedback controllers.

Decentralized implementation of the controller Continuum observers and feedback controllers are naturally distributed along the robot arm. Implementing such distributed controllers and observers using large numbers of embedded sensors and actuators may be challenging unless some mechanism can be found to decentralize the calculation of feedback gains and error corrections. Specifically, unless there is some decentralization of the controller (or observer), then coordination of n actuators (or sensors) would require $n(n-1)$ interconnections. The wiring alone required for such an implementation has the potential to exceed the mass of the robot arm itself. Furthermore, communication between distant nodes introduces potentially destabilizing delay into the system. Unfortunately, the design of a decentralized H_∞-optimal controller or estimator is known to be NP-hard, except in the case of Quadratically Invariant systems [7].

Although the design of a finite-dimensional H_∞-optimal decentralized controller or observer is an NP-hard non-convex optimization problem, in Section 4, we show that controllers and observers of the PI form admit a naturally decentralized implementation. Specifically, the number of interconnections in this implementation scales as $2n$ (as opposed to $n(n-1)$).

A brief summary of main results The main contribution of this work is to apply the techniques developed to design optimal observers and controllers for linear PDE to a soft robot manipulator. We model the dynamics of the robot arm as a nonlinear PDE and discuss one of the strategies to linearize the nonlinear PDE. Then the linear PDE is converted to a linear PIE. The design of a distributed H_∞-optimal observer or controller for the PIE is posed as a convex-optimization problem with LPI constraints. The optimization problem is then solved using PIETOOLS. Furthermore, a decentralized implementation of these observers and controllers is proposed.

Note on Continuum Controllers and Observers The methods discussed in this chapter aid in the design of the optimal controllers and observers without approximating the PDE by an ODE. The approximation is done at the implementation stage after the design of the optimal controller or observer. We will use a finite number of equispaced actuators and sensors. However other non-equispaced strategies, e.g. [4], can be implemented in a decentralized manner due to the form of PI operators. The key point is that the design process needs to be performed only once and different discretization strategies for the observer can be used — which is not case if the PDE is discretized prior to the design stage. Furthermore, models which approximate PDEs by a finite-dimensional system before finding a controller or observer, e.g. hyper-redundant continuum robotic arm [18], often result in a controller that requires significant computational effort in motion planning - thus making it impractical in real-time applications.

2 Modelling Of The Soft-robot Arm

In this section, we introduce the nonlinear PDE model which describes the dynamics of the soft-robot followed by linearization of the system to obtain a linear PDE model for the soft-robot arm.

2.1 Nonlinear Euler Beam Model

In this section, we introduce a nonlinear model of the soft-robot arm. The robot is assumed to move in a 2D plane. The deformations are defined by the axial and transverse coordinates. The derivation of this model can be found in [10].

The dynamics of this system are given by

$$\partial_x \left(EAe(x,t)cos(\theta(x,t))\right) + \partial_x \left(\frac{\partial_x(EI\partial_x\theta(x,t))sin(\theta(x,t))}{1+e(x,t)}\right) = \rho\ddot{u}(x,t) + \partial_x \left(\frac{J\ddot{\theta}(x,t)sin(\theta(x,t))}{1+e(x,t)}\right),$$

$$\partial_x(EAe(x,t)sin(\theta(x,t))) - \partial_x \left(\frac{\partial_x(EI\partial_x\theta(x,t))cos(\theta(x,t))}{1+e(x,t)}\right) + f(x,t) = \rho\ddot{w}(x,t) + \partial_x \left(\frac{J\ddot{\theta}(x,t)cos(\theta(x,t))}{1+e(x,t)}\right),$$

where u is the axial deformation, w is the transverse deformation, E is the Young's modulus, A is the area of cross-section, I is the moment of inertia about the axis perpendicular to the plane of motion, J is rotational moment of inertia about axial

direction, ρ is the mass per unit length, e is the axial stretch and rotation of the cross-section θ.

For this study, we assume rotary inertia J is small and the arm is isotropic with uniform cross-section. These assumptions are not necessary, however, and can be relaxed if necessary. Next, we can simplify the equations to obtain

$$\begin{aligned} &EA\partial_x(e(x,t)cos(\theta(x,t))) + EI\partial_x\left(\frac{\partial_x^2\theta(x,t)sin(\theta(x,t))}{1+e(x,t)}\right) = \rho\ddot{u}(x,t) \\ &EA\partial_x(e(x,t)sin(\theta(x,t))) - EI\partial_x\left(\frac{\partial_x^2\theta(x,t)cos(\theta(x,t))}{1+e(x,t)}\right) + f(x,t) = \rho\ddot{w}(x,t). \end{aligned} \tag{1}$$

Now, we replace the geometric relations $tan(\theta) = \frac{\partial_x w}{1+\partial_x u}$ and $e = ((1+\partial_x u)^2 + \partial_x w^2)^{\frac{1}{2}} - 1$ in the Eq. (1). Taylors' series expansion is then used to represent the nonlinear terms as polynomials and terms with degree higher than 2 are truncated. This yields a nonlinear PDE with 2^{nd} order derivative in time t and 4^{th} order derivatives in the spatial variable, x.

$$\begin{aligned} \rho\ddot{u}(x,t) - EA\partial_x^2 u(x,t) &= EA\partial_x w(x,t)\partial_x^2 w(x,t) \\ &\quad + EI(\partial_x^2 w(x,t)\partial_x^3 w(x,t) + \partial_x w(x,t)\partial_x^4 w(x,t)), \\ \rho\ddot{w}(x,t) + EI\partial_x^4 w(x,t) &= EA(\partial_x^2 u(x,t)\partial_x w(x,t) + \partial_x u(x,t)\partial_x^2 w(x,t)) \\ &\quad + EI(\partial_x^2 u(x,t)\partial_x^2 w(x,t) + \partial_x u(x,t)\partial_x^3 w(x,t)) \\ &\quad + EI(\partial_x^3(\partial_x u(x,t)\partial_x w(x,t))) + f(x,t). \end{aligned} \tag{2}$$

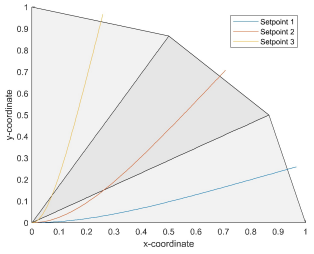
We require the solutions to Eq. (2) $u(.,t), w(.,t) \in W^{2,4}[0,L]$ to satisfy the boundary conditions corresponding to attachment of the arm at the mantle.

$$\begin{aligned} &\tilde{u}(0,t) = 0, \quad \partial_x\tilde{u}(0,t) = 0, \quad \partial_x^2\tilde{u}(L,t) = 0, \quad \partial_x^3\tilde{u}(L,t) = 0, \\ &\tilde{w}(0,t) = 0, \quad \partial_x\tilde{w}(0,t) = 0, \quad \partial_x^2\tilde{w}(L,t) = 0, \quad \partial_x^3\tilde{w}(L,t) = 0. \end{aligned}$$

These boundary conditions imply the arm is fixed at one end and free at the other. Equation (2) has polynomial nonlinearity, i.e. nonlinear terms that are products of functions and derivatives of functions.

2.2 Setpoint Linearization Of The Dynamics Of Soft-robot Arm

Typically, for nonlinear ODE systems, Jacobian linearization is used to linearize the system. Unfortunately, Jacobian Linearization cannot be directly used for nonlinear PDEs. We can, however, use a similar approach [1] by considering deviations from a pre-specified solution (setpoint) to obtain a linear PDE that approximates the behavior of the system in the neighbourhood of the chosen setpoint. Consider a



Setpoint 1:

$$u_e(x) = 0, \quad w_e(x) = 0.55x^2 - 0.38x^3 + 0.09x^4$$

Setpoint 2:

$$u_e(x) = 0, \quad w_e(x) = 2.8x^2 - 2.6x^3 + 0.94x^4$$

Setpoint 3:

$$u_e(x) = 0, \quad w_e(x) = 28.8x^2 - 74.2x^3 + 71.7x^4$$

Fig. 2: Figure on the left shows three functions chosen as setpoints. The tip of the arm is chosen to be the scheduling variable. The conic sections in gray show the operation zone, i.e. if the tip of the arm is in the cone that shares an edge with x-axis, then the first setpoint is chosen. On the right, we provide the polynomials for the setpoint.

nonlinear PDE,

$$\dot{\mathbf{x}}(t) = \mathcal{A}(\mathbf{x}(t), u(t))$$

where $\mathcal{A} : X \times U \to X$ is a nonlinear function that is locally lipschitz. A solution, $\mathbf{x}$ near the setpoint $\mathbf{x}_e$ can be written as $\mathbf{x} = \mathbf{x}_e + \epsilon\mathbf{w}$. Since the setpoint is a solution to the PDE, $\dot{\mathbf{x}}_e(t) = \mathcal{A}(\mathbf{x}_e(t), u(t))$. Then, we can find

$$\epsilon\dot{\mathbf{w}}(t) = \mathcal{A}(\mathbf{x}_e(t) + \epsilon\mathbf{w}(t), u(t)) - \mathcal{A}(\mathbf{x_e}(t), u(t)).$$

We expand $\mathcal{A}(\mathbf{x}_e(t) + \epsilon\mathbf{w}(t), u(t))$ using a Taylors' Series Expansion. The linear approximation can then be obtained by truncating the terms with degree of ϵ greater than 1.

Applying this technique to the nonlinear PDE in Eq.(2) with setpoint u_e and w_e, we obtain the following linear PDE where the independent variables are here omitted for brevity.

$$\begin{aligned}
\rho\ddot{\tilde{u}} &= EA\partial_x^2\tilde{u} + (EA\partial_x^2 w_e + EI\partial_x^4 w_e)\partial_x\tilde{w} + (EA\partial_x w_e + EI\partial_x^3 w_e)\partial_x^2\tilde{w} \\
&\quad + EI\partial_x^2 w_e\partial_x^3\tilde{w} + EI\partial_x w_e\partial_x^4\tilde{w} \\
\rho\ddot{\tilde{w}} &= (EA\partial_x^2 u_e + EI\partial_x^4 u_e)\partial_x\tilde{w} + (EA\partial_x u_e + EI\partial_x^2 u_e + 3EI\partial_x^3 u_e)\partial_x^2\tilde{w} \\
&\quad + (EI\partial_x u_e + 3EI\partial_x^3 u_e)\partial_x^3\tilde{w} + (-EI + EI\partial_x u_e)\partial_x^4\tilde{w} \\
&\quad + (EA\partial_x^2 w_e + EI\partial_x^3 w_e + EI\partial_x^4 w_e)\partial_x\tilde{u} + (EA\partial_x w_e + EI\partial_x^2 w_e + 3EI\partial_x^3 w_e)\partial_x^2\tilde{u} \\
&\quad + 3EI\partial_x^2 w_e\partial_x^3\tilde{u} + EI\partial_x w_e\partial_x^4\tilde{u} + f.
\end{aligned} \tag{3}$$

Finally, we define new states for the system Eq.(3) as $\mathbf{x} = \left[\tilde{u}\ \dot{\tilde{u}}\ \tilde{w}\ \dot{\tilde{w}}\right]^T$. The PDE can now be written as

$$\dot{\mathbf{x}}(t) = \sum_{j=0}^{4} A_j(s)\partial_s^j \mathbf{x}(t) + B_1(s)f(t), \tag{4}$$

where

$$A_0(x) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_1(x) = \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & (EA\partial_x^2 w_e + EI\partial_x^4 w_e) & 0 \\ 0 & 0 & 0 & 0 \\ (EA\partial_x^2 w_e + EI\partial_x^3 w_e + EI\partial_x^4 w_e) & 0 & (EA\partial_x^2 u_e + EI\partial_x^4 u_e) & 0 \end{bmatrix},$$

$$A_2(x) = \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ EA & 0 & (EA\partial_x w_e + EI\partial_x^3 w_e) & 0 \\ 0 & 0 & 0 & 0 \\ (EA\partial_x w_e + EI\partial_x^2 w_e + 3EI\partial_x^3 w_e) & 0 & (EA\partial_x u_e + EI\partial_x^2 u_e + 3EI\partial_x^3 u_e) & 0 \end{bmatrix},$$

$$A_3(x) = \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & EI\partial_x^2 w_e & 0 \\ 0 & 0 & 0 & 0 \\ 3EI\partial_x^2 w_e & 0 & (EI\partial_x u_e + 3EI\partial_x^2 u_e) & 0 \end{bmatrix},$$

$$A_4(x) = \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & EI\partial_x w_e & 0 \\ 0 & 0 & 0 & 0 \\ EI\partial_x w_e & 0 & (-EI + EI\partial_x u_e) & 0 \end{bmatrix}, \qquad B_1(x) = \frac{1}{\rho}\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{5}$$

In the next section, we present a way to convert a linear PDE of this format to a PIE representation using the A_i's and B_i's defined above. The A_i's and B_i' for our model are obtained by substituting the setpoint functions u_e and w_e illustrated in Figure 2.

3 Estimation and Control Of Linear PDEs

In this section, we present the PIE representation of a PDE. This representation has the advantage that it is defined by bounded linear operators (no spatial derivatives). Furthermore, H_∞-optimal observers and feedback controllers can be obtained for systems in the PIE framework using newly-developed computational tools such as PIETOOLS [15].

3.1 Partial Integral Operators

A PI operator is a parameterized linear mapping between infinite-dimensional spaces $\mathbb{R}^m \times L_2^n \to \mathbb{R}^p \times L_2^q$. We define two classes of PI-operators, 3-PI and 4-PI. As the nomenclature suggests, 3-PI operators, denoted as $\mathcal{P}_{\{N_i\}} : L_2^m[a,b] \to L_2^n[a,b]$, are parameterized by 3 matrix-valued functions $N_0 : [a,b] \to \mathbb{R}^{n\times m}$ and $N_1, N_2 : [a,b]\times[a,b] \to \mathbb{R}^{m\times n}$. 3-PI operators are bounded linear maps between the normed spaces $L_2^m[a,b]$ and $L_2^n[a,b]$, endowed with standard L_2 inner product.

$$\left(\mathcal{P}_{\{N_i\}}\mathbf{y}\right)(s) := N_0(s)\mathbf{y}(s) + \int_a^s N_1(s,\theta)\mathbf{y}(\theta)d\theta + \int_s^b N_2(s,\theta)\mathbf{y}(\theta)d\theta. \tag{6}$$

Similarly, 4-PI operators, are bounded linear operators between $\mathbb{R}^m \times L_2^n[a,b]$ and $\mathbb{R}^p \times L_2^q[a,b]$ and are parameterized by the matrix P, the matrix-valued functions Q_1, Q_1 and the 3-PI operator $\mathcal{P}_{\{N_i\}}$ where $P : \mathbb{R}^m \to \mathbb{R}^p$, $Q_1 : [a,b] \to \mathbb{R}^{p\times n}$, $Q_2 : [a,b] \to \mathbb{R}^{q\times m}$ and $\mathcal{P}_{\{R_i\}} : L_2^n[a,b] \to L_2^q[a,b]$.

$$\mathcal{P}\left[\begin{smallmatrix} P, & Q_1 \\ Q_2, & \{R\}\end{smallmatrix}\right]\begin{bmatrix}x\\ \mathbf{y}\end{bmatrix}(s) := \begin{bmatrix} Px + \int_a^b Q_1(s)\mathbf{y}(s)ds \\ Q_2(s)x + \mathcal{P}_{\{R_i\}}\mathbf{y}(s)\end{bmatrix}. \tag{7}$$

3.2 Partial Integral Equations

Partial Integral Equations (PIEs) take the form

$$\begin{aligned} \mathcal{T}\dot{\mathbf{x}}(t) &= \mathcal{A}\mathbf{x}(t) + \mathcal{B}_1 w(t) + \mathcal{B}_2 u(t), \\ z(t) &= \mathcal{C}_1\mathbf{x}(t) + \mathcal{D}_{11}w(t) + \mathcal{D}_{12}u(t), \\ y(t) &= \mathcal{C}_2\mathbf{x}(t) + \mathcal{D}_{21}w(t) + \mathcal{D}_{22}u(t), \end{aligned} \tag{8}$$

where $\mathcal{T}, \mathcal{A}, \mathcal{B}_i, \mathcal{C}_i$ and $\mathcal{D}_{ij}$ are 4-PI operators. PIEs are useful in analysis and control of linear PDEs because they do not require boundary conditions or continuity constraints. It was shown in [8] that any linear PDE with a single spatial variable can be represented as a PIE. In the following example, we provide the PI operators that transform a 4^{th}-order linear PDE (of the form (4)) to a PIE.

Consider a linear PDE of the form,

$$\begin{aligned} \dot{x}(s,t) &= \sum_{j=0}^{4} A_j(s)\partial_s^j x(s,t) + B_1(s)w(t) + B_2(s)u(t), \\ Bx_b(t) &= 0, \end{aligned} \tag{9}$$

where $A_i : \mathbb{R} \to \mathbb{R}^{n\times n}$, $B_i : \mathbb{R} \to \mathbb{R}^n$, $\text{rank}(B) = 4n$ and

$x_b(t) =$

$$\text{col}(x(a,t), \partial_s x(a,t), \partial_s^2 x(a,t), \partial_s^3 x(a,t), x(b,t), \partial_s x(b,t), \partial_s^2 x(b,t), \partial_s^3 x(b,t)).$$

We define the 3-PI operators,

$$\mathcal{T} = \mathcal{P}_{\{G_i\}}, \mathcal{A} = \mathcal{P}_{\{H_i\}}, \mathcal{B}_i = \mathcal{P}_{\{B_i,0,0\}}, C_i = 0, \mathcal{D}_{ij} = 0, \tag{10}$$

where

$$G_0(s) = 0, \quad G_1(s,\theta) = -K_0(s)(BT(b))^{-1}BQ(b,\theta) + L_0(s,\theta),$$
$$G_2(s,\theta) = -K_0(s)(BT(b))^{-1}BQ(b,\theta),$$
$$H_0(s) = A_4(s), \quad H_1(s,\theta) = -\sum_{j=0}^{3}(A_j(s)K_j(s)(BT(b))^{-1}BQ(b,\theta) + L_j(s,\theta)),$$
$$H_2(s,\theta) = -\sum_{j=0}^{3} A_j(s)K_j(s)(BT(b))^{-1}BQ(b,\theta),$$
$$T(s) = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ I & s-a & (s-a)^2/2 & (s-a)^3/3! \\ 0 & I & s-a & (s-a)^2/2! \\ 0 & 0 & I & s-a \\ 0 & 0 & 0 & I \end{bmatrix}, Q(s,\theta) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ (s-\theta)^3/3! \\ (s-\theta)^2/2! \\ s-\theta \\ I \end{bmatrix},$$
$$K_j(s) = T_{\{j+5\}}(s), L_j(s,\theta) = Q_{\{j+5\}}(s,\theta). \tag{11}$$

The row index j for T and Q stands for j^{th} subpartition row shown above.

Then for any x that satisfies linear PDE (9), $\mathbf{x} = \partial_s^4 x$ satisfies the PIE (8) for the 3-PI operators as defined in (10). Using A_i and B_i as defined in (5), we define the 3-PI operators (10) to find the PIE form of the PDE (3).

3.3 Linear Partial Integral Inequalities

Optimization problems with PI operator variables and Linear PI Inequality constraints can be solved using the software package PIETOOLS [15]. These optimization problems are referred to as Linear PI Inequalities (LPIs) and take the form

$$\mathcal{P}\begin{bmatrix} P_0, & Q_0 \\ Q_0^T, & \{R_{0i}\} \end{bmatrix} + \sum_{k=1}^{N} x_j \mathcal{P}\begin{bmatrix} P_k, & Q_k \\ Q_k^T, & \{R_{ki}\} \end{bmatrix} \succcurlyeq 0, \tag{12}$$

where the decision variable is $x \in \mathbb{R}^N$ and $\mathcal{P}\begin{bmatrix} P_k, & Q_k \\ Q_k^T, & \{R_{ki}\} \end{bmatrix} : \mathbb{R}^m \times L_2^n[a,b] \to \mathbb{R}^m \times L_2^n[a,b]$ is a given self-adjoint 4-PI operator for $0 \le k \le N$ and $k \in \mathbb{Z}$.

3.4 H_∞-optimal Observer Design

In this subsection, we extend the LMI result used for design of H_∞-optimal observers of ODEs to an LPI for observation of systems defined by PIEs.

3.4.1 H_∞-optimal Observer Design For Linear ODEs

Consider the problem of designing the H_∞ optimal observer for a linear ODE whose dynamics are governed by the equations

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bw(t), \qquad x(0) = 0, \\ z(t) &= C_1 x(t) + D_{11} w(t), \\ y(t) &= C_2 x(t) + D_{21} w(t), \end{aligned} \tag{13}$$

where $A : \mathbb{R}^n \to \mathbb{R}^n$, $B : \mathbb{R} \to \mathbb{R}^n$, $C_i : \mathbb{R}^n \to \mathbb{R}$, $D_{i1} : \mathbb{R} \to \mathbb{R}$, $w \in L_2([0, \infty))$ is a disturbance, y is the measured output and z is the regulated output. Let the observer dynamics be

$$\begin{aligned} \dot{\hat{x}}(t) &= A\hat{x}(t) + L(C_2\hat{x}(t) - y(t)), \\ \hat{z}(t) &= C_1\hat{x}(t), \end{aligned} \tag{14}$$

where $L : \mathbb{R} \to \mathbb{R}^n$, $\hat{x}$ is the estimated state and $z_e(t) = \hat{z}(t) - z(t)$ is the error in the estimate of regulated output. The closed-loop dynamics of the error system with state $e(t) := \hat{x}(t) - x(t)$ as determined by Eqs. (13) and (14) result in

$$\dot{e}(t) = (A + LC_2)e(t) - (B + LD_{21})w(t), \qquad z_e(t) = C_1 e(t) - D_{11} w(t). \tag{15}$$

Using the KYP Lemma, [2], we can formulate the following LMI that provides necessary and sufficient conditions for the existance of an H_∞-optimal observer with $\frac{\|z_e\|_{L_2}}{\|w\|_{L_2}} \leq \gamma$. The LMI used to find H_∞-optimal observer for this system is

$$\begin{aligned} &P > 0 \\ &\begin{bmatrix} -\gamma I & -D_{11}^T & -(PB + ZD_{21})^T \\ *^T & -\gamma I & C_1 \\ *^T & *^T & (PA + ZC_2)^T + PA + ZC_2 \end{bmatrix} \leq 0 \end{aligned} \tag{16}$$

where the observer gains are $L = P^{-1}Z$.

3.4.2 H_∞-optimal Observer Design For Linear PIEs

We can generalize the LMI in Subsection 3.4.1 to an LPI - yielding continuum observers for systems modeled using PIEs. Specifically, suppose the system to be ob-

served is defined by Eqns. (8). We propose an observer structure similar to Eqn. (14). Then the closed-loop error dynamics are

$$\begin{aligned} \mathcal{T}\dot{\mathbf{e}}(t) &= (\mathcal{A}+\mathcal{L}\mathcal{C}_2)\mathbf{e}(t) - (\mathcal{B}+\mathcal{L}\mathcal{D}_{21})w(t), \qquad \mathbf{e}(0)=0 \\ z_e(t) &= \mathcal{C}_1\mathbf{e}(t) - \mathcal{D}_{11}w(t) \end{aligned} \tag{17}$$

where $\mathcal{T} : L_2^n[a,b] \to L_2^n[a,b]$, $\mathcal{A} : L_2^n[a,b] \to L_2^n[a,b]$, $\mathcal{B} : \mathbb{R} \to L_2^n[a,b]$, $\mathcal{C}_i : L_2^n[a,b] \to \mathbb{R}$, $\mathcal{D}_{i1} : \mathbb{R} \to \mathbb{R}$ and $\mathcal{L} : \mathbb{R} \to L_2^n[a,b]$ are PI operators. The LPI constraints for the PIE are given below.

Suppose there exists bounded linear operators $\mathcal{P} : L_2^n[a,b] \to L_2^n[a,b]$ and $\mathcal{Z} : \mathbb{R} \to L_2^n[a,b]$, such that $\mathcal{P}$ is coercive and

$$\begin{bmatrix} -\gamma I & -\mathcal{D}_{11}^* & -(\mathcal{P}\mathcal{B}+\mathcal{Z}\mathcal{D}_{21})^*\mathcal{T} \\ (\cdot)^* & -\gamma I & \mathcal{C}_1 \\ (\cdot)^* & (\cdot)^* & (\mathcal{P}\mathcal{A}+\mathcal{Z}\mathcal{C}_2)^*\mathcal{T} + \mathcal{T}^*(\mathcal{P}\mathcal{A}+\mathcal{Z}\mathcal{C}_2) \end{bmatrix} \preccurlyeq -\epsilon I \tag{18}$$

where $\epsilon > 0$ and $I : L_2 \to L_2$ is an identity operator. Then $\mathcal{P}^{-1}$ exists and is a bounded linear operator and for $\mathcal{L} = \mathcal{P}^{-1}\mathcal{Z}$ and any $w \in L_2([0,\infty))$ any solution for the system (17) satisfies $\|z_e\|_{L_2} < \gamma\|w\|_{L_2}$.

For proof, please refer [17].

3.5 H_∞-optimal Controller Synthesis

Similar to subsection 3.4, we state the LMI for finding an H_∞ -optimal controller for a linear ODE system and then present the extension to an LPI.

3.5.1 H_∞-optimal Controller Synthesis For Linear ODEs

Consider an ODE

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B_1 w(t) + B_2 u(t), \\ z(t) &= C_1 x(t) + D_{11} w(t) + D_{12} u(t), \end{aligned} \tag{19}$$

where $A : \mathbb{R}^n \to \mathbb{R}^n$, $B_i : \mathbb{R} \to \mathbb{R}^n$, $C_1 : \mathbb{R}^n \to \mathbb{R}$, $D_{1i} : \mathbb{R} \to \mathbb{R}$, $w \in L_2([0,\infty))$ is a disturbance, u is the input and z is the regulated output. If there exists a $P > 0$, where $P \in \mathbb{S}^n$, and $Z : \mathbb{R}^n \to \mathbb{R}$ such that

$$\begin{bmatrix} PA^T + AP + Z^T B_2^T + B_2 Z & B_1 & PC_1^T + Z^T D_{12}^T \\ *^T & -\gamma I & D_{11}^T \\ *^T & *^T & -\gamma I \end{bmatrix} \leq 0,$$

then for $u(t) = ZP^{-1}x(t)$, the solutions of (19) satisfy $\|z\|_{L_2} \leq \gamma\|w\|_{L_2}$ for $w \in L_2([0,\infty))$.

3.5.2 H_∞-optimal Controller Synthesis For Linear PIEs

Similar to the ODE shown earlier, consider a PIE system

$$\begin{aligned} \mathcal{T}\dot{\mathbf{x}}(t) &= \mathcal{A}\mathbf{x}(t) + \mathcal{B}_1 w(t) + \mathcal{B}_2 u(t), \qquad \mathbf{x}(0) = 0, \\ z(t) &= \mathcal{C}_1\mathbf{x}(t) + \mathcal{D}_{11} w(t) + \mathcal{D}_{12} u(t), \end{aligned} \tag{20}$$

where $\mathcal{T} : L_2^n[a,b] \to L_2^n[a,b]$, $\mathcal{A} : L_2^n[a,b] \to L_2^n[a,b]$, $\mathcal{B}_i : \mathbb{R} \to L_2^n[a,b]$, $\mathcal{C}_1 : L_2^n[a,b] \to \mathbb{R}$ and $\mathcal{D}_{1i} : \mathbb{R} \to \mathbb{R}$ are PI operators. Then the following LPI can be used to find H_∞-optimal controller gains for a linear PIE.

Suppose there exists bounded linear operators $\mathcal{P} : L_2^n[a,b] \to L_2^n[a,b]$ and $\mathcal{Z} : L_2^n[a,b] \to \mathbb{R}$, such that $\mathcal{P}$ is coercive and

$$\begin{bmatrix} -\gamma I & \mathcal{D}_{11}^* & (\mathcal{P}\mathcal{C}_1^* + \mathcal{Z}^*\mathcal{D}_{12}^*) \\ (\cdot)^* & -\gamma I & \mathcal{B}_1 \\ (\cdot)^* & (\cdot)^* & (\cdot)^* + \mathcal{T}(\mathcal{A}\mathcal{P} + \mathcal{B}_2\mathcal{Z})^* \end{bmatrix} \preccurlyeq -\epsilon I \tag{21}$$

where $\epsilon > 0$ and $I : L_2 \to L_2$ is an identity operator. Then there exists a bounded and coercive linear operator $\mathcal{P}^{-1}$. Further, for $u = \mathcal{K}\mathbf{x}$, where $\mathcal{K} = \mathcal{Z}\mathcal{P}^{-1}$, and any $w \in L_2$ any solution for (20) satisfies $\|z\|_{L_2} \leq \gamma\|w\|_{L_2}$.

For proof, please refer [14].

3.6 PIETOOLS: A MATLAB Toolbox For Handling PI Operators

The PI-operators form a *-subalgebra, i.e. the operations such as addition, concatenation, composition, adjoint operations on PI operators are closed and result in another linear PI operator. Then, if the operators $\mathcal{P}$ and $\mathcal{Z}$ are parametrized by PI operators then the contraints in (18) and (21) are LPIs. Furthermore, in [13], it was proven that positivity constraints on LPIs can be enforced using LMIs; the numerical implementation is done in MATLAB by using PIETOOLS [15]. PIETOOLS borrows some functions for handling polynomials from SOSTOOLS [12] and can use a variety of SDP solvers such as SeDuMi, Mosek et c.

4 Decentralized Implementation Of The Controller And Observer

In this section, a method for decentralized implementation of controllers and observers in PI form is presented. To calculate the gains $\mathcal{K}$ and $\mathcal{L}$ the inverse operator $\mathcal{P}^{-1}$ must be computed. A self-adjoint, bounded and coercive operator in 4-PI form has an inverse which is necessarily a 4-PI operator. Specifically, for a self-adjoint, bounded and coercive 4-PI operator $\mathcal{P}\left[\begin{smallmatrix} P, & Q \\ Q^T, & \{R_i\} \end{smallmatrix}\right]$, if $R_1 = R_2$ then we have an exact formula for the inverse $\mathcal{P}^{-1} = \mathcal{P}\left[\begin{smallmatrix} \hat{P}, & \hat{Q}_1 \\ \hat{Q}_2, & \{\hat{R}_i\} \end{smallmatrix}\right]$ as presented in Theorem 8 of [11]. However, to find $\mathcal{P}^{-1}$, we need to find $R_0(s)^{-1}$ which is a rational function. Instead of calculating $R_0(s)^{-1}$ exactly, we approximate it numerically by a polynomial and calculate the inverse $\mathcal{P}^{-1}$. The controller ($\mathcal{K} = \mathcal{Z}\mathcal{P}^{-1}$) or observer ($\mathcal{L} = \mathcal{P}^{-1}\mathcal{Z}$) gains are in 3-PI form

$$(\mathcal{K}\mathbf{x})(s) = K_0(s)\mathbf{x}(s) + \int_a^s K_1(s,\theta)\mathbf{x}(\theta)d\theta + \int_s^b K_2(s,\theta)\mathbf{x}(\theta)d\theta,$$
$$(\mathcal{L}\mathbf{x})(s) = L_0(s)\mathbf{x}(s) + \int_a^s L_1(s,\theta)\mathbf{x}(\theta)d\theta + \int_s^b L_2(s,\theta)\mathbf{x}(\theta)d\theta,$$

where the K_i's and L_i's are matrix-valued polynomials of appropriate dimensions.

For the implementation, we discretize the domain $[0, L]$ by an equispaced grid with n grids. The distributed state $x(\cdot, t)$ is approximated by the vector of states $x_i(t) = x(s_i, t)$ where $s_i = i\Delta s$, $i \in \{0, ..., n\}$ and $\Delta s = L/(n+1)$. Then,

$$K_n x_i(t) := (\mathcal{K}x)(s_i, t) \cong K_0(s_i)x_i(t) + K_{1a}(s_i)\sum_{j=0}^{i} K_{1b}(s_j)x_j(t)\Delta s + K_{2a}(s_i)\sum_{j=i+1}^{n} K_{2b}(s_j)x_j(t)\Delta s,$$

where K_n is the finite-dimensional approximation of $\mathcal{K}$ on the grid with n segments. Note that the K_i's are polynomials and hence can be factored as $K_i(s,\theta) = K_{ia}(s)K_{ib}(\theta)$.

Typically, a full-state feedback requires n^2 information transfers to find the inputs $u_i(t) = K_n x_i(t)$ for all i. A system is said to be decentralized if it operates based on local information. However, definition of localness of information is not clearly defined. Hence, we say a system to be decentralized when the number of information transfers required to determine the inputs u_i is of order $O(N)$. As discussed below, an observer/controller of 3-PI form can indeed be implemented such that inputs are calculated using information from immediately adjacent nodes (local information).

Each node i has access to the information $K_0(s_i)x_i(t)$, $K_{1a}(s_i)$ and $K_{2a}(s_i)$. The information needed from other segments are the cumulative values $c_i(t) = \sum_{j=0}^{i-1} K_{1b}(s_j)x_j(t)$ and $d_{i+1}(t) = \sum_{j=i+1}^{n} K_{2b}(s_j)x_j(t)$. Suppose node i receives

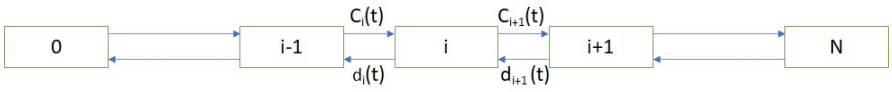


Fig. 3: Information transfer needed to find input $u_i(t)$

the sum $c_i(t)$ from the node $i-1$. Then we find $c_{i+1}(t) = c_i(t) + K_{1b}(s_i)x_i(t)$ and send the information to node $i+1$. In the opposite direction, we find $d_i(t) = d_{i+1}(t) + K_{2b}(s_i)x_i(t)$ and transmit the information to node $i-1$. Then every node uses only local information to determine the input

$$u_i(t) = K_0(s_i)x_i(t) + K_{1a}(s_i)c_i(t)\Delta s + K_{2a}(s_i)d_{i+1}(t)\Delta s$$

and the exchange of information can be achieved with just $2n$ information transfers. Thus, we do not require communication between non-adjacent nodes. Further, delay arising from communication between distal nodes are avoided and dramatically reduces communication overhead.

4.1 Observer Implementation In MATLAB

Using PIETOOLS, we design the H_∞-optimal observer for each of the setpoints shown in Figure. 2 which results in 3 different observer gains for corresponding to the different linearized PDEs. The linearized observers are implemented as a single gain-scheduled observer where the active observer is determined by the sector in which the tip of the arm is located. The domain $[0,1]$ is divided into a equispaced grid with 10 grids. The even-order derivatives in the PDE (2) were approximated by a 2^{nd} order central difference expression while the odd-order derivatives were approximated by using 1^{st} order upwind-biased difference expressions. The discretization methods are not discussed in detail because it is not the focus of this work and other discretization methods can be used to approximate the solution of PDE. MATLAB `ode` solver was used to solve the finite difference approximation of the PDE using the initial conditions

$$\hat{x}_i(s,0) = x_1(s,0) = x_2(s,0) = x_4(s,0) = 0, x_3(s,0) = (2s^2 - (8/3)s^3 + s^4)$$

and input disturbance $w(t) = \frac{sin(t)}{10t}$ where x is the system state, $\hat{x}$ is the observer state and w is the input disturbance.

The two observed outputs y_1 and y_2 are the errors in transverse displacement of the tip and average error in estimate of PIE state. The regulated output is the average error in the estimate of the PIE states.

$$y(t) = \begin{bmatrix} \hat{x}(L,t) - x(L,t) \\ \int_0^L \mathbf{e}(\eta,t)d\eta \end{bmatrix} \quad z(t) = \int_0^L \mathbf{e}(s,t)ds, \qquad \mathbf{e} = \partial_s^4\hat{x} - \partial_s^4 x$$

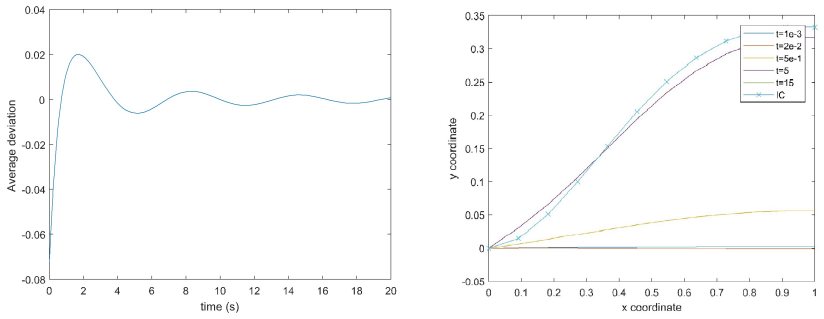
The outputs can be expressed in PIE form (8) by defining $\mathcal{C}_i$ and $\mathcal{D}_{ij}$ as

$$\mathcal{D}_{ij} = 0,\ \mathcal{C}_1 = \mathcal{P}_{\{0,I,I\}},\ \mathcal{C}_2 = \mathcal{P}_{\{0,\mathcal{C}_2,\mathcal{C}_2\}}, \qquad C_2(s,\theta) = \begin{bmatrix} G_1(L,\theta) \\ I \end{bmatrix} \tag{22}$$

where G_1 is as defined in (11). The material properties and dimensions for the soft-robot arm are set as $E = 5\times10^3 N/m^2$, $r = 0.025m$, $L = 1m$ and $\rho = 1.1\times10^3 kg/m^3$.

A stepwise procedure to implement the observer:

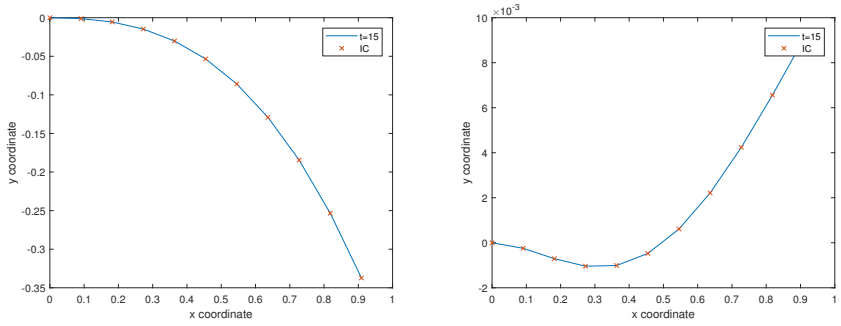
1. Define the PI operators $\mathcal{T}$, $\mathcal{A}$, $\mathcal{B}_i$ as in (5) and $\mathcal{C}_i$, $\mathcal{D}_{ij}$ as in (22).
2. Solve the LPI (18), using PIETOOLS, to find $\mathcal{P}$ and $\mathcal{Z}$.
3. Find the observer $\mathcal{L} = \mathcal{P}^{-1}\mathcal{Z}$ for each setpoint.
4. Find finite-dimensional approximation L of the observer $\mathcal{L}$.
5. Use finite-difference approximation of the PDE (4) to find A, B and C_i.
6. Implement the closed-loop error dynamics (15) in MATLAB.



(a) $z_3(t)$ vs t (b) $\hat{x}_3(s,t)$ vs s at different t values

Fig. 4: The figure on the left shows the time-evolution of the regulated output, which in this case is the average error in the estimate of the state. On the right, we plot the estimate of the transverse displacement at different times.

As seen in Fig. 4a, the error asymptotically converges to zero and the approximation of the actual shape of the arm is shown in Fig. 4b. Figure 5 shows that gain-scheduled observer converges for different initial conditions. If the LPI (18) is solved with a large positive value for ϵ then, for the obtained observer, the observer state converges to system state faster. However, using large values for ϵ can result in high observer gain values and higher H_∞-norm.



(a) (b)

Fig. 5: Observer state converges for different Initial Conditions.

5 Conclusion

In this chapter, we discussed a method for linearizing a nonlinear PDE about set-points. Next, we introduced the PIE representation of linear PDE and posed the design of H_∞-optimal observer and controller gains as an optimization problem with PI variables and LPI constraints. A computational tool, PIETOOLS, was used to solve the resulting LPI optimization problems. Next, we presented a method to implement the proposed continuum controllers or observers in a decentralized manner. Finally, we applied the methods developed for linear coupled PDE systems to the model of a soft-robot arm. MATLAB implementation was used to show convergence and stability of the observer.

Acknowledgements This work was supported by Office of Naval Research Award N00014-17-1-2117 and National Science Foundation under grant No. 1739990.

References

1. A. S. Banach and W. T. Baumann. Gain-scheduled control of nonlinear partial differential equations. In *29th IEEE Conference on Decision and Control*, pages 387–392. IEEE, 1990.
2. S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*, volume 15. SIAM, 1994.
3. M. Cianchetti, T. Ranzani, G. Gerboni, T. Nanayakkara, K. Althoefer, P. Dasgupta, and A. Menciassi. Soft robotics technologies to address shortcomings in today's minimally invasive surgery: the STIFF-FLOP approach. *Soft robotics*, 1(2):122–131, 2014.
4. N. Darivandi, K. Morris, and A. Khajepour. An algorithm for LQ optimal actuator location. *Smart materials and structures*, 22(3):035001, 2013.
5. I. S. Godage, G. A. Medrano-Cerda, D. T. Branson, E. Guglielmino, and D. G. Caldwell. Modal kinematics for multisection continuum arms. *Bioinspiration & biomimetics*, 10(3):035002, 2015.

6. R. Kang, A. Kazakidi, E. Guglielmino, D. T. Branson, D. P. Tsakiris, J. A. Ekaterinaris, and D. G. Caldwell. Dynamic model of a hyper-redundant, octopus-like manipulator for underwater applications. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4054–4059. IEEE, 2011.
7. L. Lessard and S. Lall. Quadratic invariance is necessary and sufficient for convexity. In *Proceedings of the 2011 American Control Conference*, pages 5360–5362. IEEE, 2011.
8. M. M. Peet, A. Das, S. Shivakumar, and S. Weiland. Representation and stability analysis of PDE-ODE coupled systems. In *Proceedings of the 3rd IFAC/IEEE CSS Workshop on Control of Systems Governed by Partial Differential Equations CPDE and XI Workshop Control of Distributed Parameter Systems*, 2019.
9. F. Matsuno and K. Suenaga. Control of redundant 3D snake robot based on kinematic model. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 2061–2066. IEEE, 2003.
10. A. H. Nayfeh and P. F. Pai. *Linear and nonlinear structural mechanics*. John Wiley & Sons, 2008.
11. M. M. Peet. H_∞-optimal control of systems with multiple state delays: Part 1. In *Proceedings of the American Control Conference*, 2019.
12. S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. SOSTOOLS and its control applications. In *Positive polynomials in control*, pages 273–292. Springer, 2005.
13. S. Shivakumar, A. Das, and M. M. Peet. PIETOOLS: A MATLAB toolbox for manipulation and optimization of partial integral operators. In *Proceedings of the American Control Conference*, 2020.
14. S. Shivakumar, A. Das, S. Weiland, and M. M. Peet. Duality and H_∞-optimal control of coupled ODE-PDE systems. Technical report, arXiv.org, 2020. https://arxiv.org/abs/2004.03638.
15. S. Shivakumar and M. Peet. *PIETOOLS for Time-Delay Systems*. https://codeocean.com/capsule/7653144/.
16. V. Vikas, E. Cohen, R. Grassi, C. Sözer, and B. Trimmer. Design and locomotion control of a soft robot using friction manipulation and motor–tendon actuation. *IEEE Transactions on Robotics*, 32(4):949–959, 2016.
17. S. Wu, S. Shivakumar, M. M. Peet, and C. Hua. H_∞-optimal observer design for linear systems with delays in states, outputs and disturbances. Technical report, arXiv.org, 2020. https://arxiv.org/abs/2004.04482.
18. T. Zheng, D. T. Branson, R. Kang, M. Cianchetti, E. Guglielmino, M. Follador, G. A. Medrano-Cerda, I. S. Godage, and D. G. Caldwell. Dynamic continuum arm model for use with underwater robotic manipulators inspired by octopus vulgaris. In *2012 IEEE International Conference on Robotics and Automation*, pages 5289–5294. IEEE, 2012.