16-745 Final Report: Data-Driven Modeling and Control of a Bioinspired Soft Grasper

Ravesh Sukhnandan
Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA
0000-0001-5858-9610

Nathan Zimmerer
Mechanical Engineering
Carnegie Mellon
Pittsburgh, PA
0009-0001-7239-4622

Sreeram Thirupathi
Robotics Institute
Carnegie Mellon
Pittsburgh, PA
sreethirupa@gmail.com

Helen Wang
Mechanical Engineering
Carnegie Mellon
Pittsburgh, PA
haohuiw@andrew.cmu.edu

Abstract—**Effective manipulation of soft objects poses unique challenges in robotics, such as ensuring delicate handling without compromising on precision. Applying optimal control techniques to the control of soft robots are appealing because of the under-actuated nature of many soft robots. However, such robots are difficult to accurately model analytically. In this report, we address these challenges through the use of a data-driven modelling technique to obtain linear dynamical models of a soft grasper. We then design a Linear Quadratic Regulator (LQR) to determine the optimal controls to track sinusoidally varying contact pressures on both a rigid and soft object. At low frequencies on the real hardware, the LQR controller tracks the desired trajectory closely. This research emphasizes the applicability of optimal control to bioinspired soft graspers, which can be useful for applications where gentle and precise manipulation of soft or delicate objects is needed, including food handling and medical robotics.**

Index Terms—**soft grasper, optimal control, time-embedding**

I. Introduction

Soft graspers have been increasingly employed in challenging environments, such as agriculture, where fruit and vegetables are often soft, slippery, and fragile [1]. As such, humans are often still required to harvest these crops by hand [2], [3]. Soft graspers can solve contact problems encountered in manipulating such objects by conforming to the object's surface [4], increasing the contact area and thus the frictional forces and torques which stabilize the object [5].

However, under certain conditions, even soft graspers may unintentionally deform fragile objects if only the passive compliance of the grasper is exploited [6], [7]. Combining the inherent passive compliance of soft graspers with the ability to actively control their actuation and morphology could help to further increase the applicability of such graspers in fields where manipulation of soft and fragile objects is required.

Optimal control theory presents an attractive framework that can yield appropriate commands for complex systems like soft robots. However, many traditional optimal control approaches, such as the linear quadratic regulator (LQR) and model-predicitive control (MPC) require a model of the system dynamics [8]. Soft robots often have properties that are difficult to model, such as hysteresis, non-linear stress strain behavior, viscoelasticity and near-infinite degrees of freedom [9]–[11].

Data-driven approaches that learn the dynamics of the system from experiments have become increasingly popular in soft robotics because it elides the need to have an analytical model [9], [12]. Moreover, optimal control techniques can then be used on the models obtained from data-driven approaches to create optimal control policies. While researchers have used such approaches in the position control of tentacle-like soft robots [13], [14], the use of data-driven models with optimal control on soft graspers to precisely control contact forces on deformable materials remains an open question.

Towards the grasping of fragile and deformable objects capable of active force control, we investigate the data-driven modelling and optimal control of a bioinspired soft grasper. We investigate this on a bio-inspired soft grasper mounted on a cartesian gantry (Fig. 1) [15]. The grasper can detect changes in pressure at the jaws using pneumatic sensors and expand or contract radially by applying pressure to circumferentially wrapped McKibben actuators to grasp an object. In this work, we present the characterization of the grasper's dynamics using a data-driven approach, apply optimal control to create an LQR control policy for both a rigid and elastic test object, and experimentally validate the LQR controller on the robot. We show that although the contact dynamics of the rigid and elastic cylinder are different, the LQR controller trained on the rigid cylinder can effectively track sinusoidally varying contact pressures on an elastic cylinder. With this work, we set the foundation to do the following:

- Explore the suitability of data-driven linear dynamical models for modeling the contact dynamics of our soft grasper.
- Investigate the robustness of the LQR control policy on objects of different material properties
- Create a benchmark to compare our existing neural-network based control (previously described in [16] and [15]) against.

II. Methods

Towards precision application of contact forces to fragile and deformable materials, we intend to track contact pressure trajectories exerted by the soft grasper on an object. The contact pressures ($p \in \mathbb{R}^3$), which serve as proxies for contact force, are measured at the soft grasper's three soft jaws (Fig. 1). The control inputs to the grasper ($u \in \mathbb{R}^3$) include the pneumatic pressure applied to the McKibben actuators (P_c) to contract the grasper radially (Fig. 1), and the grasper's x and y position.

To leverage modern controls approaches such as LQR in this application, a model of the system is required. Given that these algorithms yield a feedback control system, the model does not have to be perfect, it only needs to be accurate enough such that the controls are capable of compensating for divergent behavior from model inaccuracies. However, due to non-linearities present in both the actuation architectures and materials common to soft robotic systems, arriving at a suitable dynamics model can be challenging.

Koopman Operator Theory dictates that if the dimensionality of a system's observable state space is increased infinitely (ie *lifting* it), even a non-linear system can be represented linearly. Approaches to estimate this Koopman Operator by increasing the dimensionality of the observable state space to some higher finite dimensionality have been successfully demonstrated both with the use of a family of lifting functions [13] and by embedding the state space with its own time history [14]. Considering the difficulty with identifying an appropriate set of lifting functions, we set out to use a time delay embedding approach as a methodology for data-driven dynamic system identification via linear regression [17].

Our approach to gather appropriate datasets is described in II-A. Several approaches to the regression is discussed in II-B. To validate our modeling approaches, we also used them to estimate linear models for a simulated pendulum as its dynamics are non-linear, but intuitive (Appendix C). Designing controllers for the grasper that leveraged the models, both in a simulated environment and on real hardware is outlined in II-C.

A. Experimental Data

To determine the dynamics that related the inputs, $u = [P_c, x, y]^T$, to the outputs, $p = [P_1, P_2, P_3]^T$, we performed the following experiments where the grasper was randomly perturbed by step input to the controls (Fig. 1). Step changes were chosen because they allowed observation of both fast transient dynamics at the moment of the step change, as well as the static steady state dynamics during the hold time before the next step change. Similar experiments have been used to fit a Koopman model to a soft tentacle [14].

The object was placed on the grasping platform, with a pair of magnets used to secure it in position to prevent it moving during testing. At the $n-th$ trial, where $n \in [1..N]$, the control inputs u_k were pseudorandomly generated using numpy's `random` such that $u_{min} \leq u_n \leq u_{max}$. u was then held at these values for 15 seconds before the next pseudorandom control input, u_{n+1}, was executed.

The hardware used to control and digitize measurements from the soft grasper was previously described in [18]. Briefly, the actuation pressure P_c was maintained using a bang-bang pressure controller. The output contact pressures at the jaws, p_k, where $k \in [1..N \times f_s]$ and f_s is the sampling rate (Hz), was measured using digital pressure gauges [18]. Measurements and controls were executed at a rate of $f_s = 16$ Hz.

P_c was limited to $[6.5, 10]$ psi. The lower limit was selected because the grasper just started to made contact with the object at this pressure. The upper limit was chosen to prevent damage to the grasper during testing. The x, y and z positions of the grasper were manually set to be centered on the object. x and y were limited to moves of $[-5, 5]$ mm from the center of the object to avoid moving the grasper outside the range where it could reasonably maintain contact. The grasper height was fixed and was not perturbed during testing.

The pseudorandom step experiments were performed on cylinders (diameter 40 mm) (Fig. 1C). Two different materials were used. A rigid cylinder was 3D printed (Prusa MK4, PLA). An elastic cylinder was created using Ecoflex™00-30 silicone rubber which was poured into a 3D-printed mold and allowed to cure for 4 hours. A total of $N = 90$ pseudorandom trials were executed for each material type.

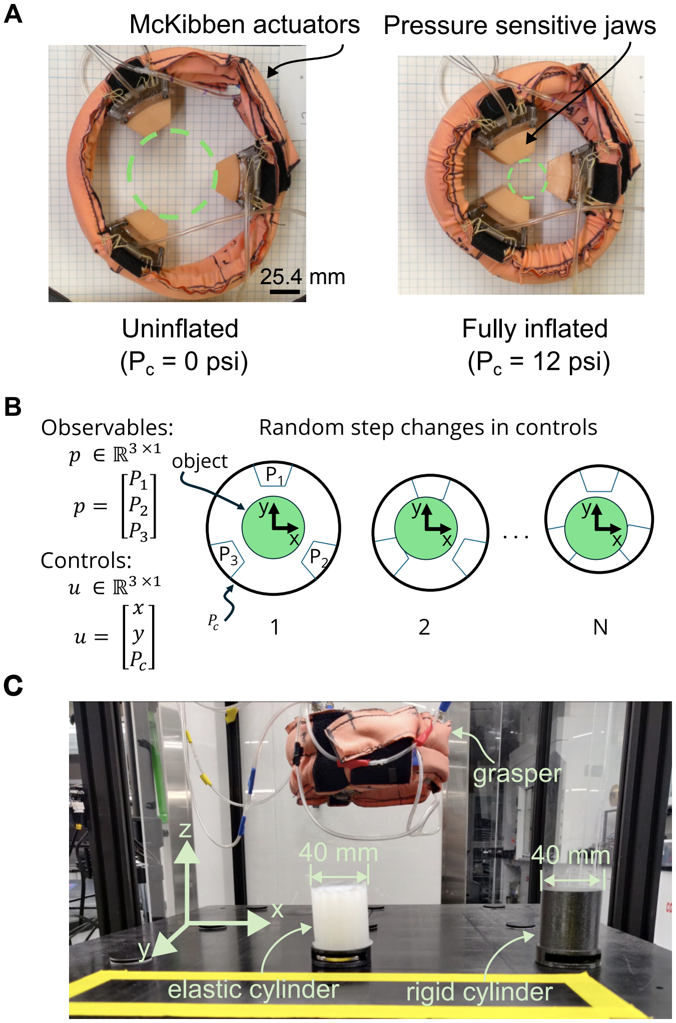

Fig. 1. **A** The bio-inspired soft grasper is capable of enveloping an object by actuating radially when pneumatic pressure is applied to the McKibben actuators. Elastic jaws are used to measure changes in contact pressures. **B** Data for the data-driven models was gathered using random step changes in controls for N trials. The controls were the applied McKibben pressure (P_c) and $[x, y]$ position of the grasper. The pressures at the jaws $[P_1, P_2, P_3]$ were the output. **C** Image of the actual experimental setup. Data were gathered on 40 mm diameter cylinders of rigid and elastic materials.

B. Linear Regression

With data gathered for each of the three jaw pressures given random inputs to the constricting McKibben actuator, we set out to identify a linear model that would adequately describe the system dynamics. This would require us to solve for A and B matrices for the typical linear discrete dynamics form:

$$z_{k+1} = Az_k + Bu_k \tag{1}$$

For our system, considering the jaw pressures as observable states and input pressure and position as a control, this would mean that $x \in \mathbb{R}^3$ and $u \in \mathbb{R}^3$. However, given our desire to work with the system in a higher dimensional state, such that the linear model better correlates for the non-linear system, a more apt representation of our application is:

$$\begin{aligned} z_{k+1} &= Az_k + Bu_k \\ p_k &= Cz_k \end{aligned} \tag{2}$$

In this form, z_k, A, and B all exist in some higher, lifted dimension while p_k is our observable set of jaw pressures ($p \in \mathbb{R}^3$) and C serves to map between the lifted state, z_k, and the observable state, p_k. The control input, u_k, is maintained in its observable form.

1) Matlab's N4SID: Matlab's System Identification Toolbox function `n4sid` estimates a state space model for a dynamic system as described by Eq. 2. In addition, it outputs an output error gain matrix, L, which can be used in a state observer to estimate the state z, which is not directly observable. The experimental data from contact testing performed with the 40 mm cylinder was fed into the function to provide it with the corresponding sets of controls, u_k, and jaw pressures, p_k. The algorithm then identifies a consistent set of dominant modes that govern the system dynamics by calculating the SVD associated with Hankel matrices of the data. Once the relative energy decays to within some tolerance of zero, it is likely that any further contributing modes are responsible for noise in the data as opposed to true underlying dynamics so it is prudent to exclude those extraneous modes. This presumes that the experimental data is clean and dynamic enough such that noise is not one of the dominant modes in the dataset (a common challenge for data driven system identification). Given our limited intuition of how many modes would be required to describe the grasper system, we specified a sweep across twenty modes.

2) Time Delay Embedding: In parallel with leveraging Matlab's built in tools for system identification, we attempted to design our own pipeline to linearly regress to a dynamics model. Generating our own custom pipeline provides us with more flexibility to partition our dataset based on free vs contact dynamics and more straightforward cross validation testing. Our algorithm was based on incorporating time delays of the observable state space which has been used in other works to model position of an end effector in a pneumatically actuated soft limb [14].

The linear regression is rooted in reformatting the discrete dynamics as an optimization problem for the matrices, A and B:

$$\min_{A,B} \sum_{k=1}^{m-1} ||Az_k + Bu_k - z_{k+1}||_2 \tag{3}$$

Minimizing each term of this summation with respect to an L2 norm is equivalent to minimizing the following system with respect to a Froebenius norm, where each vector, z_k, u_k, and z_{k+1} can be consolidated into matrices.

$$Z := \begin{bmatrix} z_1 & z_2 & ... & z_{m-1} \end{bmatrix} \tag{4}$$

$$U := \begin{bmatrix} u_1 & u_2 & ... & u_{m-1} \end{bmatrix} \tag{5}$$

$$Z^+ := \begin{bmatrix} z_2 & z_3 & ... & z_m \end{bmatrix} \tag{6}$$

$$\min_{A,B} || \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} Z \\ U \end{bmatrix} - Z^+ ||_F \tag{7}$$

This system can then be solved simply with a pseudoinverse operation:

$$\begin{bmatrix} A & B \end{bmatrix} = Z^+ \begin{bmatrix} Z \\ U \end{bmatrix}^\dagger \tag{8}$$

By incorporating some finite number of time delays, r, the dynamics model solved for via regression can serve as an estimate of the Koopman Operator for the system. This estimate approaches a true Koopman Operator as the number of time delays embedded into the system approaches infinite. Using an infinite number of time delays is infeasible so identifying a finite number of delays for the estimate is apt. Additionally, this helps guard against overfitting to training data and has computational benefits as the size of the solved for A matrix increases with the quantity of time delays embedded. To incorporate time delays, the state matrices are simply expanded as follows:

$$Z_{lift} := \begin{bmatrix} z_{r+1} & z_{r+2} & z_{r+3} & \cdots & z_{m-1} \\ \vdots & \vdots & \vdots & ... & \vdots \\ z_2 & z_3 & z_4 & \cdots & x_{m-r} \\ z_1 & z_2 & z_3 & \cdots & x_{m-1-r} \end{bmatrix} \tag{9}$$

$$Z^+_{lift} := \begin{bmatrix} z_{r+2} & z_{r+3} & z_{r+4} & \cdots & z_m \\ \vdots & \vdots & \vdots & ... & \vdots \\ z_3 & z_4 & z_5 & \cdots & z_{m-r+1} \\ z_2 & z_3 & z_4 & \cdots & z_{m-r} \end{bmatrix} \tag{10}$$

By inspection, we can see that there is some inherent sparsity built into the A and B matrices if we directly plug this into Eq. 8. For example, with two time delays embedded where the observables $\in \mathbb{R}^n$ and control input $\in \mathbb{R}^u$, the system takes on the form:

$$\begin{bmatrix} A_{i,nx3n} & B_{i,nxu} \\ I_{2nx2n} & 0_{2nx(n+u)} \end{bmatrix} \begin{bmatrix} Z_{lift} \\ U \end{bmatrix} = Z^+_{lift} \tag{11}$$

To take advantage of this, our implementation trims the problem down to:

$$\begin{bmatrix} A_{i,nx3n} & B_{i,nxu} \end{bmatrix} \begin{bmatrix} Z_{lift} \\ U \end{bmatrix} = Z^+ \tag{12}$$

Revisiting the general form described in Eq. 2, when using time delay embedding, the C matrix to map between the lifted state, z_k, and the observable state, p_k, is trivial:

$$p_k = \begin{bmatrix} I_{nxn} & 0_{nxnr} \end{bmatrix} \begin{bmatrix} z_k \; z_{k-1} \; \ldots \; z_{k-r} \end{bmatrix}^T \tag{13}$$

C. LQR Controller Design

Using our linear dynamics system model, we designed an LQR to track target contact pressure trajectories at each timestep k, $p_{ref,k} \in \mathbb{R}^3$. We chose an LQR because for a quadratic cost function with linear dynamics constraints (which is the case for our data-driven model), the LQR will produce the controls to optimally minimize the cost function.

In our problem formulation:

$$\begin{aligned}\min_{p_{1:N}, u_{1:N-1}} J = \sum_{k=1}^{N-1} \frac{1}{2}[(p_k - p_{ref,k})^T S(p_k - p_{ref,k}) \\ + u_k^T R u_k] \\ + \frac{1}{2}(p_N - p_{ref,N})^T S_N (p_N - p_{ref,N}) \\ \text{s.t. } z_{k+1} = A z_k + B u_k \\ p_k = C z_k \quad (14)\end{aligned}$$

where $S \in \mathbb{R}^{3x3}$. To make this compatible with LQR, we can substitute $p_k = Cz_k$ to yield:

$$\begin{aligned}\min_{z_{1:N}, u_{1:N-1}} J = \sum_{k=1}^{N-1} \frac{1}{2}[(Cz_k - p_{ref,k})^T S(Cz_k - p_{ref,k}) \\ + u_k^T R u_k] \\ + \frac{1}{2}(Cz_N - p_{ref,N})^T S_N (Cz_N - p_{ref,N}) \\ \text{s.t. } z_{k+1} = A z_k + B u_k \\ (15)\end{aligned}$$

When this problem is solved with dynamic programming (see Appendix A for the complete derivation), we obtain the following control policy with finite-horizon LQR for every step k:

$$u_k = u_{f,k} - K_k z_k \quad (16)$$

where $u_{f,k}$ is a feedforward term and $-K_k z_k$ is a feedback term.

1) Desired trajectory and LQR cost selection: To explore the grasper's performance on both slower and faster contact pressure trajectories, we defined pressure trajectories that linearly increased in frequency, f_k, with time from 0.001 Hz to 2 Hz over the span of 1000 seconds:

$$\begin{aligned}p_{ref,k} = [P_{1,ref,k}, P_{2,ref,k}, P_{3,ref,k}]^T \\ p_{ref,k} = [0.05, 0.015, 0.025]^T sin(2\pi f_k f_s k) + 0.06 \text{ psi} \quad (17)\end{aligned}$$

The minimum and maximum frequencies were chosen based on knowledge of the system, as it was unlikely that the grasper in its current forumlation would be able to track frequencies greater than 1 Hz. Similarly, the scale of the pressure sinusoids was based on knowledge of how the pressure changes with contact force, and was chosen to avoid squeezing the object too tightly which could potentially damage the grasper. Note that the amplitudes for the three jaws were not the same, and this was intentionally done to see the degree to which the contact pressures could be independently controlled. Because of the self-centering nature of the compliant grasper, and that the position of the grasper is expected to have limited influence on the contact pressures, we do not expect to achieve arbitrary pressure tracking independently.

Based on `n4sid`'s recommendation, the lifted state was a 7 element vector, i.e. $z \in \mathbb{R}^7$. After some testing, the values of S, R, and S_f that worked best with this FHLQR control were $S = \begin{bmatrix} 10000 & 0_{1\times2} \\ 0_{2\times1} & \mathbf{I}_6 \end{bmatrix}$, $R = diag(0.001, 0.1, 0.1)$, and $S_f = 10 * S$. This prioritized tracking the desired trajectory on jaw 1 (i.e. P_1), and so we would expect that jaw to have the best performance.

D. Experimental Validation of Controller

We validated the performance of the LQR controller tuned in Julia using the same experimental setup described in II-A. At each timestep, k, the LQR control policy is executed:

$$u_k = u_{f,k} - K_k \tilde{z}_k \quad (18)$$

where $u_{f,k}$ is the LQR feedforward term (Eq. 23), K_k is the LQR feedback gain (Eq. 23) and $\tilde{z_k}$ is the estimate of the latent state z_k.

Because we only observe the output, p_k, and not the latent state, z_k, we use a Luenberger observer to estimate the latent state [19]. The observer estimates z_{k+1} using our data-driven dynamics model:

$$\tilde{p}_k = C\tilde{p}_k \quad (19)$$

$$\tilde{z}_{k+1} = A\tilde{z}_k + Bu_k + L(p_k - \tilde{p}_k) \quad (20)$$

where $\tilde{p}_k$ is the estimate of p_k given our model and L is the output error gain matrix from `n4sid`. We initialize $\tilde{z}_k$ using $\tilde{z}_0 = C^{\dagger} p_0$.

With this control policy, three experiments were performed. In the first experiment, the LQR controller trained on the rigid cylinder was rolled out on the rigid cylinder (*Rigid LQR - Rigid Cylinder*). In the second experiment, the LQR controller trained on the soft cylinder was rolled out on the soft cylinder (*Soft LQR - Soft Cylinder*). In the third experiment, the LQR controller trained on the rigid cylinder was rolled out on the soft cylinder (*Rigid LQR - Soft Cylinder*). These experiments allowed us to validate how well the controllers did when tested on the same material type it was trained on, as well as on a material with dynamics different from what it was trained on.

III. Results And Discussion

A. Result of the dynamics fits

The outputs of the linear dynamics (Eq. 2) fit to the rigid and elastic cylinders largely approximate the measured response (Fig. 2). This was not necessarily the case for the dynamics fit to one type of material that was rolled out on the other type. For the elastic object, if the control inputs are input to the rigid cylinder dynamics model, then the model tends to over-predict the change in pressure (Fig. 2A). This aligns with the intution

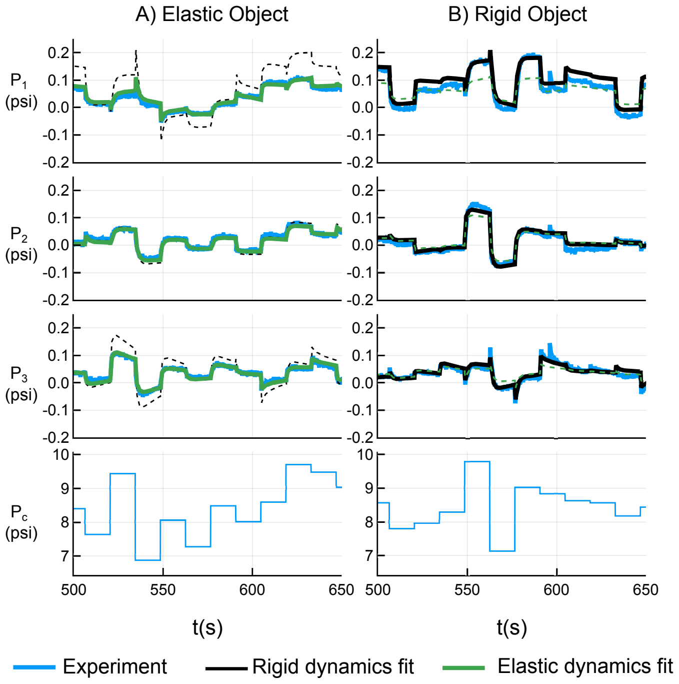

Fig. 2. Measured and simulated contact pressures (P_1, P_2, P_3) during random step experiments for the elastic object (**A**) and the rigid object (**B**). The simulated response of the n4sid models fit to the elastic (green line) and rigid cylinders (black line) are plotted. Only one of the controls (P_c) and 150 seconds of the collected data are shown for brevity. On the elastic object, the rigid dynamics tended to over-predict the contact pressure change. On the rigid object, the elastic dynamics tended to under-predict the pressure change.

for a simple spring model of contact, where for a given control input, a more rigid material will impart a greater reaction contact force for the same amount of identation. Since the contact pressure serves as a proxy for contact force, we would expect the contact pressure to be larger. Similarly, for the rigid object, the elastic cylinder dynamics model tended to under-predict the change in pressure (Fig. 2). For a spring model of contact, a less rigid material would produce less reaction force for a given amount of indentation. These results emphasize that the dynamics of the rigid and elastic materials are not the same, and may play a role in how well a controller trained on one material type may transfer to a different material type.

B. LQR performance in simulation

The simulated LQR policy on the data-driven dynamics models was able to track the desired trajectory on jaw 1 (Fig. 3). It was less successful on jaw 2 and 3, producing overshoot on jaw 2 and undershoot on jaw 3. This aligns with our expectations because the S matrix was biased towards penalizing tracking error on jaw 1, and because the dominant control input, P_c, affected all three jaws equally. This is corroborated by the magnitude of the control inputs, where the LQR computed control inputs for the position of the grasper change by at most 0.5 mm (Fig. 3B).

These results indicate that while the LQR can track time-varying contact pressure trajectories, it may not be possible to track contact pressures independently on each jaw with the grasper's current design. The ability to independently track contact pressures could be useful in certain manipulation contexts. For instance, handling objects like fruits that can have spatially varying stiffness related to levels of ripeness [20], or applying heterogenous contact forces for moving and orienting objects. Future work will look into modifications to the grasper's design to enable greater dexterity.

C. LQR performance on real hardware

On the physical hardware, the LQR controllers largely tracked the simulated LQR pressure trajectories at lower frequencies (Fig. 3A). On the elastic cylinder, this was true for both the LQR trained on the elastic dynamics and for the LQR trained on the rigid dynamics. This is an indication that for the range of frequencies tested, the control policy produced for either dynamics could work on the other material. Further testing is required to see if this property is true for other objects of varying size and material properties. At >1 Hz frequencies, the tracking performance degraded. We discuss this more in Appendix B.

There were some notable deviations between the simulated LQR response and the LQR on the physical system however. From the experiments, the LQR was not able to track the rise of the pressure at the very beginning of the frequency sweep, which resulted in a large overshoot (Fig. 3A, top-left). Moreover, the LQR would sometimes fail to track the rising limb of the sinusoid (Fig. 3A, top-left at 20 seconds), before eventually catching up. Such discrepancies could point to non-linear and discontinuous behavior of the grasper as it makes contact with the object that are not captured by the linear dynamics models presented here.

The controls, particularly the actuator pressure, P_c, also shows quite a large deviation from the simulated LQR control. While the simulated LQR P_c stays below the 12 psi limit (Fig. 3B, top-right), the actual P_c was usually much larger, and would occasionally exceed the 12 psi limit. These results provide further indication that more work should be done to obtain a more accurate model. Convex MPC may be implemented in the future to determine the optimum controls when faced with actuator limits, which LQR cannot natively reason about.

IV. Conclusions and Future Work

We demonstrate that the finite-horizon LQR policy trained on data-driven linear models can effectively track low frequency sinusoidal changes in contact pressures in both simulation and on the real-hardware. We show that an LQR policy trained on the rigid dynamics can be tried on the elastic dynamics with no qualitative loss in fidelity. This is exciting because of the potential to generalize a control policy to unseen material properties. Future work will explore the extent to which such feedback policies generalize to different materials of different sizes, including plastically deforable materials. Additionally, to guard against overfitting to a training dataset, cross validation testing shall be performed to validate the trained model on unseen test datasets.

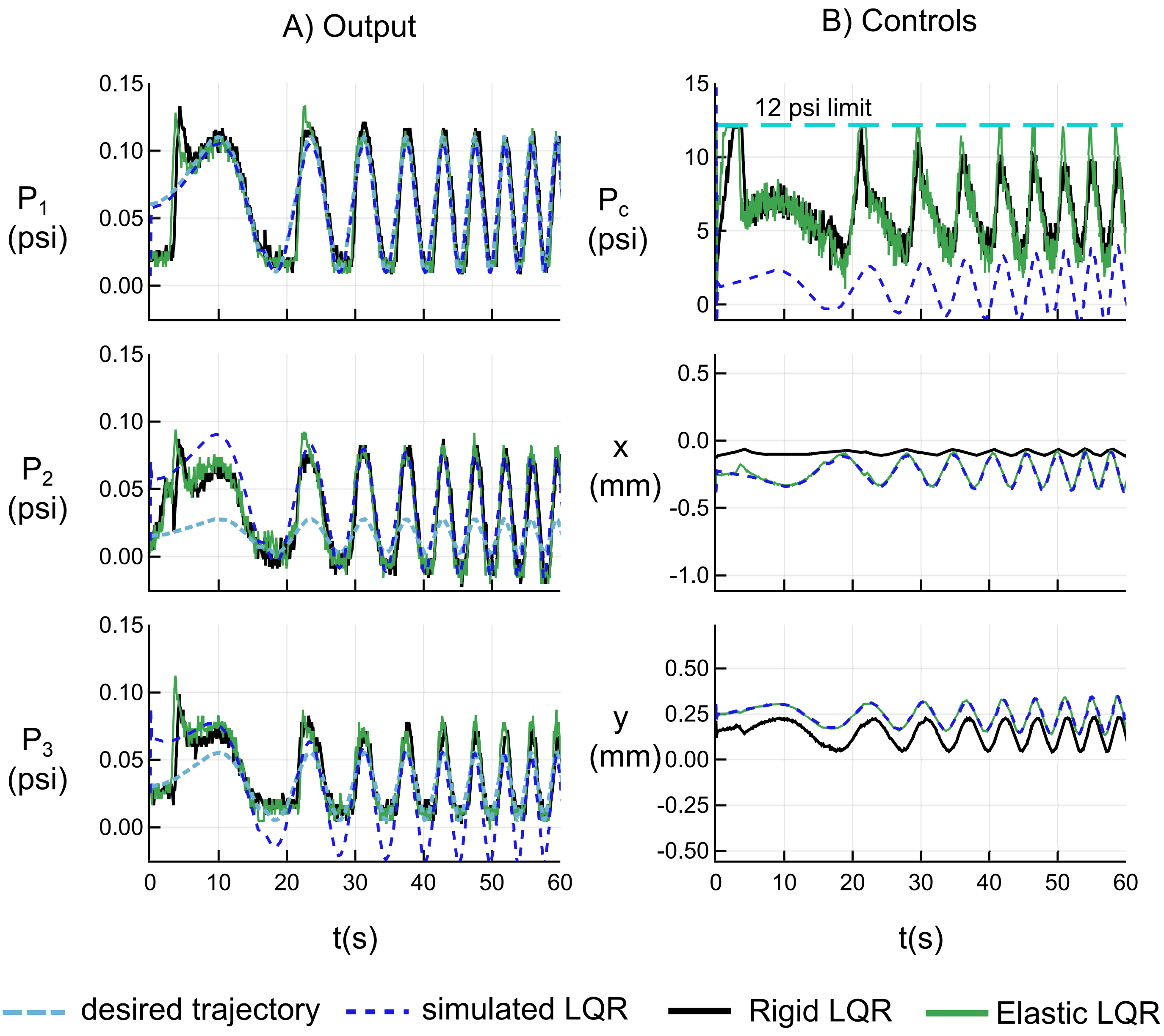

Fig. 3. Comparison of LQR contact pressure trajectories (**A**) and controls (**B**) on the elastic cylinder. Based on the biased weighting of the S matrix, the simulated LQR performance (blue dashed lines) closely follows the desired contact pressure trajectory for P_1(teal dashed lines). For P_2 and P_3, the simulated LQR trajectory does not achieve close tracking of the desired trajectories, which is expected because of the limited influence of the x and y controls on the dynamics (**B**, mid-right and bottom-right figures). On the physical hardware, both the LQR trained on the rigid dynamics (black line) and elastic dynamics (green line) largely follow the simulated trajectories (**A**). However, the actual actuator pressure, P_c, on the hardware was much larger than the simulated control inputs (**B**, top-right figure), occasionally reaching the maximum imposed pressure limit of 12 psi.

On the real system, performance degraded under certain conditions, such as when dealing with initial transients, when tracking the ascending limb of the sinusoids and when tracking higher-frequency sinusoids. This may be due to additional non-linearities and the discontinuous nature of contact that were not captured by the linear models. Future work will look at improving the tracking performance by modeling these non-linear phenomena such as through Koopman Theory, and including hybrid models of contact in the problem formulation. We will also pursue convex optimization based MPC on the hardware in order to better deal with actuation limits.

Finally, as it relates to tracking forces on plastically deformable objects like fruits, biological tissues and clays, we anticipate that we will need more ways to sense deformation and impart contact forces. Future work will look at further expanding the grasper's design to produce heterogeneous contact forces through additional actuation inputs, as well as tactile and vision based approaches for estimating the state of a deformable object while it undergoes irreversible deformation.

Acknowledgment

We would like to thank Prof. Zachary Manchester and the teaching staff of 16-745, particularly J.J. Lee, Arun Bishop and Kevin Tracy, for their guidance this semester. We would also like to thank Prof. Victoria Webster-Wood for allowing us to use the soft grasper for this project.

References

[1] B. Zhang, Y. Xie, J. Zhou, K. Wang, and Z. Zhang, "State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review," en, *Computers and Electronics in Agriculture*, vol. 177, p. 105 694, Oct. 2020, ISSN: 01681699.

[2] G. Kootstra, A. Bender, T. Perez, and E. J. van Henten, "Robotics in Agriculture," en, in *Encyclopedia of Robotics*, M. H. Ang, O. Khatib, and B. Siciliano, Eds., Berlin, Heidelberg: Springer, 2020, pp. 1–19, ISBN: 978-3-642-41610-1. DOI: 10.1007/978-3-642-41610-1_43-1. [Online]. Available: https://doi.org/10.1007/978-3-642-41610-1_43-1 (visited on 08/27/2023).

[3] A. Zahid, M. S. Mahmud, L. He, P. Heinemann, D. Choi, and J. Schupp, "Technological advancements towards developing a robotic pruner for apple trees: A review," en, *Computers and Electronics in Agriculture*, vol. 189, p. 106 383, Oct. 2021, ISSN: 01681699. DOI: 10.1016/j.compag.2021.106383. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0168169921004002 (visited on 08/27/2023).

[4] J. Shintake, V. Cacucciolo, D. Floreano, and H. Shea, "Soft Robotic Grippers," en, *Advanced Materials*, vol. 30, no. 29, p. 1 707 035, 2018, ISSN: 1521-4095.

[5] M. Ciocarlie, A. Miller, and P. Allen, "Grasp analysis using deformable fingers," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ISSN: 2153-0866, Aug. 2005, pp. 4122–4128.

[6] P. Paoletti, G. W. Jones, and L. Mahadevan, "Grasping with a soft glove: Intrinsic impedance control in pneumatic actuators," *Journal of the Royal Society Interface*, vol. 14, no. 128, p. 20 160 867, Mar. 2017, ISSN: 1742-5689. DOI: 10.1098/rsif.2016.0867. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5378126/ (visited on 05/06/2024).

[7] T. Nishimura, Y. Suzuki, T. Tsuji, and T. Watanabe, "Fluid Pressure Monitoring-Based Strategy for Delicate Grasping of Fragile Objects by A Robotic Hand with Fluid Fingertips," *Sensors (Basel, Switzerland)*, vol. 19, no. 4, p. 782, Feb. 2019, ISSN: 1424-8220. DOI: 10.3390/s19040782. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6412748/ (visited on 03/07/2023).

[8] A. Bryson, "Optimal control-1950 to 1985," en, *IEEE Control Systems*, vol. 16, no. 3, pp. 26–33, Jun. 1996, ISSN: 1066-033X, 1941-000X. DOI: 10.1109/37.506395. [Online]. Available: https://ieeexplore.ieee.org/document/506395/ (visited on 05/06/2024).

[9] C. Della Santina, C. Duriez, and D. Rus, "Model-Based Control of Soft Robots: A Survey of the State of the Art and Open Challenges," en, *IEEE Control Systems*, vol. 43, no. 3, pp. 30–65, Jun. 2023, ISSN: 1066-033X, 1941-000X. DOI: 10.1109/MCS.2023.3253419. [Online]. Available: https://ieeexplore.ieee.org/document/10136424/ (visited on 02/06/2024).

[10] E. Coevoet, A. Escande, and C. Duriez, "Soft robots locomotion and manipulation control using FEM simulation and quadratic programming," in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, Apr. 2019, pp. 739–745. DOI: 10.1109/ROBOSOFT.2019.8722815. [Online]. Available: https://ieeexplore.ieee.org/document/8722815 (visited on 05/06/2024).

[11] J. Wang and A. Chortos, "Control Strategies for Soft Robot Systems," en, *Advanced Intelligent Systems*, vol. 4, no. 5, p. 2 100 165, 2022, ISSN: 2640-4567.

[12] Z. Chen, F. Renda, A. L. Gall, *et al.*, "Data-Driven Methods Applied to Soft Robot Modeling and Control: A Review," en, *IEEE Transactions on Automation Science and Engineering*, pp. 1–16, 2024, ISSN: 1545-5955, 1558-3783. DOI: 10.1109/TASE.2024.3377291. [Online]. Available: https://ieeexplore.ieee.org/document/10477253/ (visited on 05/07/2024).

[13] D. Bruder, C. D. Remy, and R. Vasudevan, *Nonlinear System Identification of Soft Robot Dynamics Using Koopman Operator Theory*, en, arXiv:1810.06637 [cs], May 2019. [Online]. Available: http://arxiv.org/abs/1810.06637 (visited on 05/06/2024).

[14] D. Haggerty, M. Banks, E. Kamenar, *et al.*, "Control of soft robots with inertial dynamics," *Science Robotics*, vol. 8, Sep. 2023. DOI: 10.1126/scirobotics.add6864.

[15] R. Sukhnandan, Y. Li, Y. Wang, *et al.*, "Synthetic Nervous System Control of a Bioinspired Soft Grasper for Pick-and-Place Manipulation," en, in *Biomimetic and Biohybrid Systems*, F. Meder, A. Hunt, L. Margheri, A. Mura, and B. Mazzolai, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2023, pp. 300–321, ISBN: 978-3-031-38857-6. DOI: 10.1007/978-3-031-38857-6_23.

[16] Y. Li, R. Sukhnandan, J. P. Gill, H. J. Chiel, V. Webster-Wood, and R. D. Quinn, "A Bioinspired Synthetic Nervous System Controller for Pick-and-Place Manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 8047–8053. DOI: 10.1109/ICRA48891.2023.10161198.

[17] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.

[18] K. Dai, R. Sukhnandan, M. Bennington, *et al.*, "SLUG-BOT, an Aplysia-Inspired Robotic Grasper for Studying Control," en, in *Biomimetic and Biohybrid Systems*, A. Hunt, V. Vouloutsi, K. Moses, *et al.*, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2022, pp. 182–194, ISBN: 978-3-031-20470-8. DOI: 10.1007/978-3-031-20470-8_19.

[19] D. Luenberger, "Observers for multivariable systems," *IEEE Transactions on Automatic Control*, vol. 11, no. 2, pp. 190–197, Apr. 1966, Conference Name: IEEE Transactions on Automatic Control, ISSN: 1558-2523. DOI: 10.1109/TAC.1966.1098323. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1098323?casa_token=uRi59xl28zQAAAAA:aTlYAdRKQoo2gjtuIQcGvyoTne9G9n4_hQmewy6qhA6ghztSU-27ZOW8ag7t5mPaXpQtzYo (visited on 05/07/2024).

[20] L. Scimeca, P. Maiolino, D. Cardin-Catalan, A. P. d. Pobil, A. Morales, and F. Iida, "Non-Destructive Robotic Assessment of Mango Ripeness via Multi-Point Soft Haptics," in *2019 International Conference on Robotics and Automation (ICRA)*, ISSN: 2577-087X, May 2019, pp. 1821–1826. DOI: 10.1109/ICRA.2019.8793956.

Appendix A
LQR Derivation

Given our cost function, J, from Eq. 15:

$$\begin{aligned}\min_{p_{1:N},u_{1:N-1}} J = \sum_{k=1}^{N-1}\frac{1}{2}[(p_k - p_{ref,k})^T S(p_k - p_{ref,k})\\ +u_k^T R u_k]\\ +\frac{1}{2}(p_N - p_{ref,N})^T S_N (p_N - p_{ref,N})\\ \text{s.t. } z_{k+1} = Az_k + Bu_k\\ p_k = Cz_k\end{aligned}$$

Starting at $k = N$ and applying Bellman's Principle of Optimality, we can define our cost to go function, $V_k(z)$ to be:

$$V_N(z) = \frac{1}{2}(Cz_N - p_{ref,N})^T S_N (Cz_N - p_{ref,N})$$

which expands to:

$$\begin{aligned}V_N(z) = \frac{1}{2}(z_N^T C^T S_N C z_N - z_N^T C^T S_N p_{ref,N} -\\ p_{ref,N}^T S_N C z_{ref,N} + p_{ref,N}^T S_N p_{ref,N})\end{aligned}$$

which can be written as:

$$\begin{aligned}V_N(z) = \frac{1}{2}(z_N^T G_{1,N} z_N - z_N^T G_{2,N} p_{ref,N} -\\ p_{ref,N}^T G_{3,N} z_N + p_{ref,N}^T G_{4,N} p_{ref,N}^T)\end{aligned} \tag{21}$$

For the $N-1$ step, the V becomes:

$$\begin{aligned}V_{N-1}(z) = \min_{u_{N-1}} V_N(z) +\\ \frac{1}{2}(Cz_{N-1} - p_{ref,N-1})^T S(Cz_{N-1} - p_{ref,N-1})\\ +u_{N-1}^T R u_{N-1}\end{aligned} \tag{22}$$

Substituting the dynamics $z_N = Az_{N-1} + Bu_{N-1}$ into Eq. 21 and Eq. 22, and taking the gradient w.r.t. u_{N-1} and setting it to 0:

$$\nabla_{u_{N-1}} V_{N-1}(z) = 0$$

which yields:

$$\begin{aligned}u_{N-1} = -(R + B^T C^T S_N C B)^{-1}(B^T C^T S_N C A) z_{N-1}\\ +(R + B^T C^T S_N C B)^{-1} B^T C^T S_N p_{ref,N}\end{aligned} \tag{23}$$

which can be rewritten in the form of Eq. 16:

$$u_{N-1} = u_{f,N-1} - K_{N-1} z_{N-1} \tag{24}$$

where $u_{f,N-1}$ is the feedforward term that is a linear function of the desired next state, $p_{ref,N}$. $K_{N-1} z_{N-1}$ is a feedback term.

(25)

Substituting Eq. 23 into Eq. 22, we obtain:

$$\begin{aligned}V_{N-1} = \frac{1}{2}(z_{N-1}^T G_{1,N-1} z_{N-1} +\\ z_{N-1}^T G_{2,N-1} + G_{3,N-1} z_{N-1} + G_{4,N-1})\end{aligned} \tag{26}$$

where:

$$\begin{aligned}G_{1,N-1} = C^T S C + K_{N-1}^T R K_{N-1} +\\ (A - BK_{N-1})^T G_{1,N} (A - BK_{N-1})\end{aligned} \tag{27}$$

$$\begin{aligned}G_{2,N-1} = G_{3,N-1}^T = -C^T S \; p_{ref,N-1} +\\ (A - BK)^T G_{1,N} (G_{1,N} B u_{f,N-1} + G_{2,N})\end{aligned} \tag{28}$$

$$\begin{aligned}G_{4,N-1} = p_{ref,N-1}^T S \; p_{ref,N-1} + u_{f,N-1}^T B^T G_{1,N} B u_{f,N-1}\\ +u_{f,N-1}^T B^T G_{2,N-1} + G_{3,N-1} B u_{f,N-1} +\\ p_{ref,N}^T S_N \; p_{ref,N}\end{aligned} \tag{29}$$

Substituting $k+1$ for N and k for $N-1$ in Eqs. 23 and 26 - 29 yields the LQR control policy that can be recursively defined starting from $k = N$ in Eq. 21.

Appendix B
LQR performance at higher frequencies

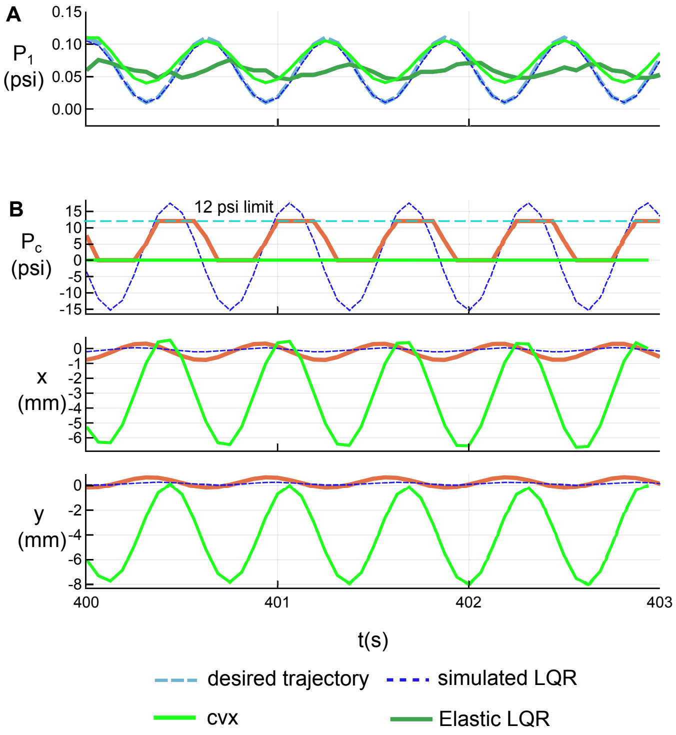

Fig. 4. LQR performance at higher frequencies on the real system. **A** The actual contact pressure at jaw 1 on the elastic cylinder with the LQR trained on the elastic dynamics (green line) shows a large deviation from either the simulated LQR response (blue lines) or the desired response (teal line). **B** Control inputs (P_c, x and y). The LQR policy on the real hardware results in the actuator pressure, P_c, reaching the control limits of 0 and 12 psi. With a convex optimization approach that incorporates these control limits (yellow line), the commanded jaw pressure is almost completely turned off, instead relying on moving the grasper in the xy plane rapidly.

At >1 Hz frequencies, the LQR policy tried on the on real hardware struggled to match the desired trajectory (Fig. 4A). In the simulated ideal LQR case, the tracking of contact pressure is near perfect even at higher frequencies because the actuator pressure control, P_c, oscillates beyond the physical bounds of 0 to 12 psi (Fig. 4B). Such a large swing in the control signal was not observed for the lower frequencies of Fig. 3. To accomodate for these control constraints, we performed convex optimization (using Julia's `convex` package with the ECOS solver) and simulated the system (yellow lines in Fig. 4). Interestingly, the optimal solution from the convex optimizer was to set P_c to virtually 0 psi, and instead move the grasper sinusoidally in x and y with much larger magnitudes than had been obtained in LQR. Future work will need to investigate the control policy from convex optimization more thoroughly to see if this indeed results in improved performance, or if it is exploiting inaccuacies in the model which will ultimately result in poor tracking performance on the harware. We believe the latter to be the case based on knowledge of the system. Future work will also look at improving the bandwidth of the system, such as through the use of higher flow rates and pressures greater than the imposed limit of 12 psi.

Appendix C
N4SID Performance vs Pure Time Embedding

As a means of validating our pipeline implementing time embedding as described in Section II-B2, we simulated the non-linear dynamics of a single DOF, unforced pendulum. We then used our algorithm across a range of time delays to fit linear models to the pendulum dynamics and saw accurate recreation of pendulum dynamics with three time delays (or more) embedded into the system (Fig. 5). We also compared the fit from our algorithm against that of the model generated by `n4sid` with 4 modes selected (a similar modeling DOF to a 3 time delay embedded system) which had comparable RMSE to our system. This indicates our approach with pure time delay embedding has validity at least in an environment where the data is noise free so when applying our approach to real data, more steps may need to be taken to filter out potential sources of noise. We will continue to work on this pipeline in the future to improve the fidelity of our data-driven models to the actual dynamics.

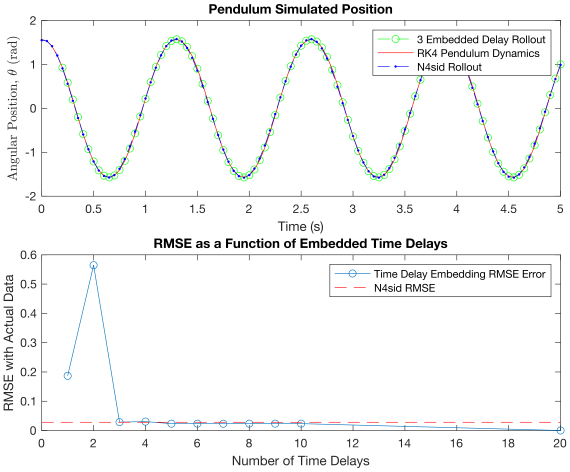

Fig. 5. With three or more time delays, the implemented time delay embedding pipeline was able to reproduce the dynamics of a simple pendulum with total mean-squared error comparable to that of `n4sid`.

Appendix D
Code

You can find the Julia notebook used to simulate the linear dynamics and compute the FHLQR here at: https://github.com/quantumRAV/16745_CourseProject.

The file is called grasper_LQR.ipynb.