Learning-based State-dependent Coefficient Form Task Space Tracking Control of Soft Robot

Rounak Bhattacharya, Ghananeel Rotithor, Ashwin P. Dani

***Abstract*—In this paper, a data-driven modeling and control framework is developed for task space control of a soft robot gripper which consists of four individual soft fingers. Each of the four fingers is modeled as a manipulator with high degrees of freedom. The corresponding task space dynamics of the manipulator are derived using a rigid-link approximation of the continuum manipulator. A neural network approach is used to learn the derived dynamics in State Dependent Coefficient (SDC) form. Using the learned SDC matrices, an asymptotically stable optimal closed-loop tracking controller which is based on solving the State Dependent Riccati Equation (SDRE) is derived. The model learning and trajectory tracking controller is implemented on an open source Soft Motion (SoMo) platform simulating the soft gripper motion and corresponding tracking results are presented.**

I. Introduction

Soft robots have made substantial progress in past decade [1], [2]. The ability of soft robots to conform to various different shapes make them suitable for grasping and manipulation of novel objects. They possess inherent compliance and adaptability, which makes them safer during their interaction with the environment for manipulation tasks and for safe human-robot collaboration. Soft robots have enabled novel ways of grasping and manipulation of objects in manufacturing applications. Although soft robots have transformed the robot grasping and manipulation, there are still many challenges, specially in improving the precision and speed of their operations.

To improve the grasping precision, model-based regulation and tracking controllers are developed in [3]–[5]. Mathematical modeling of soft robots has been widely studied for control development and shape estimation. Starting from general kinematic models developed in [6], [7], detailed dynamic models have been developed in [8]–[11]. Developing mathematical models of soft robots from first principles is a complex process. These models are generally high dimensional and complex for which model reduction techniques are used before designing a controller [12]. A model free controller is developed in [13] for soft robot locomotion. A neural network controller for fully actuated model of the soft robot is developed in [14]. Data-driven modeling using deep neural networks is used in [15] for learning nonlinear dynamic models to design a model predictive controller. Newer advancements in Koopman-based data-driven modeling and model reduction techniques are used in [16], [17] to incorporate external loading for improving the tracking accuracy and speed.

Experimental platforms such as OctArm are developed for implementation and testing of soft robot controllers [18]. Newer simulation platforms [19], [20] enable easier testing and implementation of soft robot controllers. Traditionally, a wide variety of physics-based simulation tools have been used in robotics such as Gazebo, PyBullet, Webots and MuJoCo. It is very hard to model soft robots using these general-purpose tools. Recent software platforms such as Soft Motion (SoMo) [20] provide a physics-based simulator for testing and validation of data-driven modeling, control design and planning methods. SoMo has recently been used in [21] to show dexterous in-hand manipulation.

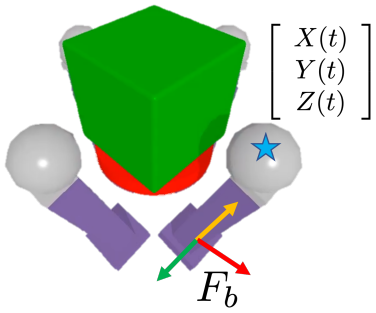

Figure 1. Snapshot of the soft gripper in SoMo simulation where F_b is the fixed frame and the end-tip (star marker) is controlled.

The main contribution of this paper is to develop an optimal nonlinear controller using state dependent Riccati equation (SDRE) method which uses a state dependent coefficient (SDC) form model learned from data following the data-driven control design strategy. The SDC parameters of the SDRE control are generally obtained using heuristic rules, see, e.g., [22], [23]. In this paper, SDC parameters of the soft robot dynamics are learned from data obtained using deep neural networks. The soft manipulator is modeled as a discrete approximation of the continuum robot which is represented using an under-actuated Euler-Lagrange (E-L) model. The E-L approximation models developed for the soft robots use full actuation, see, e.g., [3], [4], [14], i.e., the number of joints in discrete approximation and number of actuators are the same. However, in practice many soft

This work was supported by a Space Technology Research Institutes grant (number 80NSSC19K1076) from NASA Space Technology Research Grants Program. Rounak Bhattacharya, Ghananeel Rotithor and Ashwin P. Dani are with the Department of Electrical and Computer Engineering at University of Connecticut, Storrs, CT 06269. Email: {rounak.bhattacharya; ghananeel.rotithor;ashwin.dani}@uconn.edu.

robots are underactuated. The underactuation adds another level complexity to the controller development which can be easily incorporated in the SDRE control design. The controller objective is to track desired end-tip trajectories of the soft manipulator. The controller is implemented in SoMo simulator to simulate a pinching grasp motion using multiple soft manipulators controlled individually.

II. Dynamic Modeling and Neural Network Learning

A. Coordinate Frame and Forward Kinematics

Consider a soft robot gripper as shown in Figure 1. Each of the fingers is modeled as a rigid link approximation of a continuum manipulator with discrete set of joints. Additionally, the end-point of each of the fingers is represented by its $X(t)$, $Y(t)$ and $Z(t)$ position coordinates with respect to the fixed inertial frame F_b attached to the base of the continuum manipulator. In the following sections, the dynamic model of the rigid-link approximation of the continuum manipulator is developed.

Consider the end-point of the soft robot finger whose 2-dimensional (2D) position in a fixed inertial reference frame F_b is given by

$$x(t) = \begin{bmatrix} X(t) & Y(t) \end{bmatrix}^T \in \mathcal{X} \tag{1}$$

where $\mathcal{X} \subset \mathbb{R}^2$ is a bounded set. The vector $x(t)$ denotes the task space coordinate of each finger.

The finger end-point 2D position $x(t)$ and the joint angles $q(t) \in \mathcal{Q}$ of the robot are related by $x \triangleq f(q)$, where $f : \mathcal{Q} \to \mathcal{X}$ represents the forward kinematics such that $\mathcal{Q} \subset \mathbb{R}^{10}$ and $q(t)$ are the joint angles of the finger implying that each finger of the soft-manipulator consists of 10 joint angles.

B. Task Space Dynamics

To derive the task space dynamics, consider the relationships between the joint angular velocities $\dot{q}$ and the end-point velocity $\dot{x}$

$$\dot{x} = J(q)\dot{q} \tag{2}$$

where $J(q) \triangleq \frac{\partial f(q)}{\partial q} \in \mathbb{R}^{2\times 10}$ denotes the manipulator Jacobian matrix. Taking the time derivative of (2) yields

$$\ddot{x} = \dot{J}(q)\dot{q} + J(q)\ddot{q} \tag{3}$$

where $\dot{q}(t)$, $\ddot{q}(t) \in \mathbb{R}^{10}$ are the joint angular velocities and accelerations, respectively. The joint space dynamic model of the discrete approximation of continuum manipulator with n-number of joints and m control inputs in the E-L form is given by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = Bu \tag{4}$$

where $M(q) \in \mathbb{R}^{10\times 10}$ represents the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{10\times 10}$ represents centripetal and Coriolis forces, $B \in \mathbb{R}^{10\times 2}$ is an actuation matrix, $G(q) \in \mathbb{R}^{10}$ represents the effect of gravity and $u \in \mathbb{R}^2$ represents the control input vector.

Remark 1. *The actuation matrix B has full column rank, i.e.,* $\mathrm{rank}(B) = 2$.

Task space dynamics of the continuum manipulator can now be derived using the joint space dynamics in (4). Using (2), the following relation is obtained for the joint velocity

$$\dot{q} = J^{+}\dot{x} \tag{5}$$

where $J^{+} = J^T(JJ^T)^{-1} \in \mathbb{R}^{10\times 2}$ denotes the Jacobian matrix pseudo-inverse. Using the relation in (3), the following relation can be obtained for the joint space acceleration

$$\ddot{q} = J^{+}(\ddot{x} - \dot{J}J^{+}\dot{x}). \tag{6}$$

Substituting (5) and (6) in the Euler-Lagrange model in (4), the dynamic model of the manipulator in task space can be written as

$$MJ^{+}\ddot{x} - (MJ^{+}\dot{J}J^{+} - CJ^{+})\dot{x} + G = Bu. \tag{7}$$

After some algebraic manipulations, the task space acceleration $\ddot{x}(t)$ can be obtained as

$$\ddot{x} = (\dot{J}J^{+} - JM^{-1}CJ^{+})\dot{x} - JM^{-1}G + JM^{-1}Bu. \tag{8}$$

Let $\mathbf{x} \in \mathbb{R}^4$ be a state vector defined by

$$\mathbf{x}(t) = \begin{bmatrix} x^T(t) & \dot{x}^T(t) \end{bmatrix}^T \tag{9}$$

Using (8) the nonlinear state-space form is given by

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ (\dot{J}J^{+} + JM^{-1}CJ^{+})\dot{x} + JM^{-1}G \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{2\times 2} \\ JM^{-1}B \end{bmatrix} u. \tag{10}$$

The task space dynamics in (10) can be written in a compact nonlinear form as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\,u \tag{11}$$

where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \dot{x} \\ (\dot{J}J^{+} + JM^{-1}CJ^{+})\dot{x} + JM^{-1}G \end{bmatrix}, \tag{12}$$

and

$$\mathbf{G}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{2\times 2} \\ JM^{-1}B \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{2\times 2} \\ \mathbf{G}_2(\mathbf{x}) \end{bmatrix}. \tag{13}$$

The nonlinear dynamics in (11) can be represented in a SDC form as follows

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\,u = \mathbf{A}(\mathbf{x})\,\mathbf{x} + \mathbf{G}(\mathbf{x})\,u \tag{14}$$

where the state dependent matrix $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{4\times 4}$ has the form

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{2\times 2} & \mathbf{I}_2 \\ \mathbf{A_{21}}(\mathbf{x}) & \mathbf{A_{22}}(\mathbf{x}) \end{bmatrix}. \tag{15}$$

In (15), the sub-matrices $\mathbf{A}_{21}(\mathbf{x})$ and $\mathbf{A}_{22}(\mathbf{x})$ are state dependent and represent the dependence of the task space acceleration $\ddot{x}(t)$ on the task space position and velocity $\mathbf{x}(t)$. The SDC parametrization $\mathbf{A}(\mathbf{x})$ is non-unique which provides a flexibility in designing the SDRE control. In this

paper, the SDC matrix $\mathbf{A}(\mathbf{x})$ is identified using data collected from the SoMo simulator which simulates the soft robot gripper. It is important to note that the the collected position trajectories $x(t)$ are translated to ensure that $\mathbf{f}(0) = 0$. This is equivalent to learning the task-space dynamics in a translated coordinate frame. The SDC parameter model identification process is described next.

C. Learning SDC Model from Data

The SDC model is learned by collecting data using the simulation platform. To identify the model of the soft finger, position and velocity data of the end-tip is collected. The collected data is used to learn the SDC matrices $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$. In the case of the soft robot model, the exact analytical form of $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ is difficult to compute. Hence, a learning-based approach is used. The unknown components of $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ are learned using neural networks. Consider the following neural network for learning the components of $\mathbf{A}$ matrix

$$\text{vec}\left(\begin{bmatrix} \mathbf{A}_{21}(\mathbf{x}) & \mathbf{A}_{22}(\mathbf{x}) \end{bmatrix}\right) = \mathbf{NN}_{\theta_1}(\mathbf{x}), \tag{16}$$

where $\mathbf{NN}_{\theta_1} : \mathbb{R}^4 \to \mathbb{R}^8$ is a neural network function parameterized by θ_1 and $\text{vec}(\cdot)$ denotes the vectorization operation which converts matrix by stacking its columns. Similarly, consider the second neural network to learn the unknown components of the $\mathbf{G}(\mathbf{x})$ matrix

$$\text{vec}\left(\mathbf{G}_2(\mathbf{x})\right) = \mathbf{NN}_{\theta_2}(\mathbf{x}) \tag{17}$$

where $\mathbf{NN}_{\theta_2}(\mathbf{x}) : \mathbb{R}^4 \to \mathbb{R}^4$ is a neural network function parameterized by θ_2. To train the neural network functions in (16) and (17), sampled trajectories of the tuple $\left\{\dot{\mathbf{x}}^i(t), \mathbf{x}^i(t), u^i(t)\right\}_{t=0,i=1}^{t=T,i=N}$ are collected where the superscript i denotes the i^{th} training trajectory collected at every sampling instance. Similar notations apply for the acceleration $\dot{\mathbf{x}}$ and the control signal u. The training data is then used to minimize the following objective function

$$\theta^\star = \arg\min_{\theta} \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=0}^{T} \|\dot{\mathbf{x}}^i(t) - \mathbf{A}\left(\mathbf{x}^i(t)\right)\mathbf{x}^i(t) - \mathbf{G}\left(\mathbf{x}^i(t)\right)u^i(t)\|^2 \tag{18}$$

where $\theta = \{\theta_1, \theta_2\}$ are the combined parameters of both the neural networks. It should be noted that the matrices $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ depend on the parameters θ through the neural network. Although the loss function in (18) is written over the whole training data, the optimization of the parameters θ is achieved with a batch of the data using stochastic optimization.

III. SDC Tracking Control

The objective of the tracking controller is to track the desired soft finger end-point trajectory given by $\mathbf{x}_d(t) = \begin{bmatrix} X_d(t) & Y_d(t) & \dot{X}_d(t) & \dot{Y}_d(t) \end{bmatrix}^T$. To derive a tracking control law, let us define the tracking error as

$$\mathbf{e}(t) = \mathbf{x}_d(t) - \mathbf{x}(t). \tag{19}$$

The control objective can be written as

$$\|\mathbf{e}(t)\| \to 0 \text{ as } t \to \infty \tag{20}$$

Given the system in SDC parametrized form (14), consider the objective function to derive a SDC tracking control law which minimizes the infinite-horizon performance index

$$\mathbf{J}(\mathbf{e}, u) = \frac{1}{2}\int_0^\infty \left(\mathbf{e}^T(t)\mathbf{Q}(\mathbf{x})\mathbf{e}(t) + u^T(t)\mathbf{R}(\mathbf{x})u(t)\right) dt \tag{21}$$

where $\mathbf{Q}(\mathbf{x}) \in \mathbb{R}^{4\times4}$, $\mathbf{R}(\mathbf{x}) \in \mathbb{R}^{2\times2}$, $\mathbf{Q}(\mathbf{x}) \geq 0$, $\mathbf{R}(\mathbf{x}) > 0$, $\forall\, \mathbf{x}$ leads to the suboptimal tracking control law given by

$$u = \mathbf{K}(\mathbf{x})\mathbf{x} + \mathbf{K}_s(\mathbf{x})\mathbf{x}_d \tag{22}$$

with the control gains $\mathbf{K}(\mathbf{x})$, $\mathbf{K}_s(\mathbf{x}) \in \mathbb{R}^{2\times4}$ defined as

$$\begin{aligned} \mathbf{K}(\mathbf{x}) &= -\mathbf{R}^{-1}(\mathbf{x})\mathbf{G}^T(\mathbf{x})\mathbf{P}(\mathbf{x}), \\ \mathbf{K}_s(\mathbf{x}) &= -\mathbf{R}^{-1}(\mathbf{x})\mathbf{G}^T(\mathbf{x})\mathbf{A}_{cl}^{-T}(\mathbf{x})\mathbf{Q}(\mathbf{x}) \end{aligned} \tag{23}$$

which makes the closed-loop system locally asymptotically stable where $\mathbf{A}_{cl} = \mathbf{A}(\mathbf{x}) - \mathbf{G}(\mathbf{x})\mathbf{R}^{-1}(\mathbf{x})\mathbf{G}^T(\mathbf{x})\mathbf{P}(\mathbf{x})$ denotes the closed-loop transition matrix [24]. The matrix $\mathbf{P}(\mathbf{x})$ is a unique, symmetric, and positive-definite solution of the SDRE

$$\begin{aligned} &\mathbf{P}(\mathbf{x})\mathbf{A}(\mathbf{x}) + \mathbf{A}^T(\mathbf{x})\mathbf{P}(\mathbf{x}) + \mathbf{Q}(\mathbf{x}) \\ &\quad -\mathbf{P}(\mathbf{x})\mathbf{G}(\mathbf{x})\mathbf{R}^{-1}(\mathbf{x})\mathbf{G}^T(\mathbf{x})\mathbf{P}(\mathbf{x}) = 0 \end{aligned} \tag{24}$$

Since the SDC matrix $\mathbf{A}(\mathbf{x})$ is non-unique, it provides flexibility in design of the SDRE control. That is because it functions as a gain in addition to the weighting matrices $\mathbf{Q}(\mathbf{x})$, $\mathbf{R}(\mathbf{x})$ and enables the adjustment of the performance, stability, and optimal performance of the closed loop system [25].

Remark 2. *The control law in (22) requires the matrix $\mathbf{A}_{cl}(\mathbf{x})$ to be full-rank for the computation of the gain $\mathbf{K}_s(\mathbf{x})$. In addition, all the eigenvalues of the matrix $\mathbf{A}_{cl}(\mathbf{x})$ must have negative real parts in order to obtain asymptotic stability.*

Remark 3. *For the asymptotic stability of the closed-loop system, the solution $\mathbf{P}(\mathbf{x})$ of the Riccati equation in (24) must be unique, symmetric and positive definite. For this to hold it is required that the pair $(\mathbf{A}(\mathbf{x}), \mathbf{G}(\mathbf{x}))$ is pointwise stabilizable and the pair $\left(\mathbf{A}(\mathbf{x}), \mathbf{Q}^{1/2}(\mathbf{x})\right)$ is pointwise detectable on an open bounded set as stated in [26].*

IV. Simulation Results

A. Simulation Platform

For testing the performance of the developed SDRE controller, SoMo framework, is used. SoMo is a software that can be used to control a continuum manipulator in a physics simulator. The software is a soft gripper simulation consisting of 4 fingers. Objects can be manipulated using the soft gripper. The simulator is fast, open-source, and simple to use for simulation studies. Hence, SoMo provides an accessible way

to test control algorithms for soft robots. In SoMo each of the four fingers are modeled as 10-link continuum manipulators. The dynamic model of a 10-link continuum manipulator is given by the standard E-L equation as described in Section II. Each manipulator takes only two input torques which is represented using the control input $u \in \mathbb{R}^2$. Thus, the control effectiveness matrix B in (4) is $B \in \mathbb{R}^{10\times 2}$. A snapshot of SoMo simulation is shown in Fig. 1. SoMo automatically builds rigid-link approximations of continuum manipulators using a standard format (universal robot description file, or URDF), and imports them into the PyBullet physics engine.

For this paper, the four fingers of the gripper were used to test the SDRE controller performance. Since the continuum model of the manipulator and high degree of freedom model of the manipulator is complex for deriving a SDC form, the SDC-form model of the manipulator is learned by collecting data from the SoMo simulator. For identifying the model in task space, the tuple $\left\{\mathbf{x}^i(t), u^i(k)\right\}_{t=0,i=1}^{t=10,i=4}$ implying four different input and state trajectories of length 10 s are collected at a sampling rate of $\Delta t = 0.001$s. For training the NN model in Section II-B, the corresponding accelerations $\ddot{x}^i(t)$ are required to compute $\dot{\mathbf{x}}^i(t)$. The accelerations are computed using a Kalman Filter (KF). The equations of the implemented KF has been shown in the appendix. The task space data is collected for one finger by applying series of hand designed control inputs. The following control inputs $u_1(t)$, $u_2(t)$, $u_3(t)$ and $u_4(t)$ were provided as input torques to the manipulators:

$$\begin{aligned} u_1(t) &= [25s(\pi t),\ 25s(\pi t)]^T \\ u_2(t) &= [25s(0.5\pi t) + 2,\ 25s(0.5\pi t) + 2]^T \\ u_3(t) &= [10s(2\pi t) + 15s(\pi t),\ 10s(2\pi t) + 15s(\pi t)]^T \\ u_4(t) &= [6s(3\pi t) + 7s(2\pi t) + 8s(\pi t), \\ &\qquad 6s(3\pi t) + 7s(2\pi t) + 8s(\pi t)]^T \end{aligned} \tag{25}$$

where $s(\cdot)$ denotes the $\sin(\cdot)$ function.

B. Details of Model Learning

For each of the control inputs $u_1(t)$, $u_2(t)$, $u_3(t)$ and $u_4(t)$ data is collected over an interval of 10000 time steps each of which has a duration of Δt. The collected data is segregated into two datasets viz. training dataset and validation dataset. The 30000 instances of data corresponding to $u_1(t)$, $u_2(t)$ and $u_3(t)$ collectively formed the training data. The neural networks in Section II-B are trained using the training dataset. The validation data is composed of 10000 instances of data that were collected when the control input $u_4(t)$ is applied as the input torque to the manipulator. $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ in (14) are represented as $\mathbf{NN}_{\theta_1}$ and $\mathbf{NN}_{\theta_2}$, respectively as shown in (16) and (17). The first neural network $\mathbf{NN}_{\theta_1}$ was composed of three fully connected layers with hidden layer dimensions of 32. The second neural network $\mathbf{NN}_{\theta_2}$, consisted of three fully connected layers with hidden layer dimension of 32. In both the networks, each layer except the last layer is followed by the rectified linear unit activation function. The networks were trained and validated using the PyTorch library on a computer with 32 GB RAM, Intel i7 processor and a NVIDIA RTX 2080 GPU. For training phase, the ADAM optimizer was used with a learning rate of 10^{-5} for 70 epochs. The training loss reduction for the specified training parameters and the training dataset is shown in Figure 2. The predicted accelerations vs. the true accelerations are shown in Figure 3 where the neural network acceleration predictions are denoted by $\ddot{\hat{x}}(t) = \begin{bmatrix} \ddot{\hat{X}}(t) & \ddot{\hat{Y}}(t) \end{bmatrix}^T$.

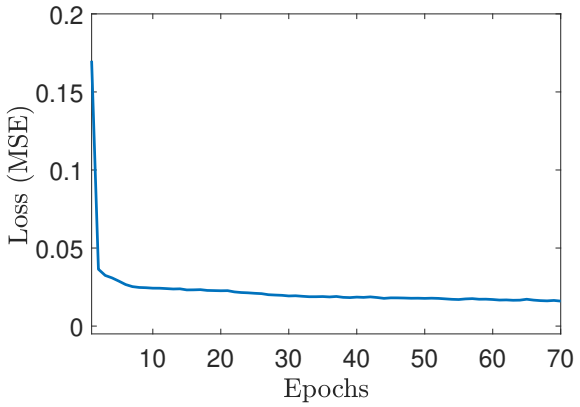

Figure 2. Training loss shown as a function of the number of epochs.

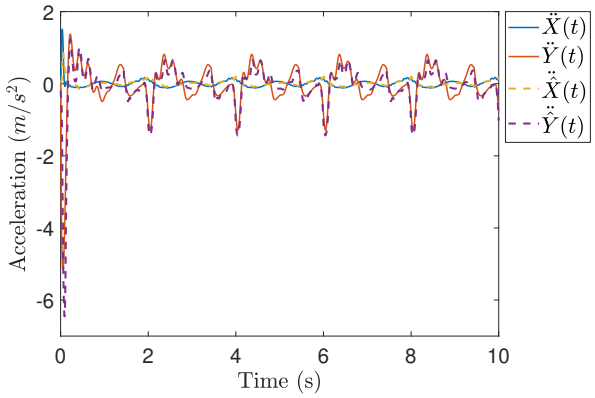

Figure 3. The acceleration predicted by the trained neural network model (dashed) vs. the true acceleration (solid) using the validation dataset.

C. Trajectory Tracking Results

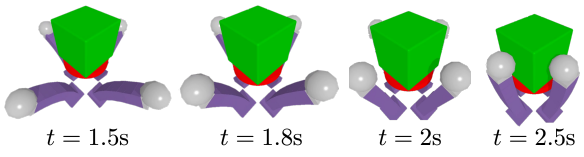

Figure 4. Snapshots of trajectory tracking result from the SoMo simulator for the cube grasping task.

Once the SDC model in Section II-B has been learned using the neural networks $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ can be obtained at every time step by querying the state as an input to the two neural networks. Using computed $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$, the closed-loop tracking controller developed in Section II is implemented. The desired trajectory chosen for the tracking controller is derived from the practical use case of the soft gripper opening and closing for grasping. In this trajectory, two opposite fingers move close to the object (green) in Figure 1 to grasp the object and the move away from the object thereby releasing the grasp. The state of each of the fingers is measured and the controller is computed for each finger separately using the developed control law (22). Although the controller is computed separately based on the state of each finger, only two neural networks are trained to model the SDC parameters as all the fingers follow the same dynamics. To solve the SDRE in (24), $\mathbf{Q} = \text{diag}\,(2, 2, 2, 2)$ and $\mathbf{R} = \text{diag}\,(0.1, 0.1)$ are chosen as the parameters, where $\text{diag}\,(\cdot)$ refers to the diagonalization operation. The results of the trajectory tracking simulation are shown in Figures 5-7. The desired finger end-point position trajectories (solid) in comparison to the actual position trajectories (dashed) is shown in Figure 5. The snapshots in Figure 4 show that the grasping motion of the fingers between $1.5-2$s. The fingers track a sinusoidal motion to open and close the gripper in X and Y direction. The position remains constant in the Y direction when the finger touches the green cube as shown in Figure 5 between $2-3$s. Similarly, the desired finger end-point velocity trajectories (solid) in comparison to the actual velocity trajectories (dashed) is shown in Figure 6. Each of the two control input torques generated by the feedback controller are shown in Fig. 7.

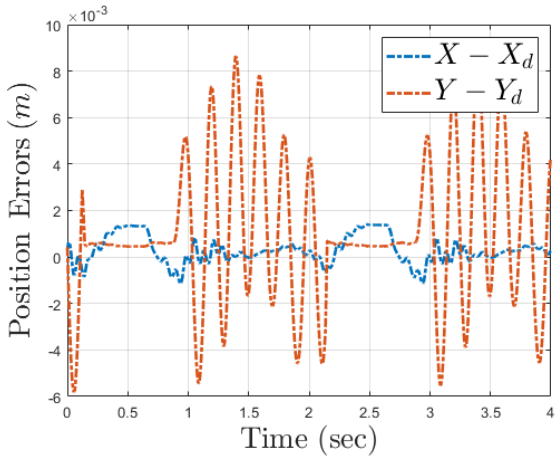

Figure 5. The desired finger end-point position trajectories (solid) and actual end-point position trajectories (dashed).

D. Discussion of Results

The trajectory tracking results obtained in Section IV-C rely on the accuracy of predictions of $\mathbf{A}(\mathbf{x})$ and $\mathbf{G}(\mathbf{x})$ generated by the data-driven NNs model. Deriving accurate

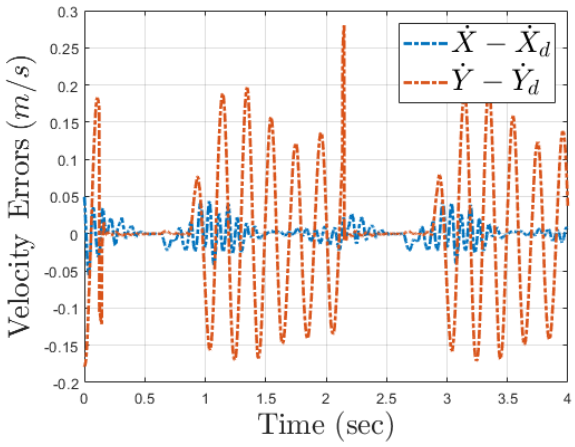

Figure 6. The desired finger end-point velocity trajectories (solid) and actual end-point velocity trajectories (dashed).

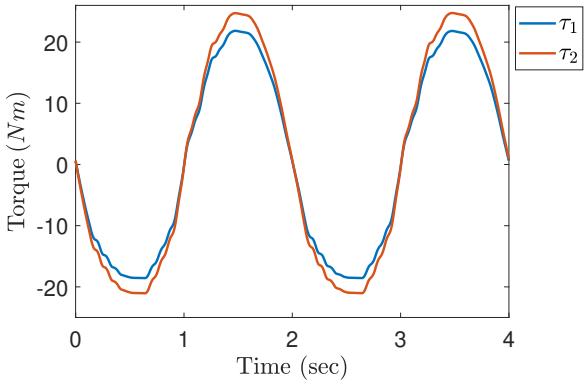

Figure 7. The control input torques generated by the SDRE controller.

first principles physics-based model of the soft robot is a challenging task and there are always unmodeled effects present in the learned dynamics which makes the trajectory tracking task challenging. The SDRE controller is tuned to account for such modeling uncertainties and measurement uncertainties using Q and R parameters. Although no explicit controllability and observability constraints are imposed during the NN training, the controllability of the pair $(\mathbf{A}(\mathbf{x}), \mathbf{G}(\mathbf{x}))$ and the observability of the pair $\left(\mathbf{A}(\mathbf{x}), \mathbf{Q}^{1/2}(\mathbf{x})\right)$ is checked at every iteration to ensure that the SDRE in (24) has a unique symmetric positive definite solution. The existence a solution to SDRE also ensures that the closed-loop transition matrix $\mathbf{A}_{cl}(\mathbf{x})$ is pointwise Hurwitz stable. During the simulations it is found that for sufficiently exciting torque inputs during SDC model identification process, the controllability, observability and closed-loop stability conditions are satisfied for each time instant. To implement the control law a constant velocity KF is used with the position measurements obtained from the simulator to generate the state estimate. This ensures stable tracking performance in the presence of process noise and measurement noise.

V. Conclusion

A data-driven modeling and SDRE control approach is developed for the task space tracking control of a soft robot gripper. The dynamical model of each finger on the soft gripper is learned using two neural networks in the SDC form. The learned SDC model is used for closed-loop tracking control which is computed by solving the SDRE at each time step. The proposed framework demonstrates promising results for task space trajectory tracking when implemented on the open source SoMo soft gripper simulation platform. Future research will involve addition of observability and controllability constraints in the SDC learning and choosing state dependent $\mathbf{Q}$ and $\mathbf{R}$ parameters for SDRE control to further improve the modeling and tracking precision.

References

[1] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Applied Bionics and Biomechanics*, vol. 5, no. 3, pp. 99–117, 2008.

[2] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[3] A. D. Kapadia, I. D. Walker, D. M. Dawson, and E. Tatlicioglu, "A model-based sliding mode controller for extensible continuum robots," in *WSEAS International Conference on Signal processing, Robotics and Automation*, 2010, pp. 113–120.

[4] A. Kapadia and I. D. Walker, "Task-space control of extensible continuum manipulators," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 1087–1092.

[5] T. George Thuruthel, Y. Ansari, E. Falotico, and C. Laschi, "Control strategies for soft robotic manipulators: A survey," *Soft robotics*, vol. 5, no. 2, pp. 149–163, 2018.

[6] B. A. Jones and I. D. Walker, "Kinematics for multisection continuum robots," *IEEE Transactions on Robotics*, vol. 22, no. 1, pp. 43–55, 2006.

[7] G. Krishnan, J. Bishop-Moser, C. Kim, and S. Kota, "Kinematics of a generalized class of pneumatic artificial muscles," *Journal of Mechanisms and Robotics*, vol. 7, no. 4, p. 041014, 2015.

[8] D. Trivedi, A. Lotfi, and C. D. Rahn, "Geometrically exact models for soft robotic manipulators," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 773–780, 2008.

[9] E. Tatlicioglu, I. D. Walker, and D. M. Dawson, "Dynamic modelling for planar extensible continuum robot manipulators," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1357–1362.

[10] G. S. Chirikjian, "Hyper-redundant manipulator dynamics: A continuum approximation," *Advanced Robotics*, vol. 9, no. 3, pp. 217–243, 1994.

[11] G. Olson, R. L. Hatton, J. A. Adams, and Y. Mengüç, "An euler–bernoulli beam model for soft robot arms bent through self-stress and external loads," *International Journal of Solids and Structures*, vol. 207, pp. 113–131, 2020.

[12] M. Thieffry, A. Kruszewski, O. Goury, T.-M. Guerra, and C. Duriez, "Dynamic control of soft robots," in *IFAC World congress*, 2017.

[13] V. Vikas, P. Grover, and B. Trimmer, "Model-free control framework for multi-limb soft robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 1111–1116.

[14] D. Braganza, D. M. Dawson, I. D. Walker, and N. Nath, "A neural network controller for continuum robots," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1270–1277, 2007.

[15] M. T. Gillespie, C. M. Best, E. C. Townsend, D. Wingate, and M. D. Killpack, "Learning nonlinear dynamic models of soft robots for model predictive control with neural networks," in *IEEE International Conference on Soft Robotics*, 2018, pp. 39–45.

[16] D. Bruder, C. D. Remy, and R. Vasudevan, "Nonlinear system identification of soft robot dynamics using koopman operator theory," in *International Conference on Robotics and Automation*, 2019, pp. 6244–6250.

[17] D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, and R. Vasudevan, "Koopman-based control of a soft continuum manipulator under variable loading conditions," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6852–6859, 2021.

[18] W. McMahan, B. A. Jones, and I. D. Walker, "Design and implementation of a multi-section continuum robot: Air-octor," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 2578–2585.

[19] C. Lekakou, S. M. Mustaza, T. Crisp, Y. Elsayed, and C. M. Saaj, "A material-based model for the simulation and control of soft robot actuator," in *Annual Conference Towards Autonomous Robotic Systems*, 2017, pp. 557–569.

[20] M. A. Graule, C. B. Teeple, T. P. McCarthy, R. C. S. Louis, G. R. Kim, and R. J. Wood, "SoMo: Fast and accurate simulation of continuum robots in complex environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[21] C. B. Teeple, G. R. Kim, M. A. Graule, and R. J. Wood, "An active palm enhances dexterity of soft robotic in-hand manipulation," in *IEEE International Conference on Robotics and Automation*, 2021.

[22] Y.-W. Liang and L.-G. Lin, "Analysis of SDC matrices for successfully implementing the SDRE scheme," *Automatica*, vol. 49, no. 10, pp. 3120–3124, 2013.

[23] T. Cimen, "Survey of state-dependent Riccati equation in nonlinear optimal feedback control synthesis," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, 2012.

[24] T. Çimen, "Approximate nonlinear optimal SDRE tracking control," in *17th IFAC Symp. Automatic Control in Aerospace*, 2007, pp. 147–152.

[25] S. Kim and S. Kwon, "Nonlinear optimal control design for underactuated two-wheeled inverted pendulum mobile platform," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2803–2808, 2017.

[26] Y. Batmani, M. Davoodi, and N. Meskin, "Nonlinear suboptimal tracking controller design using state-dependent Riccati equation technique," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 5, pp. 1833–1839, 2017.

Appendix

The Kalman Filter (KF) used to obtain the state estimates for the implementation is described in brief.

A. Dynamic Model

The continuous-time dynamical system for $x(t)$ is given by $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{D}\tilde{v}(t)$ where

$$A = \begin{bmatrix} \mathbf{0}_{2\times2} & \mathbf{I}_2 \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix},\ \mathbf{D} = \begin{bmatrix} \mathbf{0}_{2\times1} \\ \mathbf{1}_2 \end{bmatrix},\ \mathbf{H} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times2} \end{bmatrix},$$

and $\mathbf{1}_2 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$ and $\tilde{v}(t)$ is a zero mean continuous white noise and $\mathbf{H}$ is the measurement matrix. The model is discretized and following KF equations are implemented.

1) Predict:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1}$$
$$\mathbf{S}_{k|k-1} = \mathbf{F}\mathbf{S}_{k-1|k-1}\mathbf{F}^T + \mathbf{\Sigma}_v$$

2) Update:

$$\mathbf{K}_k = \mathbf{S}_{k|k-1}\mathbf{H}^T\left(\mathbf{H}\mathbf{S}_{k|k-1}\mathbf{H}^T + \mathbf{\Sigma}_r\right)^{-1}$$
$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}\right)$$
$$\mathbf{S}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}\right)\mathbf{S}_{k|k-1}$$

where $\mathbf{F} = e^{A(\Delta t)}$, $\mathbf{S}_{k|k}$ is the covariance associated with the state estimate $\hat{\mathbf{x}}_{k|k}$, $\mathbf{\Sigma}_r$ is the measurement covariance and $\mathbf{\Sigma}_v$ is the process noise covariance.