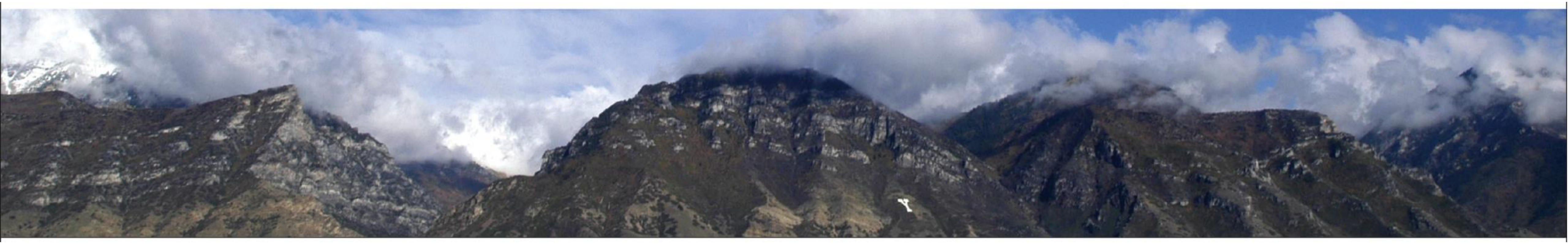
Model-based Control of Soft Robots: From First Principles to Learned Models

Marc D. Killpack

with work performed by members of the RAD lab

Aug 7, 2020

IEEE Robosoft Workshop on Modeling Soft Robots



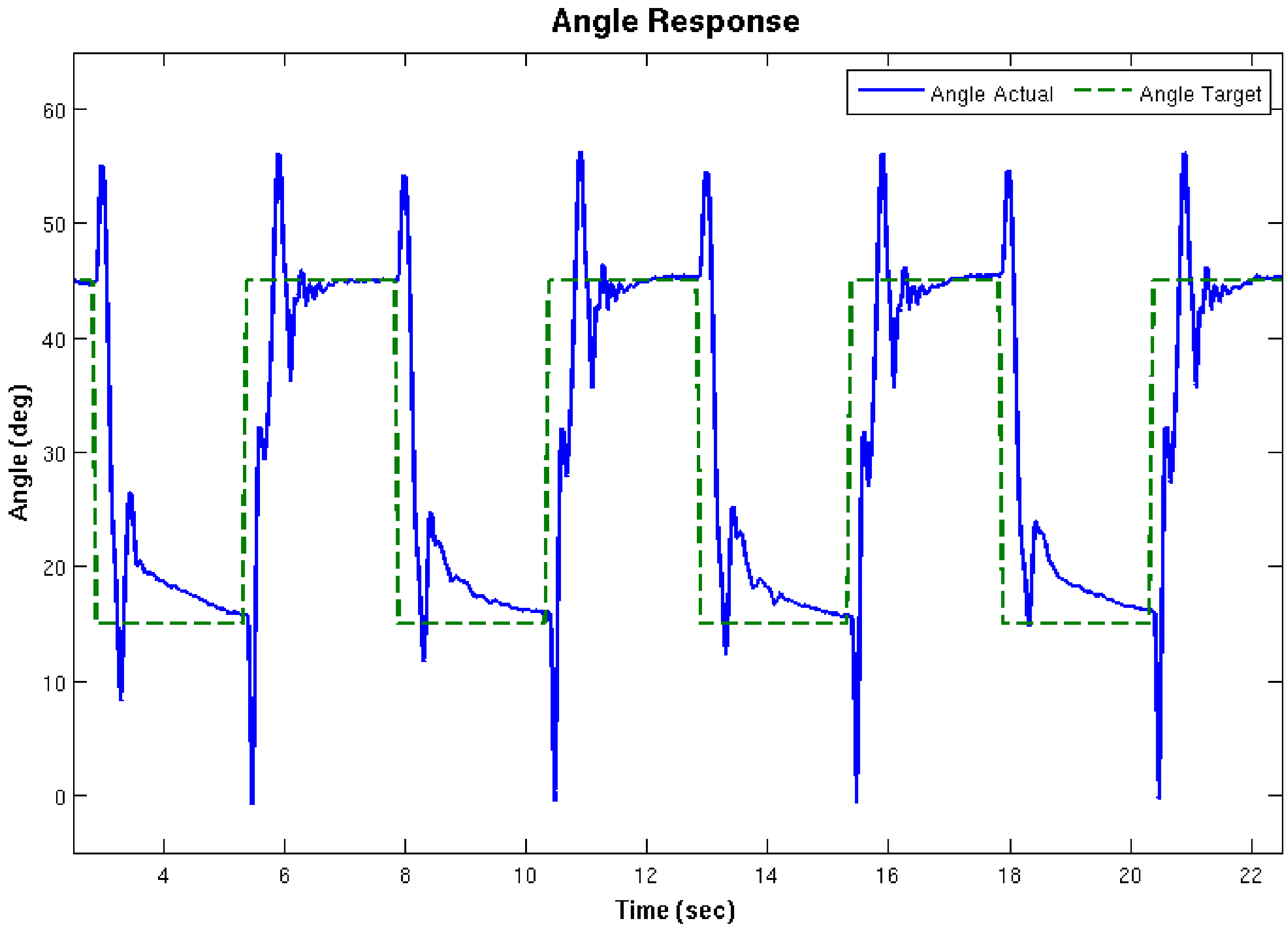
Acknowledgements

RAD LAB

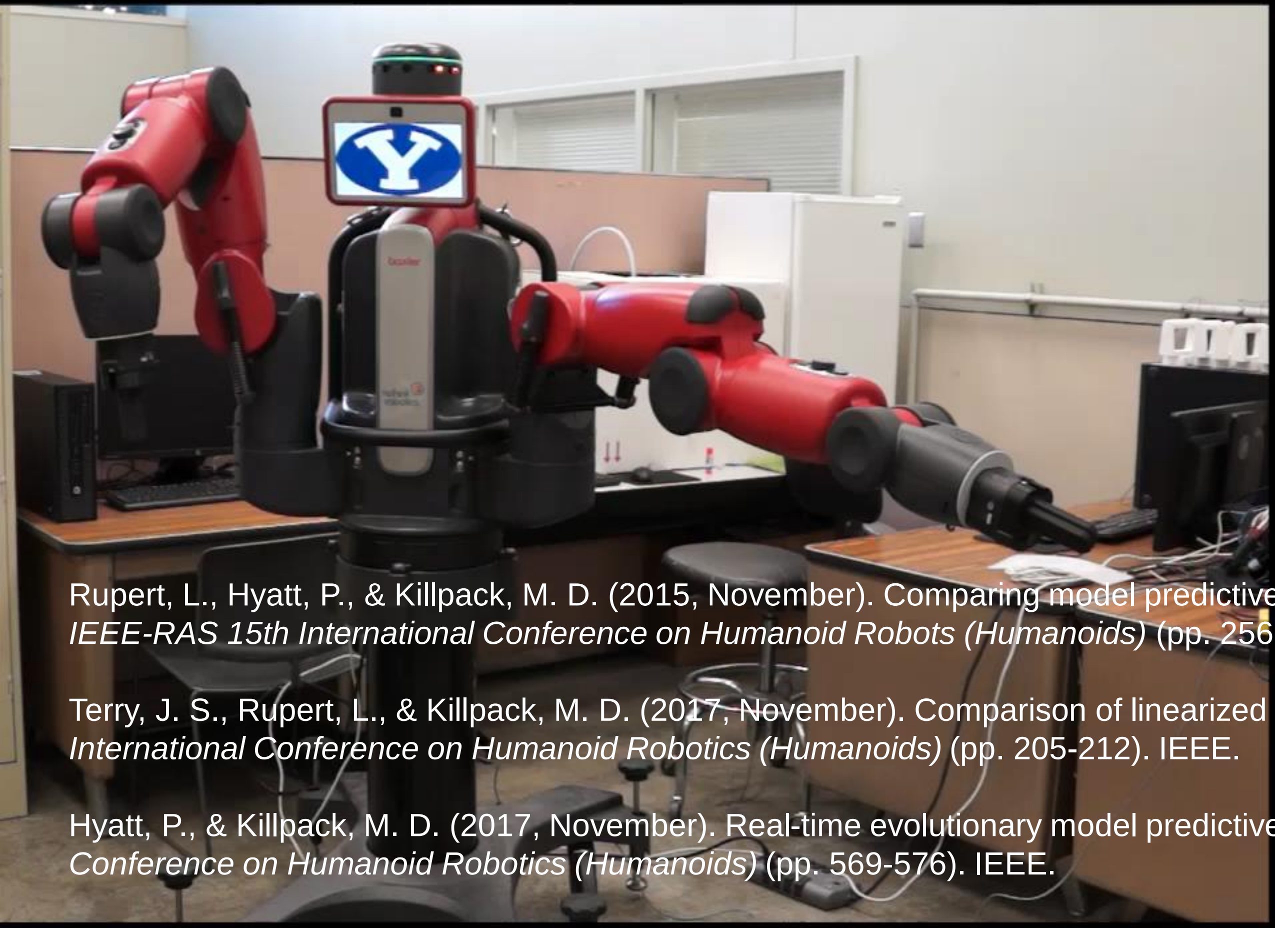




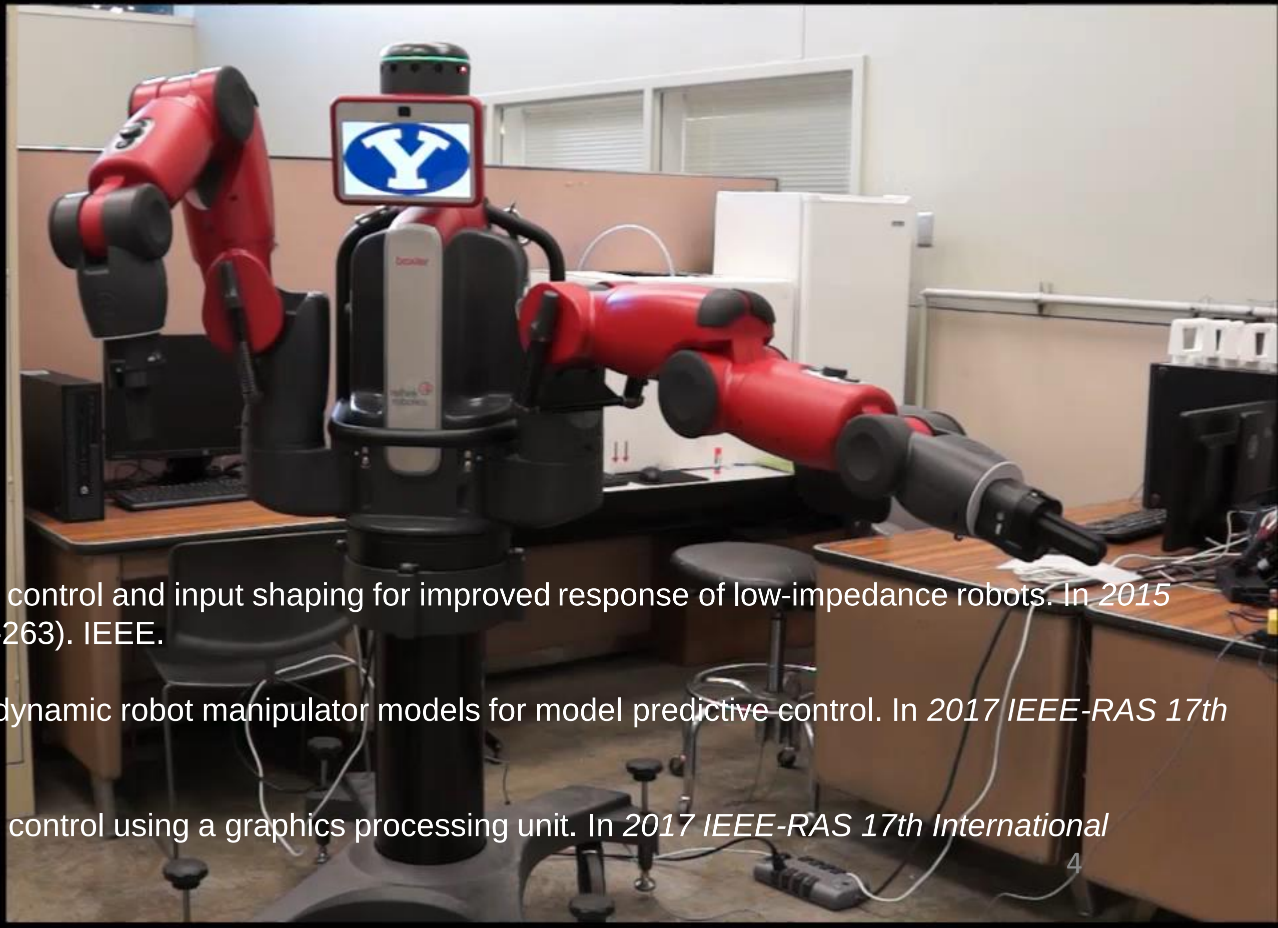

Why Model-based Control?



Impedance Control



Model Predictive Control



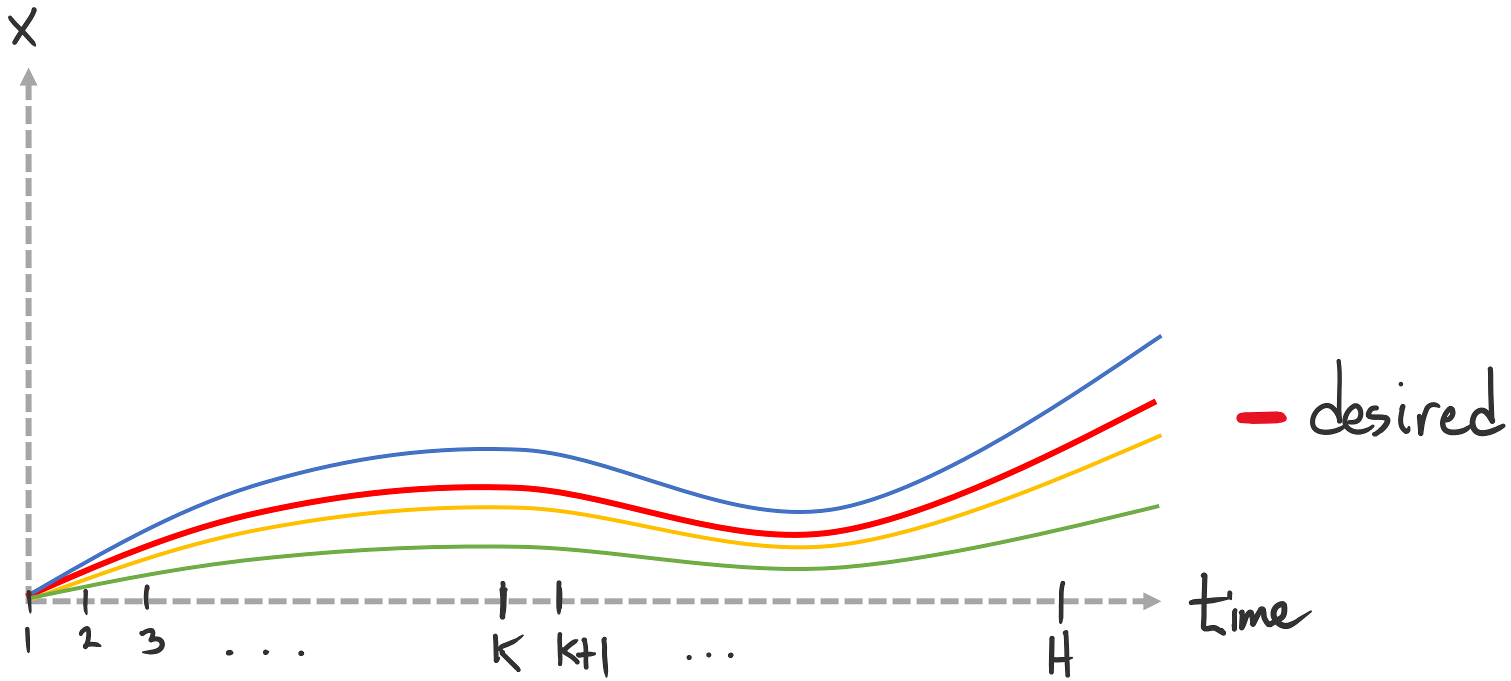
Rupert, L., Hyatt, P., & Killpack, M. D. (2015, November). Comparing model predictive control and input shaping for improved response of low-impedance robots. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)* (pp. 256-263). IEEE.

Terry, J. S., Rupert, L., & Killpack, M. D. (2017, November). Comparison of linearized dynamic robot manipulator models for model predictive control. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (pp. 205-212). IEEE.

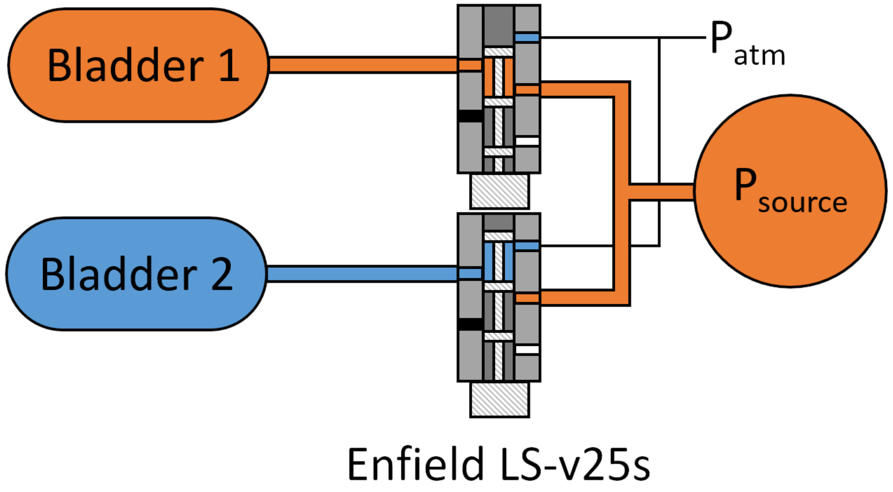
Hyatt, P., & Killpack, M. D. (2017, November). Real-time evolutionary model predictive control using a graphics processing unit. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (pp. 569-576). IEEE.

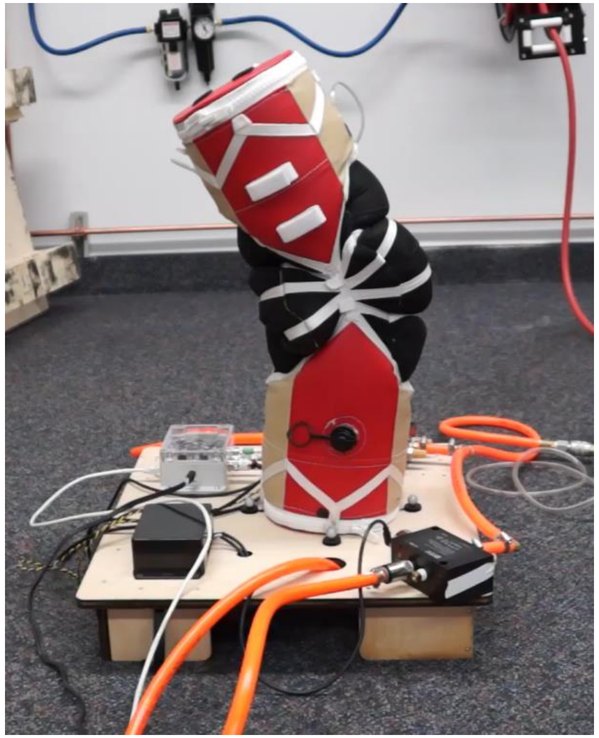
How Is the Dynamics Model Used?

$$\dot{x} = f(x, u) \quad \Rightarrow \quad x_{k+1} = f(x_k, u_k)$$

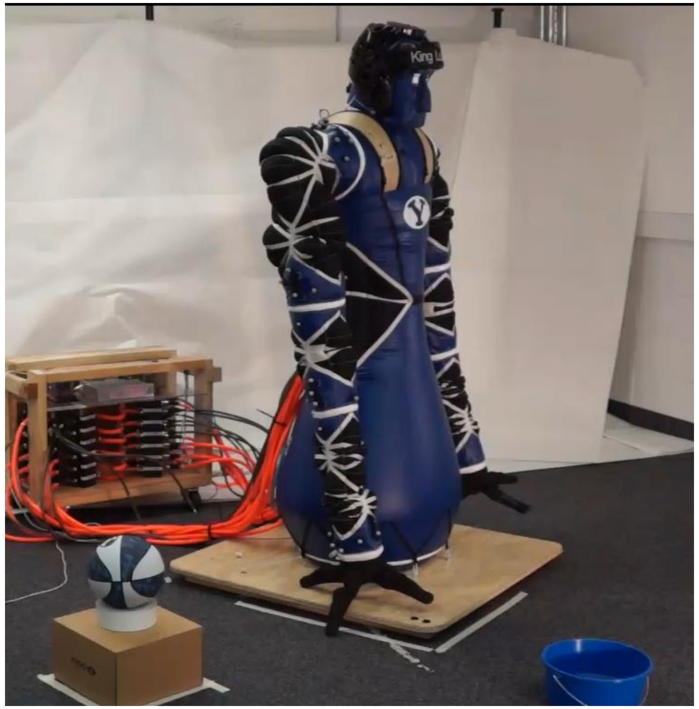


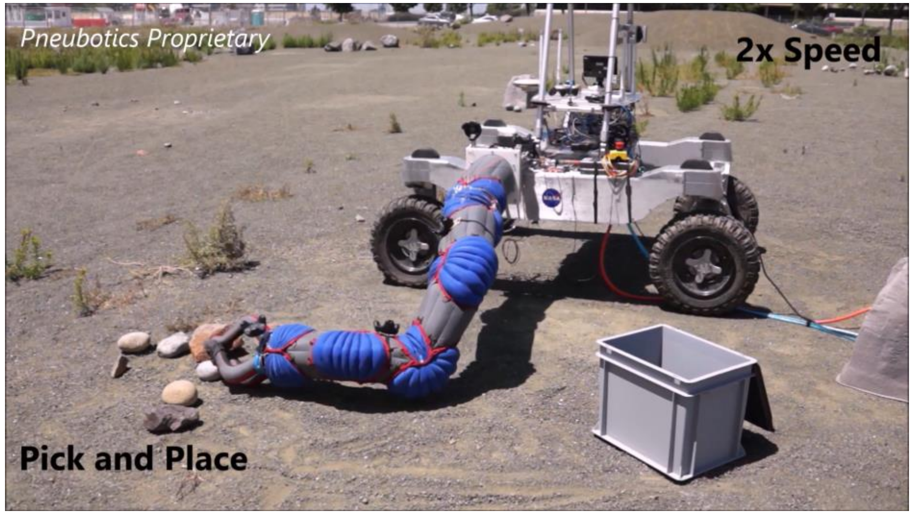
Pneumatic Joints

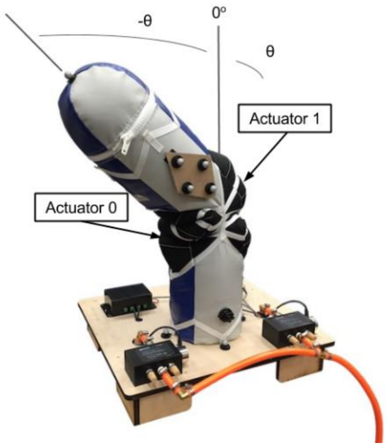


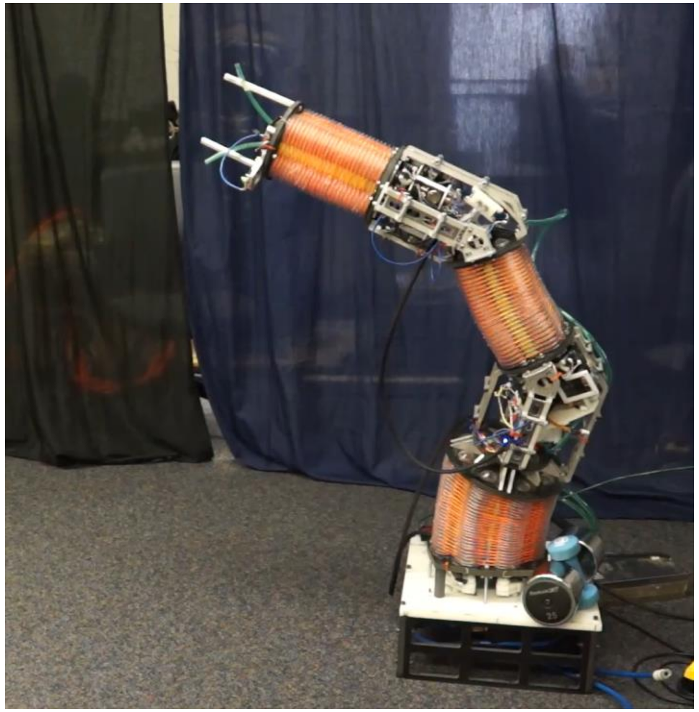


Hardware Platforms









Three different modeling approaches to enable control

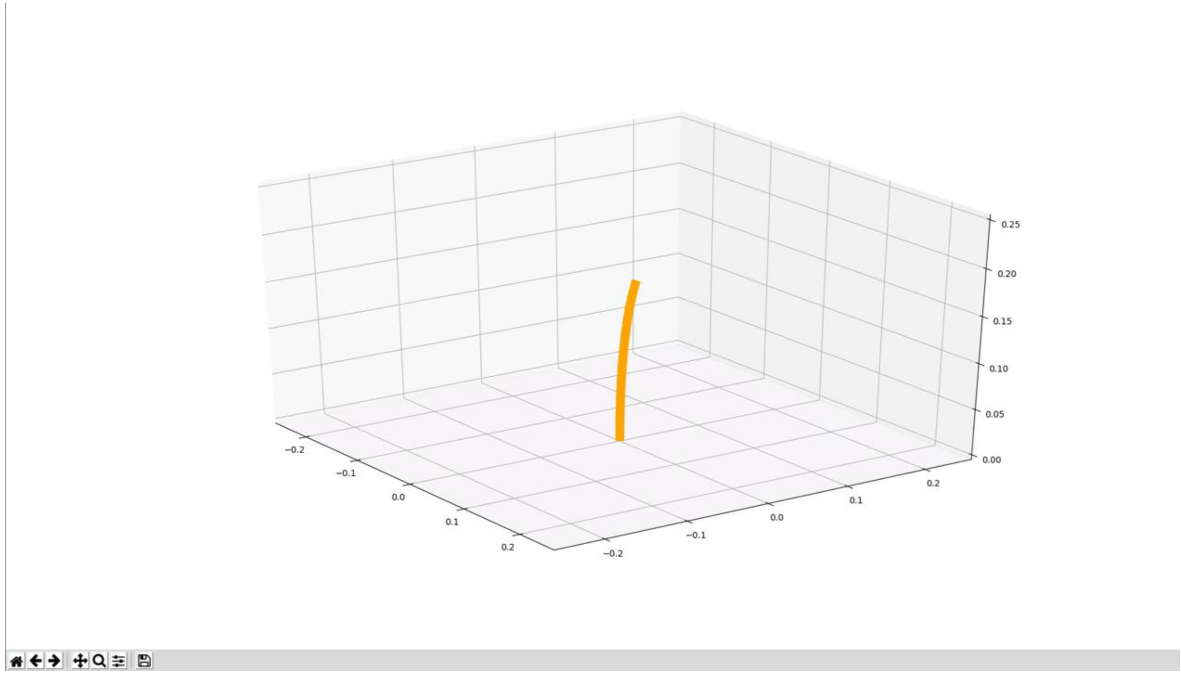
1. First principles modeling (based on physical phenomenon).

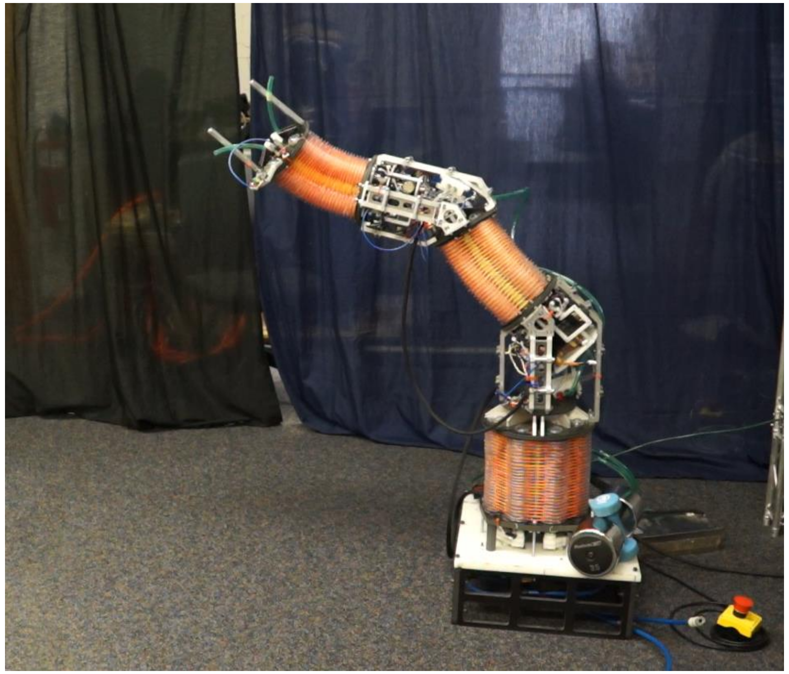
2. Adapting unknown terms in dynamics.

3. Fully learned models.

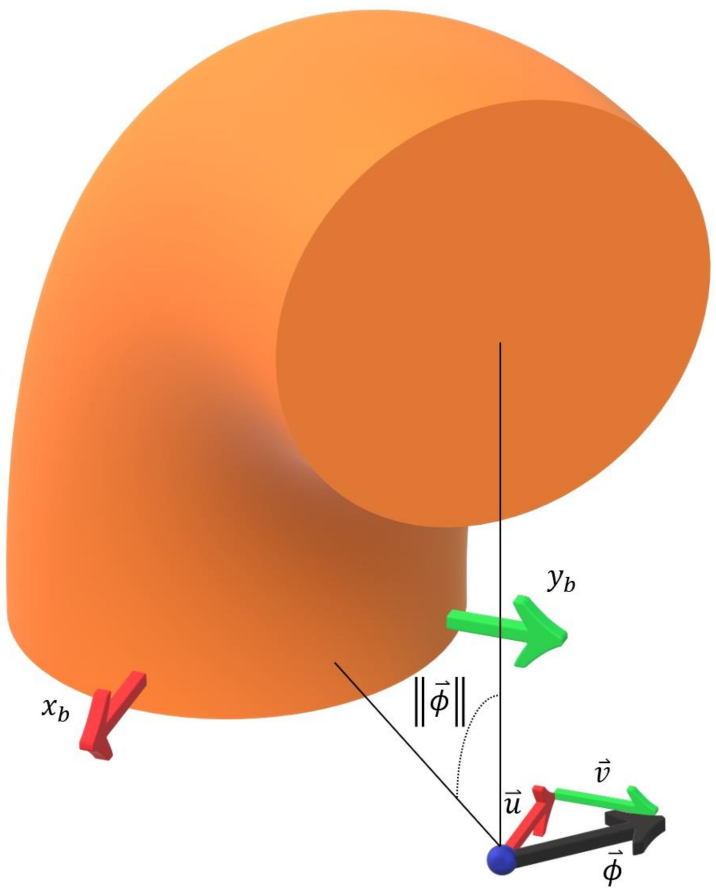
First Principles Modeling for Control

First Principles Models



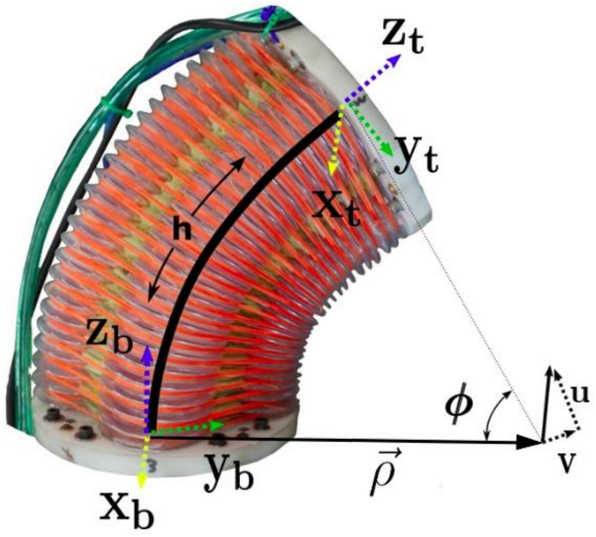


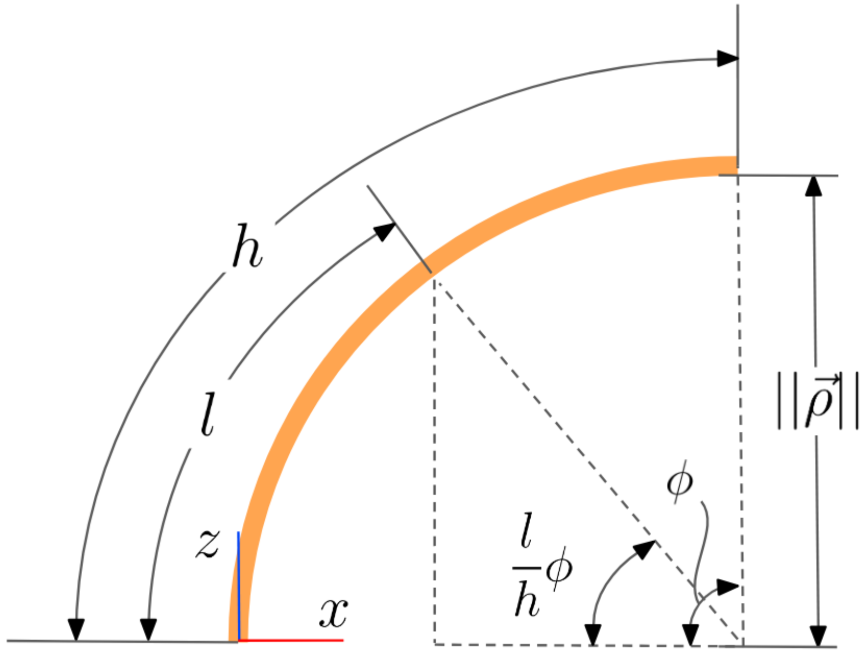
Kinematic Modeling



Allen, T. F., Rupert, L., Duggan, T. R., Hein, G., & Albert, K. (2020, May). Closed-Form Non-Singular Constant-Curvature Continuum Manipulator Kinematics. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)* (pp. 410-416). IEEE.

Kinematic Modeling





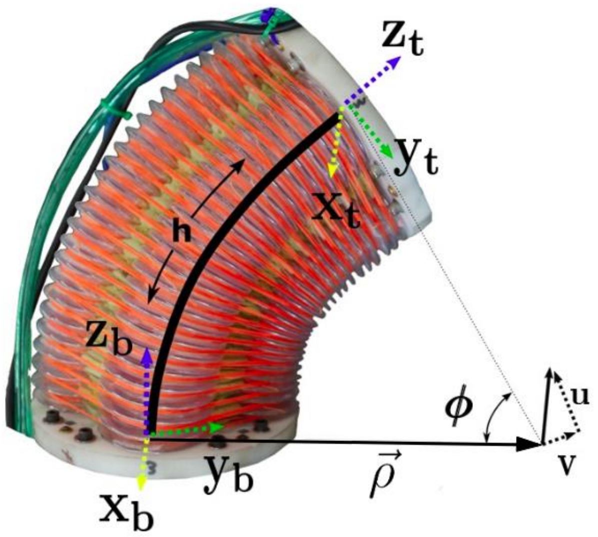
$$\begin{bmatrix} v_l \\ u_l \\ 0 \end{bmatrix} = \frac{l}{h} \begin{bmatrix} v_h \\ u_h \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{v}_l \\ \dot{u}_l \\ 0 \end{bmatrix} = \frac{l}{h} \begin{bmatrix} \dot{v}_h \\ \dot{u}_h \\ 0 \end{bmatrix}$$

Assumptions:

- Piecewise constant curvature

Pneumatically Actuated Robot Dynamics (using Euler-Lagrange Dynamics)



$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = Q$$

$$q = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_h \\ v_h \end{bmatrix}$$

$$\dot{q} = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \dot{u}_h \\ \dot{v}_h \end{bmatrix}$$

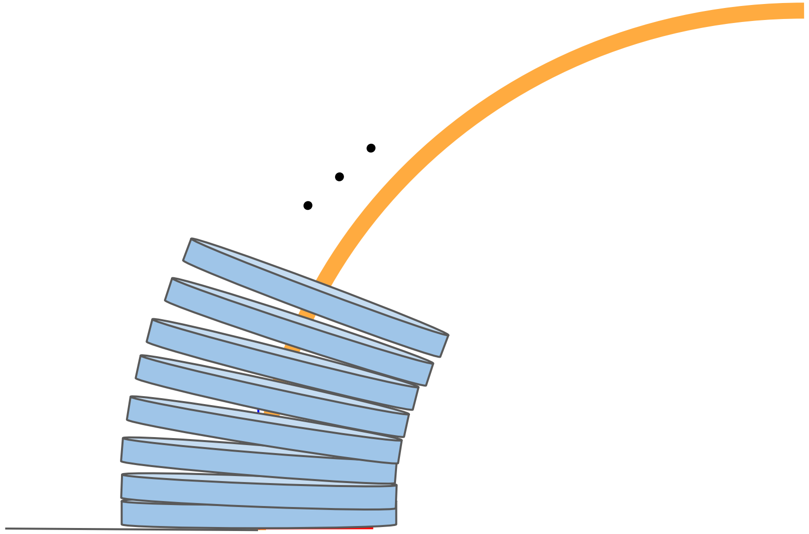
$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + g(q) = \tau$$

$$L = T - V$$

Where:

- L is called the Lagrangian
- T is kinetic energy
- V is potential energy

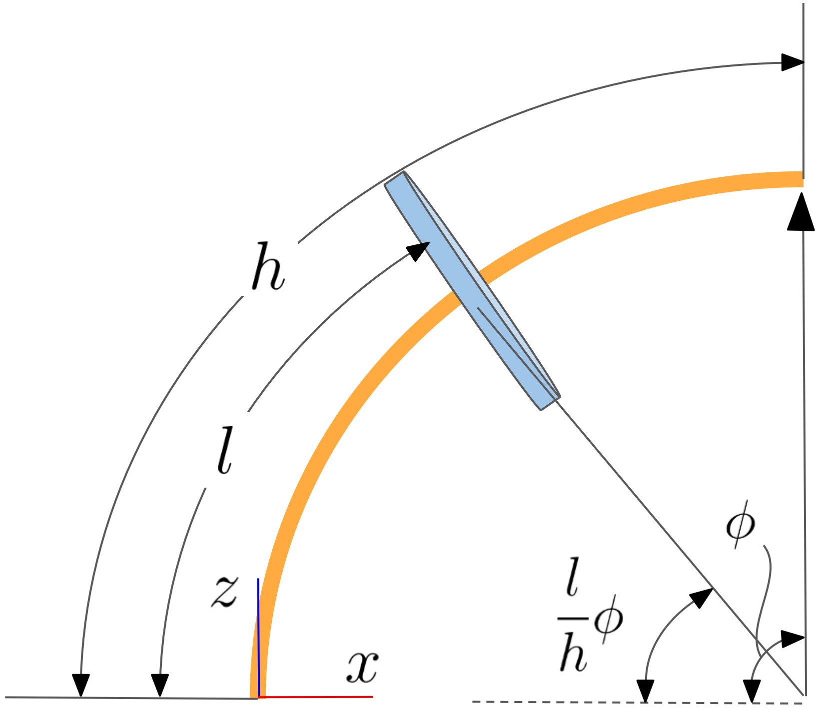
Model Mass as a Series of Disks



Let each disk represent a section of the continuum joint with some mass and rotational inertia.

Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots.
(with a journal paper under review)

Kinematics (location and velocity) of each Disk



$$\begin{bmatrix} v_l \\ u_l \\ 0 \end{bmatrix} = \frac{l}{h} \begin{bmatrix} v_h \\ u_h \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{v}_l \\ \dot{u}_l \\ 0 \end{bmatrix} = \frac{l}{h} \begin{bmatrix} \dot{v}_h \\ \dot{u}_h \\ 0 \end{bmatrix}$$

Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots.
(with a journal paper under review)

Soft Robot Modeling and Estimation

$$J_{weighted} = \begin{bmatrix} \sqrt{\mu} J_{\dot{p}_{l,x}} \\ \sqrt{\mu} J_{\dot{p}_{l,y}} \\ \sqrt{\mu} J_{\dot{p}_{l,z}} \\ \frac{\sqrt{\mu} r}{2} J_{\omega_{l,x}} \\ \frac{\sqrt{\mu} r}{2} J_{\omega_{l,y}} \\ \frac{\sqrt{\mu} r}{\sqrt{2}} J_{\omega_{l,z}} \end{bmatrix}$$

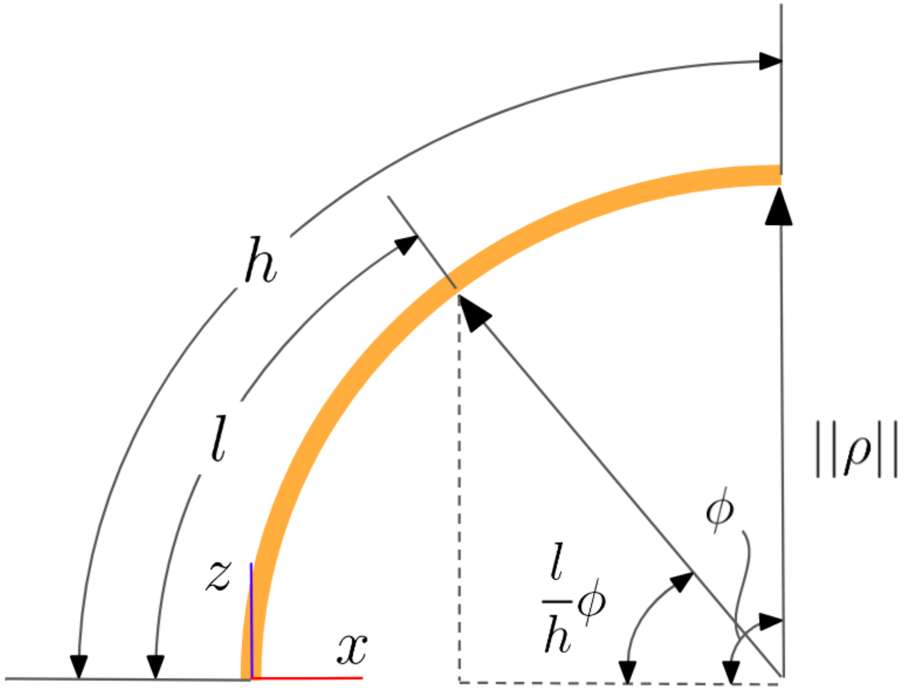
Now T (kinetic energy) for a single disk can be written as a function of time derivative of the generalized coordinates:

$$T_l = \frac{1}{2} \dot{q}^T J_{weighted}(u_l, v_l, l)^T J_{weighted}(u_l, v_l, l) \dot{q} \, dl$$

$$\begin{bmatrix} \dot{v}_l \\ \dot{u}_l \\ 0 \end{bmatrix} = \frac{l}{h} \begin{bmatrix} \dot{v}_h \\ \dot{u}_h \\ 0 \end{bmatrix}$$

Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots.
(with a journal paper under review)

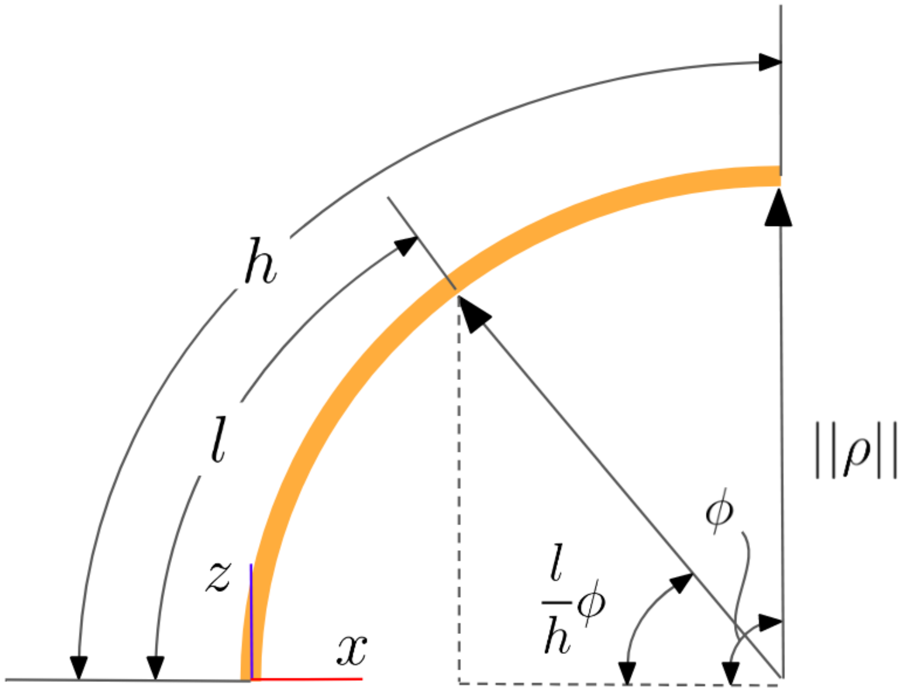
Soft Robot Modeling and Estimation



$$T = \frac{1}{2}\dot{q}^T \left[\int_0^h J_{weighted}(u_l, v_l, l)^T J_{weighted}(u_l, v_l, l)\, dl \right] \dot{q}$$

Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots.
(with a journal paper under review)

Soft Robot Modeling and Estimation



For potential energy P, we just need the center of mass of each continuum segment as a function of our generalized coordinates:

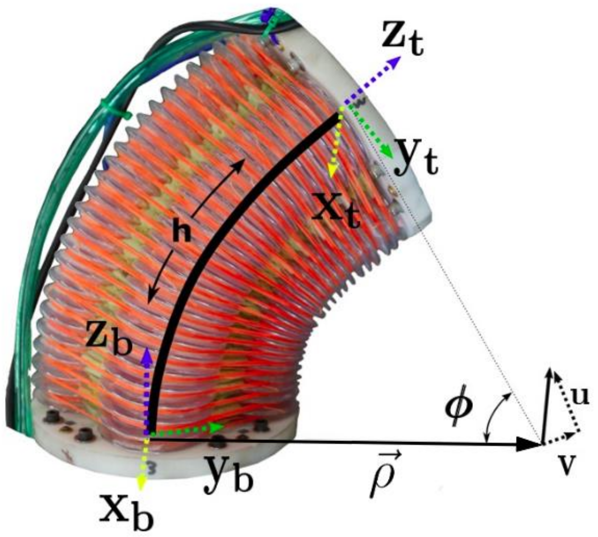
$$\vec{p} = \frac{h}{\phi^2}\begin{bmatrix}(\phi - \sin(\phi))\frac{v}{\phi} \\ (\phi - \sin(\phi))\frac{-u}{\phi} \\ (1 - \cos(\phi))\end{bmatrix}$$

If “G” is the gravity vector expressed in the same frame as “p”, then potential energy “V” can be written as:

$$V = \vec{p} \cdot \vec{G}$$

Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots.
(with a journal paper under review)

Pneumatically Actuated Robot Dynamics (using Euler-Lagrange Dynamics)



$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = Q$$

$$q = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_h \\ v_h \end{bmatrix}$$

$$\dot{q} = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \dot{u}_h \\ \dot{v}_h \end{bmatrix}$$

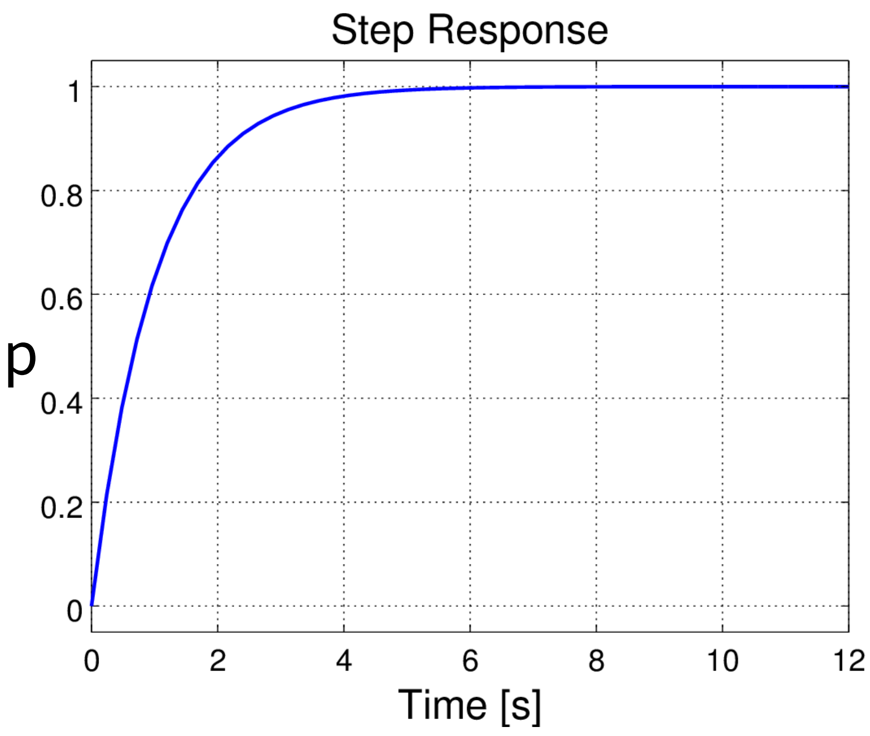
$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + g(q) = \tau$$

$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + g(q) = K_{pressure}\, p + K_{spring}\, q + K_{damper}\, \dot{q}$$

Where:
- M is the mass matrix in the generalized coordinate space
- C is the Coriolis and centripetal terms
- g is the torque due to gravity
- Tau is the friction AND parasitic torque AND actuation torque from pressure

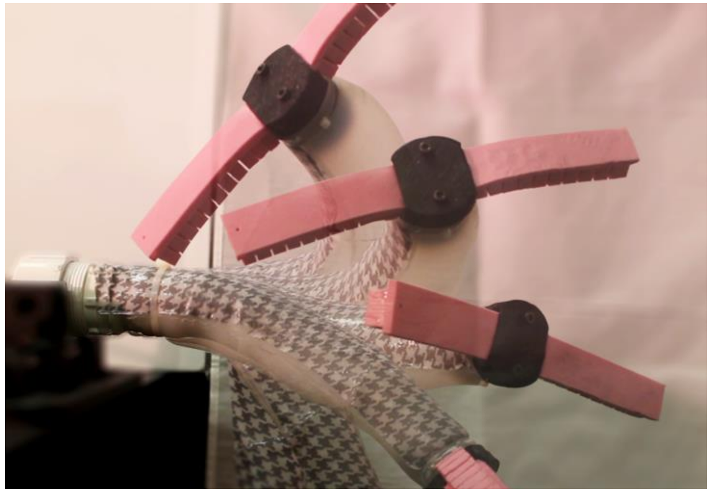
Pressure Dynamics

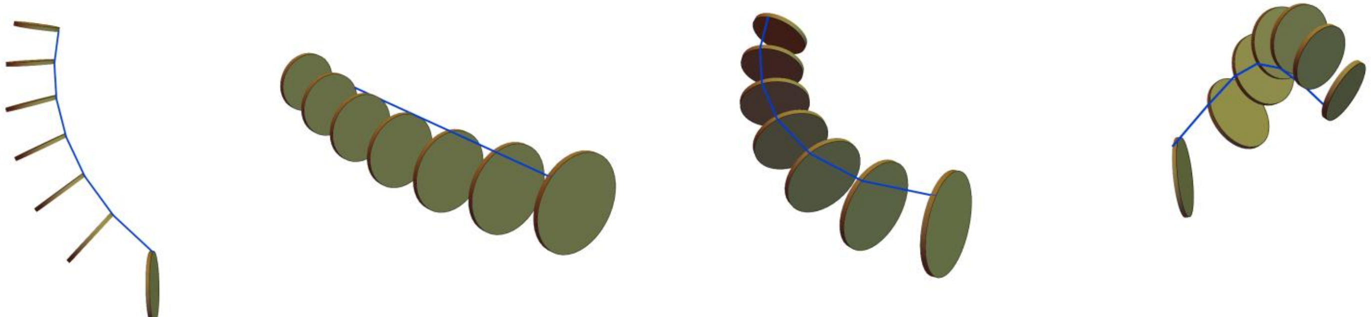
$$\dot{p} = \alpha(p_{ref} - p)$$

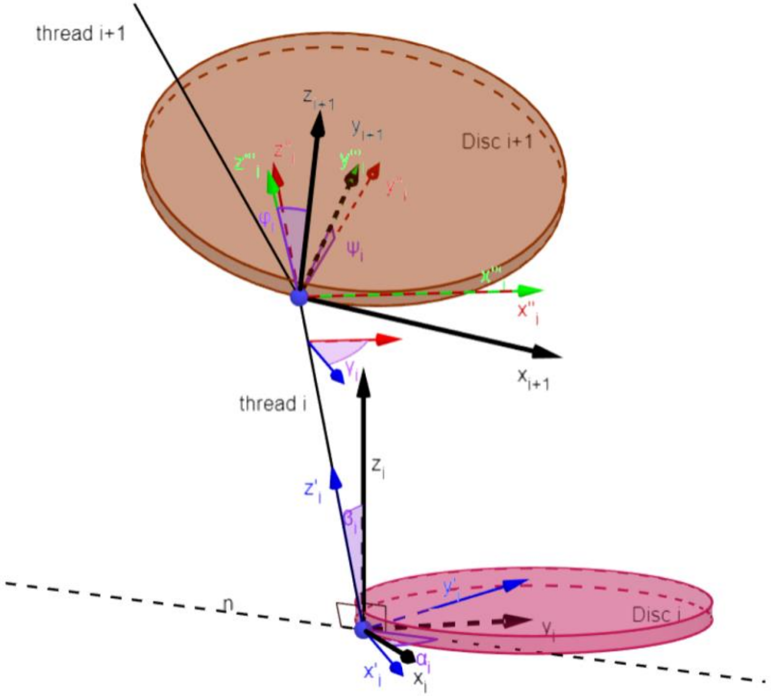


Thread-Disk Model Preview

- In collaboration with Dr. Joshua Schultz at the University of Tulsa
- Although lumped models, extra degrees of freedom allow it to bend, buckle, and twist.





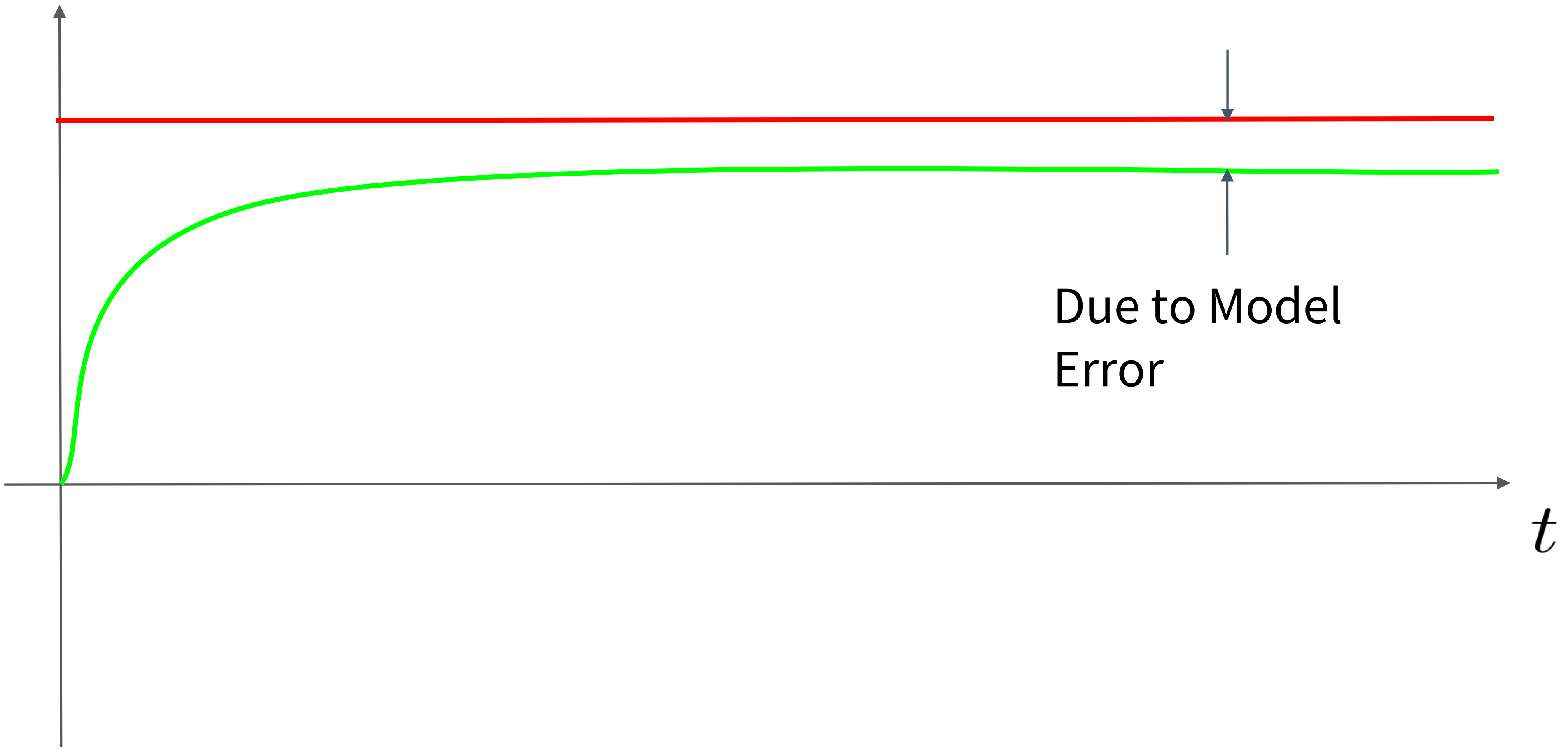


First Principles Modeling for Control

- Pros –
  - Intuition behind model and its meaning for control
  - Can easily add terms (like friction or parasitic torque)
  - Can look at dominant terms in model
  - Clear methods for system ID
  - Likely makes proving stability much easier
- Cons -
  - No guarantee that your model is accurate enough for control
  - Can take lots of work to develop a completely new model (disk-thread)
    - And may still not represent the system well
  - Still requires collecting data to do system ID (although not as much as ML)
  - May not be tractable for control

Adapting Unknown Dynamics

Model-based Control Still Gives Steady State Error



Model Reference Adaptive Control (MRAC)

Let control be the following:

$$\tau = M\ddot{q}_{ref} + C\dot{q} + g$$

$$M\ddot{q} + C\dot{q} + g = \tau$$

$$\ddot{q} = M^{-1}\left[\tau - C\dot{q} - g\right]$$

$$\ddot{q} = M^{-1}\left[\left(M\ddot{q}_{ref} + C\dot{q} + g\right) - C\dot{q} - g\right]$$

$$\ddot{q} = \cancel{M^{-1}(M}\ddot{q}_{ref}) + M^{-1}\cancel{(C\dot{q} - C\dot{q})} + M^{-1}\cancel{(g - g)}$$

$$\ddot{q} = \ddot{q}_{ref}$$

Model Reference Adaptive Control (MRAC)

$$\tau = M\ddot{q}_{ref} + C\dot{q} + g$$
$$\tau = Y(q, \dot{q}, \dot{q}_{ref}, \ddot{q}_{ref})a$$

- We can rewrite our dynamics model in terms of a regressor matrix, and an unknown set of parameters in the vector "a"
- Only the regressor (Y) is needed
  - Usually starts with form from first principles model, but can add whatever terms we think are relevant.
- Provably stable convergence to the reference trajectory
- Given enough control authority, you can make your system behave like whatever "reference" system you chose

MRAC – what's the catch?

What about terms that don't show up in the regressor?
(because we don't know how to express them?)

$$M\ddot{q} + C\dot{q} + g + ??? = \tau$$

$$\downarrow$$

$$\ddot{q} = \ddot{q}_{ref} + M^{-1}???$$

MRAC + MPC = MRPAC

Can we combine the robustness of model-based MPC, with the adaptation of unknown terms from MRAC?

$$\dot{x} = Ax + Bu + w$$

$$\Downarrow$$

$$\dot{x} = Ax + Bu + (w + Ya)$$

Disturbances to linear model in general when doing MPC alone.

Disturbances that include a dynamics term based on our regressor AND the adaptive control term "a" for MRAC+MPC

Comparison of Control with Three Different Methods

- Model predictive control (MPC, model-based approximation for optimal control)
- Model reference adaptive control (MRAC)
- Model reference adaptive control + MPC (MRPAC)

Comparisons:

1. with a parameter mismatch (in inertia) – sim

2. with a model mismatch (in spring return equilibrium position) – sim

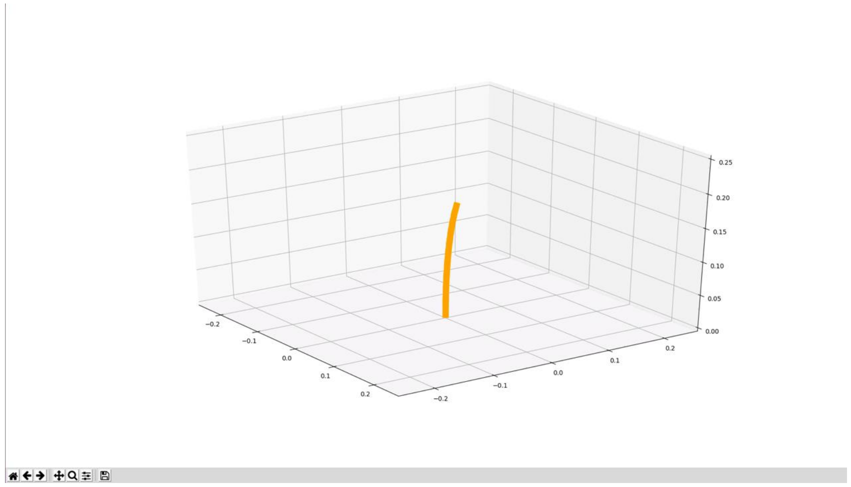
3. on real hardware where there is both parameter and model mismatch

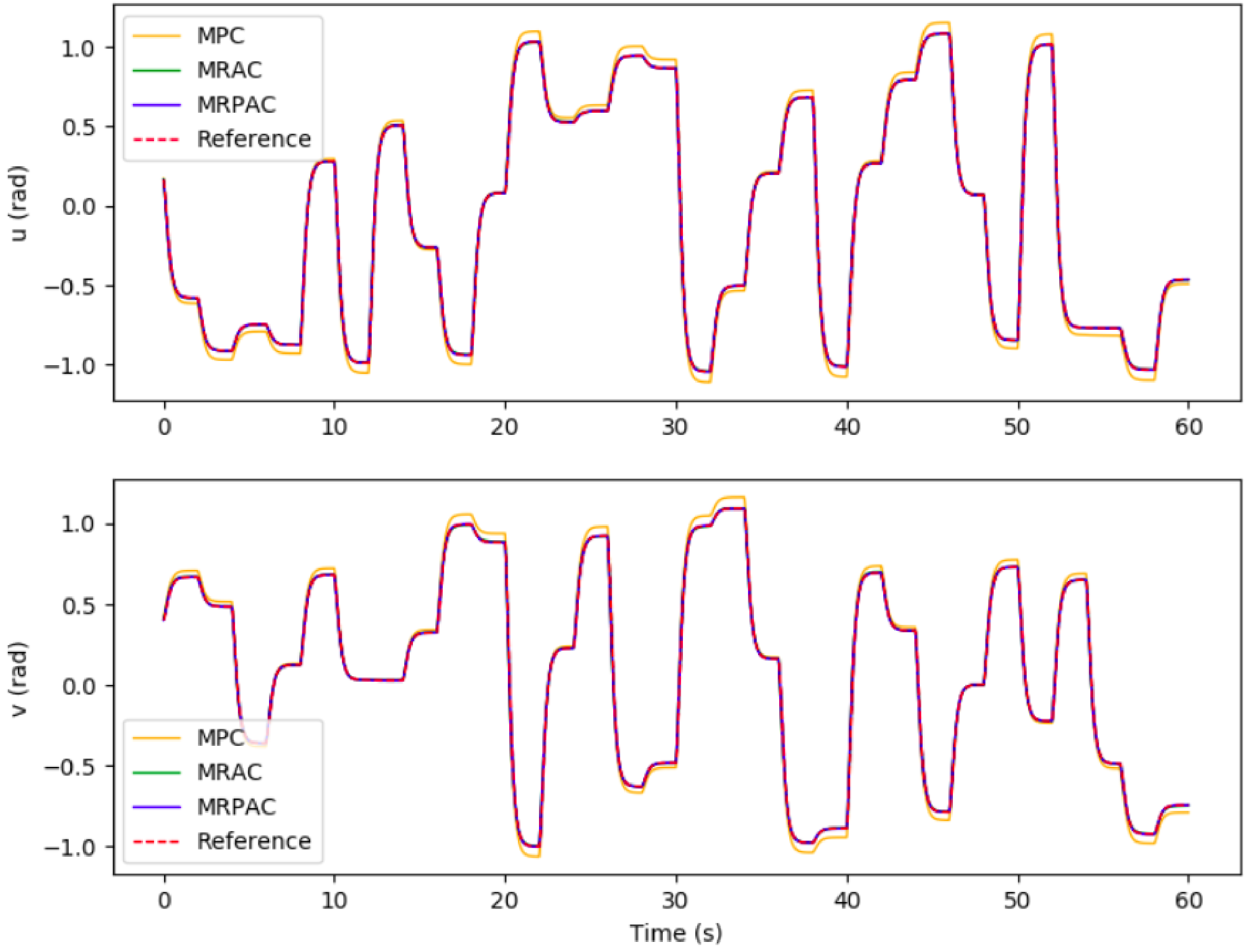
Effect of Model Mismatch

(significant error in inertia)

Model parameter error

(e.g incorrect estimates of length, mass, stiffness, and/or inertia)

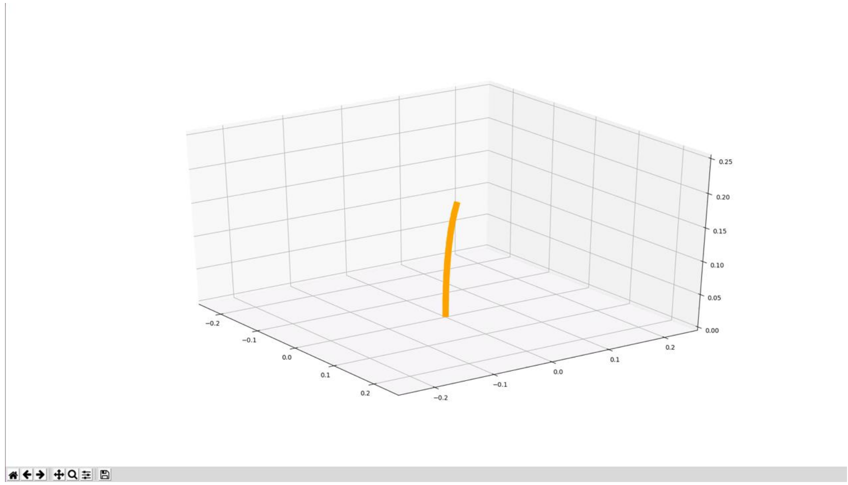


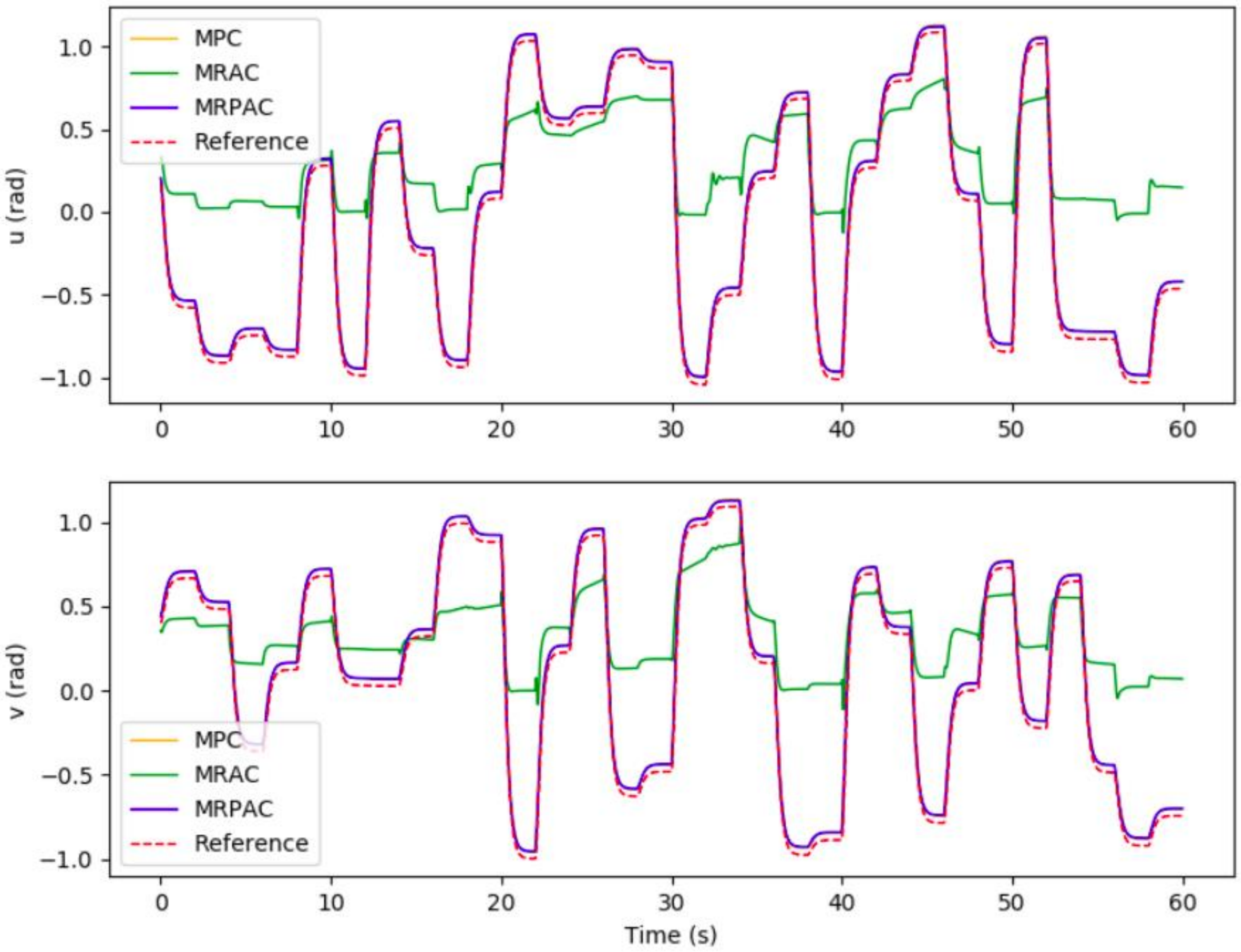


Effect of Structural Mismatch
(error in equlibrium config)

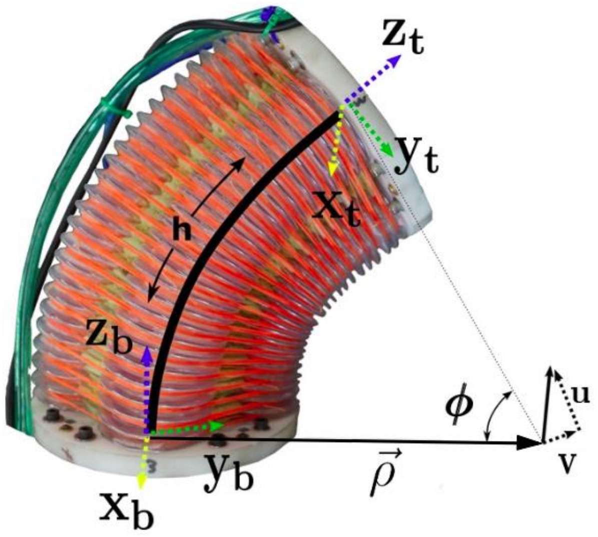
Offset Force/Torque

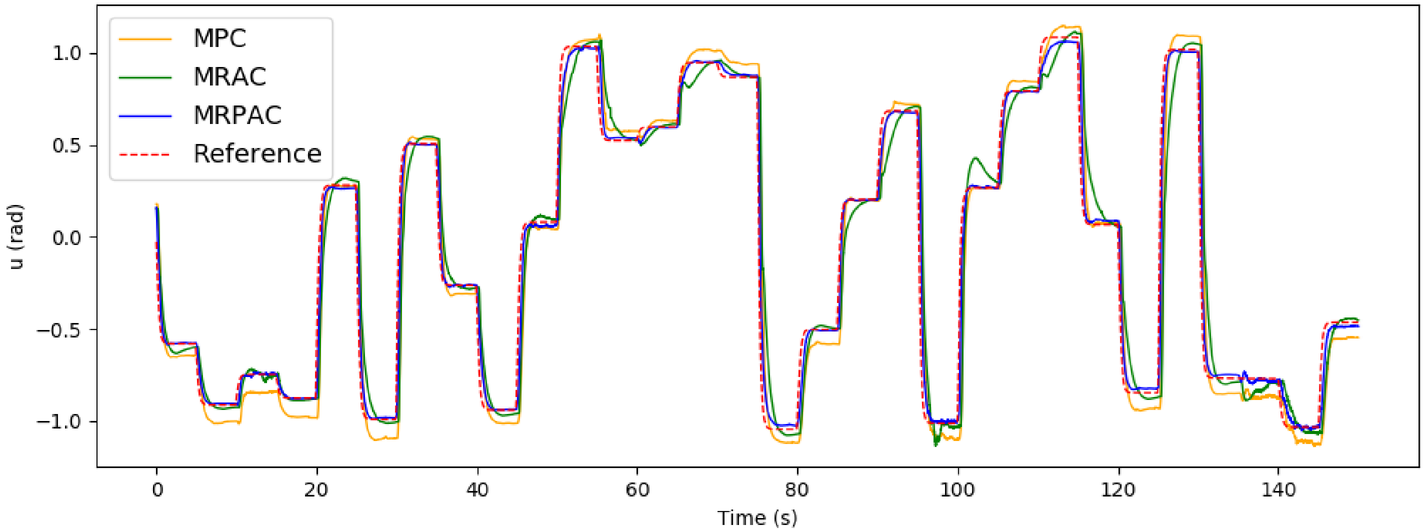
(e.g. wrong spring equilibrium value)

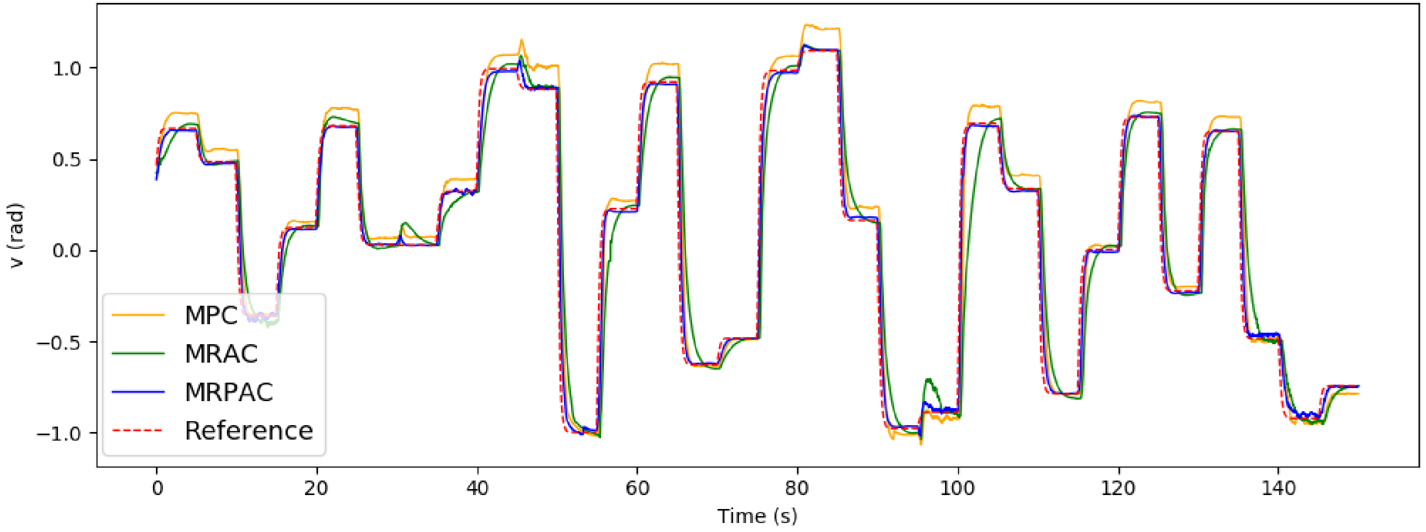




Hardware Results







Dissertation - Hyatt, P. E. (2020). Robust Real-Time Model Predictive Control for High Degree of Freedom Soft Robots. (with a journal paper under review)

Terry, J. S., Whitaker, J., Beard, R. W., & Killpack, M. D. (2019, October). Adaptive Control of Large-Scale Soft Robot Manipulators With Unknown Payloads. In *ASME 2019 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection.

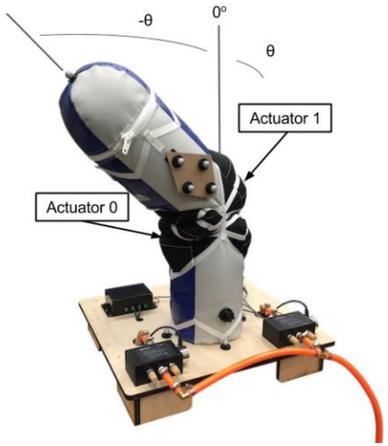
Online Adaptive Modeling for Control

- Many of the same pros and cons of modeling based on first principles …
- Pros -
  - Don't need to accurately know all terms in model (we'll adapt them over time)
  - Can adapt parameters based on tracking error (MRAC) instead of model error alone

- Cons -
  - Still have to identify or guess the form of the regressor terms
  - Adaptive control alone is not robust to unmodeled terms (like hysteresis, friction, choked/unchoked flow in pneumatics)

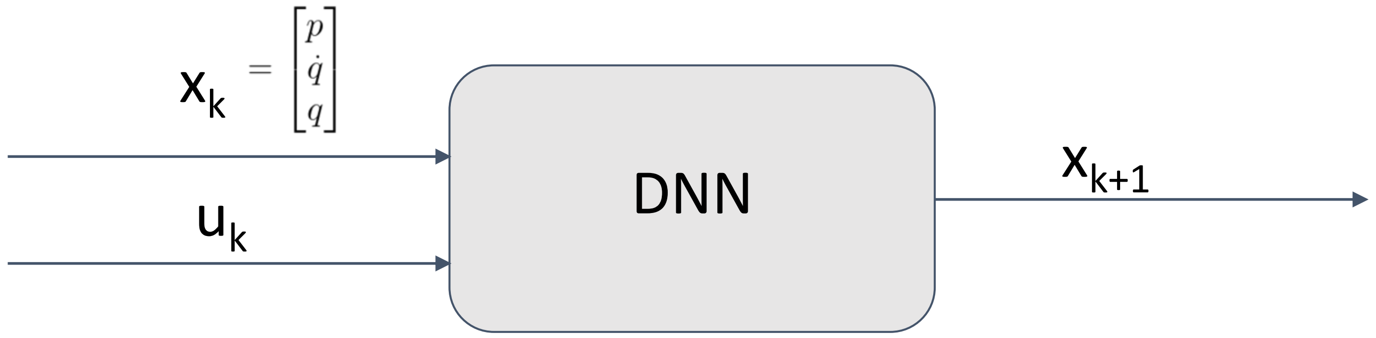
Fully Learned
Models

Current State-of-the-Art for Large Scale Soft Robot Manipulation



- Results involve years of work using first principle models, hand-tuned controllers
- THAT’S NO GOOD if …
  - Every soft robot platform is different from another
  - Every platform still requires significant system identification

Learned Discrete-Time Models



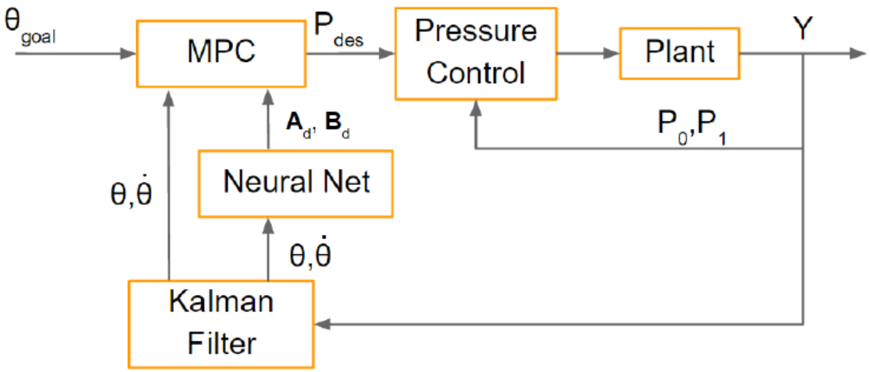
Learned DNN models for control

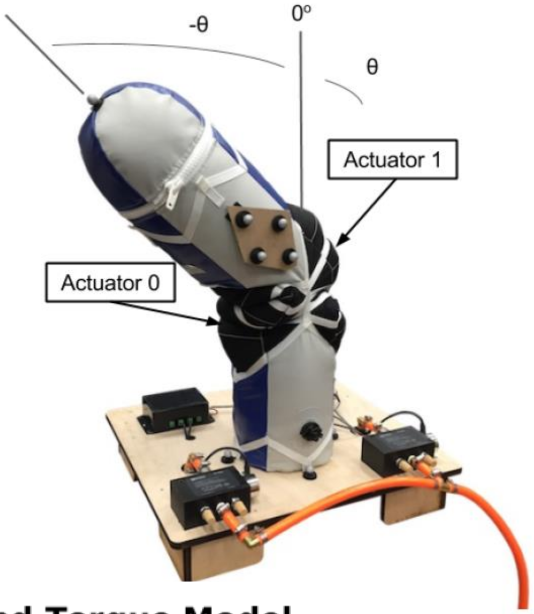
$$\mathbf{x}[k+1] = f\left(\mathbf{x}[k], \mathbf{u}[k]\right)$$

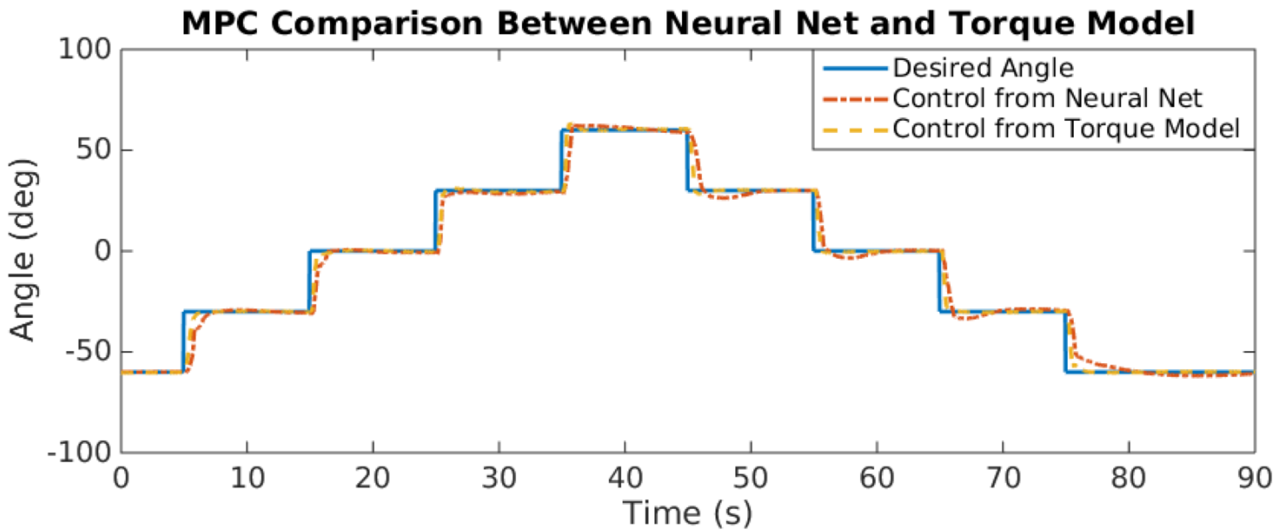
$$\mathbf{A}_d = \begin{bmatrix} \frac{\partial f_1}{\partial \dot{\theta}} & \frac{\partial f_1}{\partial \theta} & \frac{\partial f_1}{\partial P_0} & \frac{\partial f_1}{\partial P_1} \\ \frac{\partial f_2}{\partial \dot{\theta}} & \frac{\partial f_2}{\partial \theta} & \frac{\partial f_2}{\partial P_0} & \frac{\partial f_2}{\partial P_1} \\ \frac{\partial f_3}{\partial \dot{\theta}} & \frac{\partial f_3}{\partial \theta} & \frac{\partial f_3}{\partial P_0} & \frac{\partial f_3}{\partial P_1} \\ \frac{\partial f_4}{\partial \dot{\theta}} & \frac{\partial f_4}{\partial \theta} & \frac{\partial f_4}{\partial P_0} & \frac{\partial f_4}{\partial P_1} \end{bmatrix}$$

$$\mathbf{B}_d = \begin{bmatrix} \frac{\partial f_1}{\partial P_{0,des}} & \frac{\partial f_1}{\partial P_{1,des}} \\ \frac{\partial f_2}{\partial P_{0,des}} & \frac{\partial f_2}{\partial P_{1,des}} \\ \frac{\partial f_3}{\partial P_{0,des}} & \frac{\partial f_3}{\partial P_{1,des}} \\ \frac{\partial f_4}{\partial P_{0,des}} & \frac{\partial f_4}{\partial P_{1,des}} \end{bmatrix}$$

$$\mathbf{x}[k+1] = \mathbf{A}_d\mathbf{x}[k] + \mathbf{B}_d\mathbf{u}[k]$$







Gillespie, M. T., Best, C. M., Townsend, E. C., Wingate, D., & Killpack, M. D. (2018, April). Learning nonlinear dynamic models of soft robots for model predictive control with neural networks. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)* (pp. 39-45). IEEE.

Extension of Learned DNN models for Multi-DoF Control

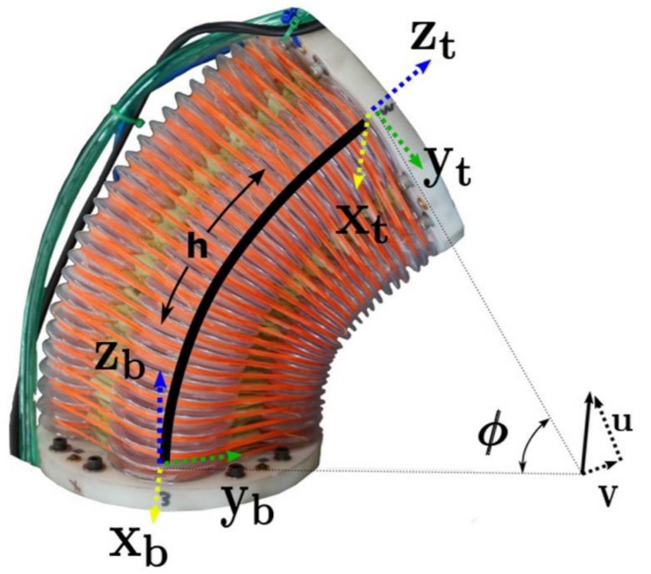
- Use DNN to approximate – $\dot{q}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k)$.
- Then dynamic model can be written as follows:

$$\mathbf{x_{k+1}} = \mathbf{A}_d(\mathbf{x}_k - \mathbf{x}_k) + \mathbf{B}_d(\mathbf{u}_k - \mathbf{u}_k) + \mathbf{w}_d$$

$$\mathbf{A}_d = \begin{bmatrix} (I - \alpha\Delta t) & 0 & 0 \\ \frac{\partial f}{\partial p_k} & \frac{\partial f}{\partial \dot{q}_k} & \frac{\partial f}{\partial q_k} \\ \frac{\partial f}{\partial p_k}\frac{\Delta t}{2} & (\frac{\partial f}{\partial \dot{q}_k} + I)\frac{\Delta t}{2} & \frac{\partial f}{\partial q_k}\frac{\Delta t}{2} + I \end{bmatrix}$$

$$\mathbf{B}_d = \begin{bmatrix} \alpha\Delta t \\ \frac{\partial f}{\partial p_{ref,k}} \\ 0 \end{bmatrix}$$

$$\mathbf{w}_d = \begin{bmatrix} p_0 \\ f(\mathbf{x}_0, \mathbf{u}_0) \\ \frac{\Delta t}{2}\dot{q}_0 + q_0 \end{bmatrix}$$



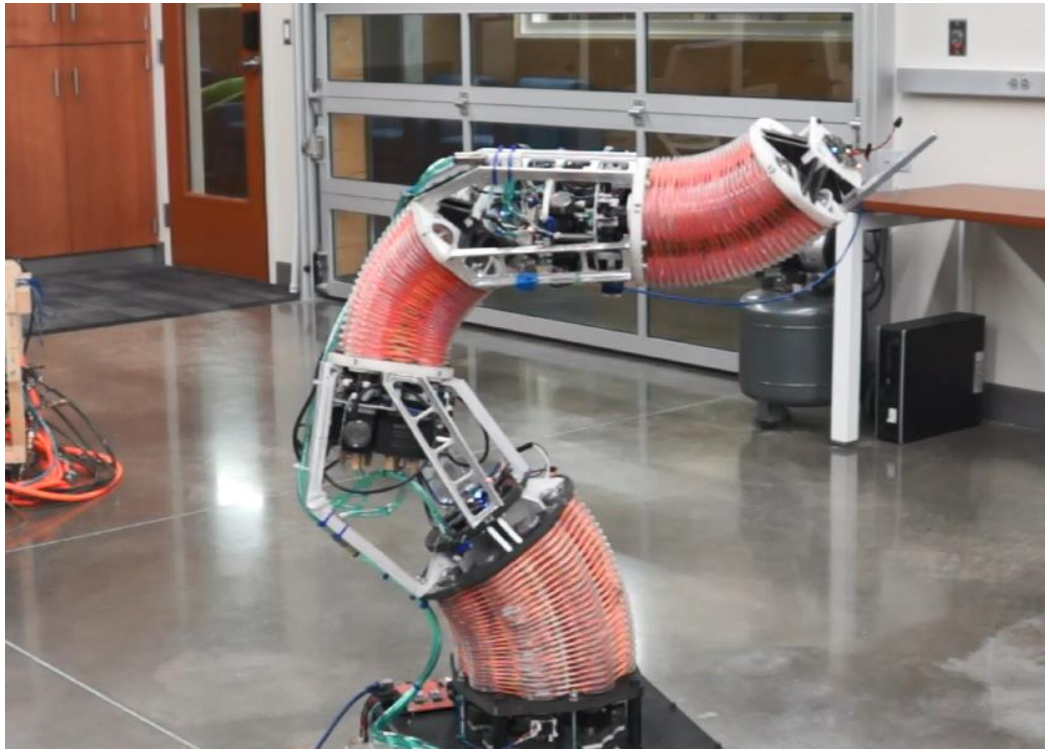
Hyatt, P., Wingate, D., & Killpack, M. D. (2019). Model-based Control of Soft Actuators Using Learned Nonlinear Discrete-time Models. *Frontiers in Robotics and AI, 6, 22.*

Limitations

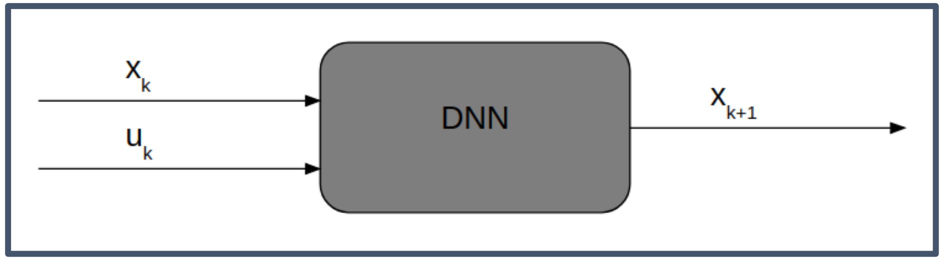
1. Using convex solver, still slow if want more joints

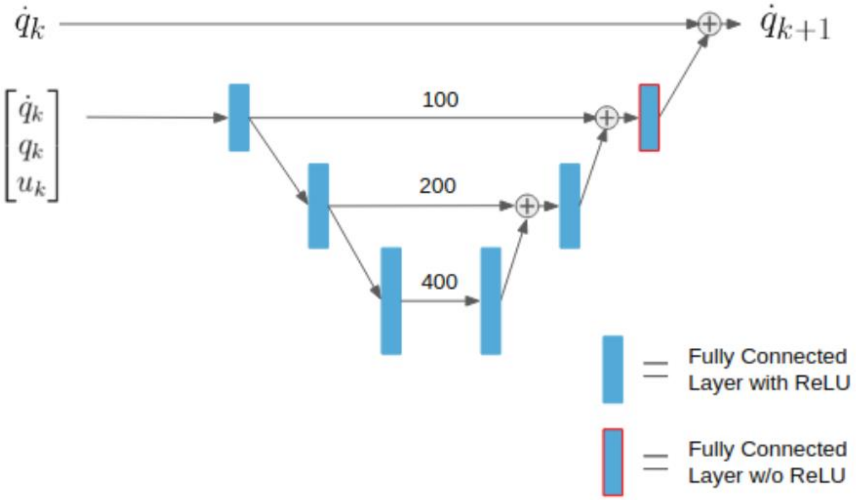
2. Not really making use of fast GPU evaluation

3. Cannot represent nonlinear effects over a long time horizon

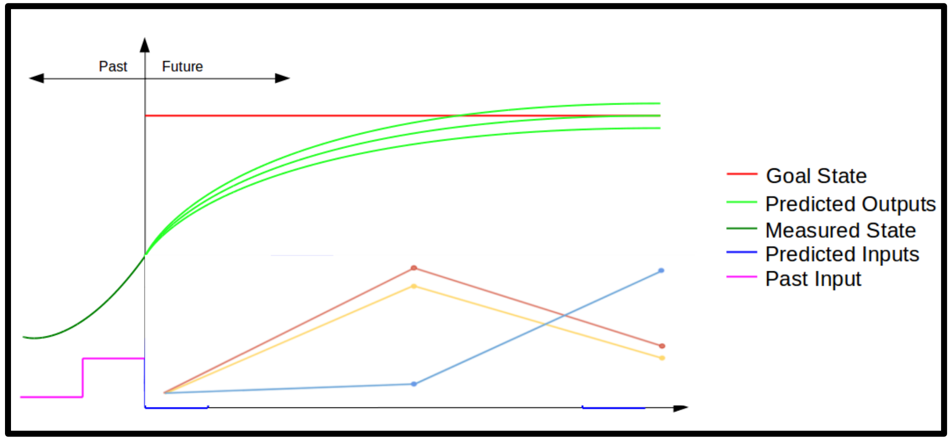


Linear/Nonlinear EMPC





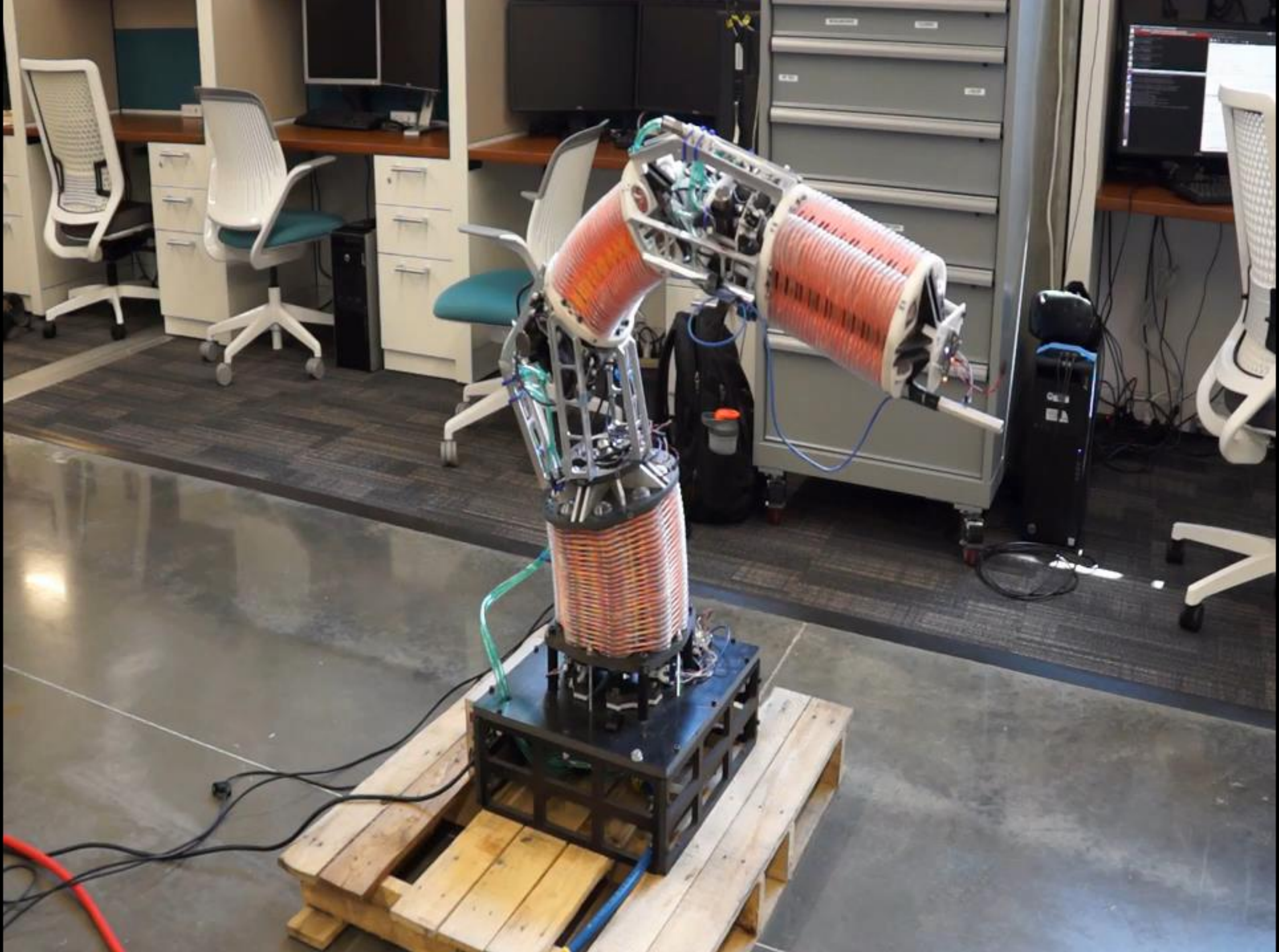
+



Hyatt, P., & Killpack, M. D. (2019). Real-Time Nonlinear Model Predictive Control Using a Graphics Processing Unit. *IEEE Robotics and Automation Letters 2020.*

Hyatt, P., & Killpack, M. D. (2017, November). Real-time evolutionary model predictive control using a graphics processing unit. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (pp. 569-576). IEEE.

Hyatt, P., Williams, C. S., & Killpack, M. D. (2020). Parameterized and GPU-Parallelized Real-Time Model Predictive Control for High Degree of Freedom Robots. *arXiv preprint arXiv:2001.04931*.



Learned Models for Control

- Pros
  - Just collect data and hit "go“
  - Can use GPUs for fast evaluation (and solution for MPC)
  - Can easily represent nonlinear models

- Cons
  - Questions about scalability for realtime evaluation vs accuracy
  - Picking the right DNN architecture matters a lot
  - Collecting "useful" data to get good models for control is difficult
  - Lack of intuition (explainable AI could help)
  - No guarantees about stability or even generality of model

Acknowledgements

Work also funded under a SBIR Phase II grant in collaboration with Pneubotics at Otherlab

Work funded by a NASA Space Technology Research Grant under the Early Career Faculty program

Work funded by a NSF EFRI program


Thank You