Simulation for a Two Degree of Freedom Robot Arm Actuated by Pneumatic Muscle Groups

R. Nicholas Vandemark
University of Maryland
ENPM640 - Rehabilitation Robotics, Project
nickvand@umd.edu

***Abstract*—Research on sliding-mode control techniques for a two degree of freedom planar robot arm actuated by pneumatic muscles is observed and discussed. Mathematical derivations for the robot model and control are described. A real-time capable simulation suite was developed and implementation details are discussed.**

***Index Terms*—Pneumatic muscle actuators, rehabilitative robotics, ROS2 simulation**

I. Introduction

A pneumatic muscle (PM) is an artifical bladder device whose expansion/contraction can be controlled via changes in pressure. It can induce actuation with many advantageous properties such as its light-weight and low cost materials, as well as its subsequent high power-to-weight ratio [1]. Actuators that are comprised of PM's can be aptly referred to as a pneumatic muscle actuator (PMA), and can be seen in antagonistic pairs when representing a system similar to a simple human joint. For example, a bicep and tricep muscle pair, which would represent two antagonist PM's, can act in opposing contraction and relaxation to actuate the corresponding elbow joint. Each "muscle" described here is more appropriately considered a "muscle group", where each group may also actually be comprised of many PM's, but together this muscle group creates a representation of a single e.g. bicep or tricep muscle. While in general it is not required, for the scope of this paper, each antagonistic muscle group (e.g. a bicep and tricep of a single PMA segment) consist of the same number of PM's.

Robotic systems which utilize PMA's can be implemented with numerous advantages in doing so. The benefits already mentioned would apply to each PMA utilized in the system, but additionally, the structure of these PM segments are likely more comfortable when interfacing with humans, such as in a rehabilitative exoskeleton [1]. Standard mechatronic robotic segments would likely be heavier, harder, larger, and perhaps colder to the touch. All of these downsides introduce a rehabilitative device that could be less approachable for the user.

The authors of [1] discuss the history of robotic systems comprised of PMA's, as well as addresses control for a two degree of freedom (DoF) planar robot arm. The hypothetical system consists of two robotic segments as described above, where each joint is actuated by antagnostic pairs of PM's, referred to there as a bicep and tricep muscle group. The reasoning for considering a sliding-mode control approach, above all other possible options, is also explained.

This paper descibes an attempt at implementing a simulation for this hypothetical robotic system using *ROS2* and *MoveIt 2* according to the dynamic model described in [1], with fully dynamically configurable simulation parameters. Two cantral components were designed using the *ros2_control* framework, a custom controller which implements the sliding-mode control law, and a custom hardware interface which simulates reading/writing data from/to the robotic system.

II. Derivations and Calculations

As previously mentioned, a single segment for a robotic PMA consists of two PM groups, an example of which can be seen in Figure II, where the b and t subscripts denote a bicep and tricep muscle, respectively.

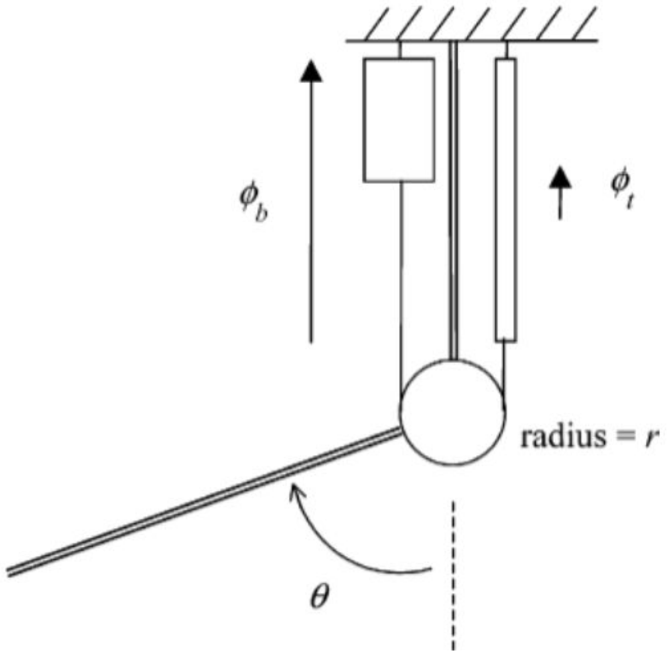

Fig. 1. Two PM's acting antagonistically to create a single robotic PMA segment, Figure 2 of [1].

The force exerted by a PM due to pressure change (i.e. expansion/contraction) is described by Eqn. (1):

$$\begin{aligned}\phi_{PM}(P,\theta,\dot{\theta}) = F_{PM}(P) - K_{PM}(P)\cdot x_{PM}(\theta)\\ - B_{PM}(P)\cdot \dot{x}_{PM}(\dot{\theta})\end{aligned} \tag{1}$$

where the force exerted by the PM is a function of its current pressure and the position and velocity of the joint. The exact means to calculate the PM length and lengthening rate ($x_{PM}(\theta)$ and $\dot{x}_{PM}(\dot{\theta})$, respectively) depend on the structure of the PM segment, but is generally a function of the joint position and velocity. The nonlinear $F(P)$, $K(P)$, and $B(P)$ coefficients have expanded forms:

$$F_{PM}(P) = F_{PM_0} + (F_{PM_1}\cdot P) \tag{2}$$

$$K_{PM}(P) = K_{PM_0} + (K_{PM_1}\cdot P) \tag{3}$$

$$B_{PM}(P) = \begin{cases} B_{PM_{0_i}} + (B_{PM_{1_i}}\cdot P) & \text{if PM is inflating}\\ B_{PM_{0_d}} + (B_{PM_{1_d}}\cdot P) & \text{else}\end{cases} \tag{4}$$

The torque acting on the PM segment due to this force is described by Eqn. (5):

$$\tau_{PM}(P,\theta,\dot{\theta}) = r_{PM}\cdot\phi_{PM}(P,\theta,\dot{\theta}) \tag{5}$$

where r_{PM} is the radius of the pulley used in the PM segment.

The following is a more detailed and generalized derivation of Eqns. 3.6-3.7 from [1]. The generalization is done so because it is how the simulation's code was implemented, and because it helps to showcase a possible reoccurring typo or unexplained simplification in this article. Consider what Lilly and Quesada stated (2004, p.351):

> If several PM's are present in a system, each one generally has its own F, K, and B coefficients, its own input pressure, and its own inflation or deflation status.

which explicitly qualifies each PM for its own unique characterization. The authors denote this with b and t subscripts in their Eqns. 2.7-2.8, but they are dropped for the F and K coefficients (but kept for the B coefficient) starting with their Eqn. 3.1, but it does not seem to be explained as to why. Furthermore, according to their Eqns. 3.7a and 3.7c, if this were the case, then neither the values for F_{0s} nor K_{0s} would actually matter, as both of those terms are added and then subtracted out.

For these reasons, it is believed to be best to generalize the equations to unique sets of coefficients for each PM group (the shoulder bicep, shoulder tricep, elbow bicep, and elbow tricep).

An input pressure for a PM can be described as the sum of some positive nominal constant pressure and some offset from it, according to Eqn. (6):

$$P = P_0 + \Delta p \tag{6}$$

See Section IV of [1] for a discussion on how this pressure for some PM can be selected, such that the PMA is well behaved in the absence of a control signal Δp_{PM} and a desired joint stiffness is produced.

The total torque acting on a PMA pulley depends on how it is described. Figure II shows that a torque due to the shoulder's tricep muscle is a positive contribution (and therefore the bicep muscle has a negative contribution), but that the elbow's tricep msucle is the negative contribution (and the bicep muscle is therefore the positive contribution). Furthermore, as mentioned previously, one e.g. 'bicep muscle' can also be respresented as a muscle group that consists of individual, identitical PM's. Considering all of this, combined with Eqn. (5) produces Eqns. (7) and (8), the torques acting on the shoulder and elbow PMA's, respectively:

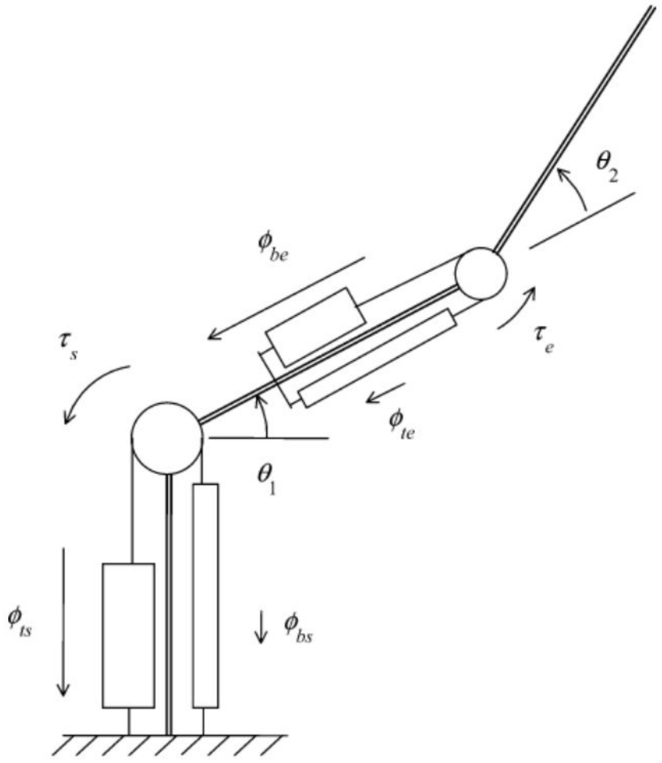

Fig. 2. Two DoF planar robot arm with two PMA's, Figure 3 of [1].

$$\tau_s(P_b,P_t) = n_s\cdot r_s\cdot[\phi_{s_t}(P_t) - \phi_{s_b}(P_b)] \tag{7}$$

$$\tau_e(P_b,P_t) = n_e\cdot r_e\cdot[\phi_{e_b}(P_b) - \phi_{e_t}(P_t)] \tag{8}$$

where the forces due to PM's as part of the shoulder segment, $\phi_{s_b}(P)$ and $\phi_{s_t}(P)$, are a function of the first joint's position and velocity, and the forces due to PM's as part of the elbow segment, $\phi_{e_b}(P)$ and $\phi_{e_t}(P)$, are a function of the second joint's position and velocity.

Assuming an input pressure to the shoulder bicep PM $P_{bs} = P_{0bs} + \Delta p_s$ and an input to the shoulder tricep PM $P_{ts} = P_{0ts} - \Delta p_s$, substituting this into Eqn. (7) produces Eqn. (9):

$$\begin{aligned}&\tau_s(P_b = P_{0bs} + \Delta p_s, P_t = P_{0ts} - \Delta p_s)\\ &= n_s \cdot r_s \cdot [F_{ts}(P_{0ts} - \Delta p_s) - K_{ts}(P_{0ts} - \Delta p_s) \cdot x_{ts}\\ &\quad - B_{ts}(P_{0ts} - \Delta p_s) \cdot \dot{x}_{ts}] - [F_{bs}(P_{0bs} + \Delta p_s)\\ &\quad - K_{bs}(P_{0bs} + \Delta p_s) \cdot x_{bs} - B_{bs}(P_{0bs} + \Delta p_s) \cdot \dot{x}_{bs}]\end{aligned} \tag{9}$$

Then, for each $F(P)$, $K(P)$, and $B(P)$ expression, make the substitution according to Eqns. (2)-(4) and group the Δp_{PM} terms together, e.g.:

$$\begin{aligned}F_{ts}(P_{0ts} - \Delta p_s) &= F_{0ts} + (F_{1ts} \cdot (P_{0ts} - \Delta p_s))\\ &= [-F_{1ts} \cdot \Delta p_s] + [F_{0ts} + F_{1ts} \cdot P_{0ts}]\end{aligned} \tag{10}$$

Doing so for each produces Eqn. (11):

$$\begin{aligned}&\tau_s(P_b = P_{0bs} + \Delta p_s, P_t = P_{0ts} - \Delta p_s) = n_s \cdot r_s \cdot [\\ &\Delta p_s \cdot [-F_{1ts} + (x_{ts} \cdot K_{1ts}) + (\dot{x}_{ts} \cdot B_{1ts})\\ &- F_{1bs} + (x_{bs} \cdot K_{1bs}) + (\dot{x}_{bs} \cdot B_{1bs})] + F_{0ts}\\ &+ (F_{1ts} \cdot P_{0ts}) - (x_{ts} \cdot (K_{0ts} + (K_{1ts} \cdot P_{0ts})))\\ &- (\dot{x}_{ts} \cdot (B_{0ts} + (B_{1ts} \cdot P_{0ts}))) - F_{0bs}\\ &- (F_{1bs} \cdot P_{0bs}) + (x_{bs} \cdot (K_{0bs} + (K_{1bs} \cdot P_{0bs})))\\ &+ (\dot{x}_{bs} \cdot (B_{0bs} + (B_{1bs} \cdot P_{0bs})))]\end{aligned} \tag{11}$$

Doing the same for Eqn. (8) and assuming an input pressure to the elbow bicep PM $P_{be} = P_{0be} + \Delta p_e$ and an input to the elbow tricep PM $P_{te} = P_{0te} - \Delta p_e$ produces Eqn. (12):

$$\begin{aligned}&\tau_e(P_b = P_{0be} + \Delta p_e, P_t = P_{0te} - \Delta p_e) = n_e \cdot r_e \cdot [\\ &\Delta p_e \cdot [F_{1be} - (x_{be} \cdot K_{1be}) - (\dot{x}_{be} \cdot B_{1be})\\ &+ F_{1te} - (x_{te} \cdot K_{1te}) - (\dot{x}_{te} \cdot B_{1te})] + F_{0be}\\ &+ (F_{1be} \cdot P_{0be}) - (x_{be} \cdot (K_{0be} + (K_{1be} \cdot P_{0be})))\\ &- (\dot{x}_{be} \cdot (B_{0be} + (B_{1be} \cdot P_{0be}))) - F_{0te}\\ &- (F_{1te} \cdot P_{0te}) + (x_{te} \cdot (K_{0te} + (K_{1te} \cdot P_{0te})))\\ &+ (\dot{x}_{te} \cdot (B_{0te} + (B_{1te} \cdot P_{0te})))]\end{aligned} \tag{12}$$

This is the derivation for Eqns. 3.6-3.7 from [1], but important to note is that these results do not agree. To put the findings here in that form would produce Eqns. (13)-(18):

$$\tau_s(P_b = P_{0bs} + \Delta p_s, P_t = P_{0ts} - \Delta p_s) = \tau_{0_s} + (\Delta p_s \cdot \tau_{1_s}) \tag{13}$$

$$\tau_e(P_b = P_{0be} + \Delta p_e, P_t = P_{0te} - \Delta p_e) = \tau_{0_e} + (\Delta p_e \cdot \tau_{1_e}) \tag{14}$$

$$\begin{aligned}\tau_{0_s} = n_s \cdot r_s \cdot [&F_{0ts} + (F_{1ts} \cdot P_{0ts})\\ &- (x_{ts} \cdot (K_{0ts} + (K_{1ts} \cdot P_{0ts})))\\ &- (\dot{x}_{ts} \cdot (B_{0ts} + (B_{1ts} \cdot P_{0ts})))\\ &- F_{0bs} - (F_{1bs} \cdot P_{0bs})\\ &+ (x_{bs} \cdot (K_{0bs} + (K_{1bs} \cdot P_{0bs})))\\ &+ (\dot{x}_{bs} \cdot (B_{0bs} + (B_{1bs} \cdot P_{0bs})))]\end{aligned} \tag{15}$$

$$\begin{aligned}\tau_{1_s} = n_s \cdot r_s \cdot [&-F_{1ts} + (x_{ts} \cdot K_{1ts}) + (\dot{x}_{ts} \cdot B_{1ts})\\ &- F_{1bs} + (x_{bs} \cdot K_{1bs}) + (\dot{x}_{bs} \cdot B_{1bs})]\end{aligned} \tag{16}$$

$$\begin{aligned}\tau_{0_e} = n_e \cdot r_e \cdot [&F_{0be} + (F_{1be} \cdot P_{0be})\\ &- (x_{be} \cdot (K_{0be} + (K_{1be} \cdot P_{0be})))\\ &- (\dot{x}_{be} \cdot (B_{0be} + (B_{1be} \cdot P_{0be})))\\ &- F_{0te} - (F_{1te} \cdot P_{0te})\\ &+ (x_{te} \cdot (K_{0te} + (K_{1te} \cdot P_{0te})))\\ &+ (\dot{x}_{te} \cdot (B_{0te} + (B_{1te} \cdot P_{0te})))]\end{aligned} \tag{17}$$

$$\begin{aligned}\tau_{1_e} = n_e \cdot r_e \cdot [&F_{1be} - (x_{be} \cdot K_{1be}) - (\dot{x}_{be} \cdot B_{1be})\\ &+ F_{1te} - (x_{te} \cdot K_{1te}) - (\dot{x}_{te} \cdot B_{1te})]\end{aligned} \tag{18}$$

Eqns. (16) and (18) do not quite agree with Eqns. 3.7b and 3.7d from [1]. Inspection of Eqns. (1) and (2)-(4) would reveal that the signs for the K_{PM_1} and B_{PM_1} terms must be the same for Eqns. (16) and (18), which is the case here but not Eqns. 3.7b and 3.7d in [1]. As such, it is believed the equations here are in the correct format.

With these values derived, the sliding-mode control law can be implemented with an estimation of the sliding-mode gain matrix G and vector a using Eqns. 3.9 and 3.10 from [1], with the commanded pressures calculated via Eqn 5.10 from the same.

III. Simulation

Development began for a simulation suite which sits on top of the *ROS2* framework, and more specifically, utilizes *ros2_control* and *MoveIt 2*. *ros2_control* is a comprehensive hardware-agnostic framework for robotic control, which offers modular solutions for, among other things, trajectory control and simulated hardware interfaces, both of which were utilized for this project. *MoveIt 2* is the robotic manipulation platform used to connect all the components in the simulation pipeline, including some boiler-plate functionality as well as the custom components designed specifically for this project.

Figure III shows an example, high-level block diagram of a system architecture such as this one. For this project, custom interfaces were designed to take the role of the "arm_controller" and "Simulation HW Interface" blocks as seen in this diagram (though they have different command and state interfaces as the ones shown here, this is described in more detail in the upcoming sections).

Unfortunately, while considerable progress was made on a full simulation suite for the type of robotic manipulator described in the previous sections, it could not be finished at this time. All algorithms have been implemented, and all *ros2_control* and *MoveIt 2* connections have been made, but it does not perform as expected. In other words, there is a lingering bug in the design that could not be isolated in time. The following subsections describe the solutions attempted (including its current development state), as well as plans for future development.

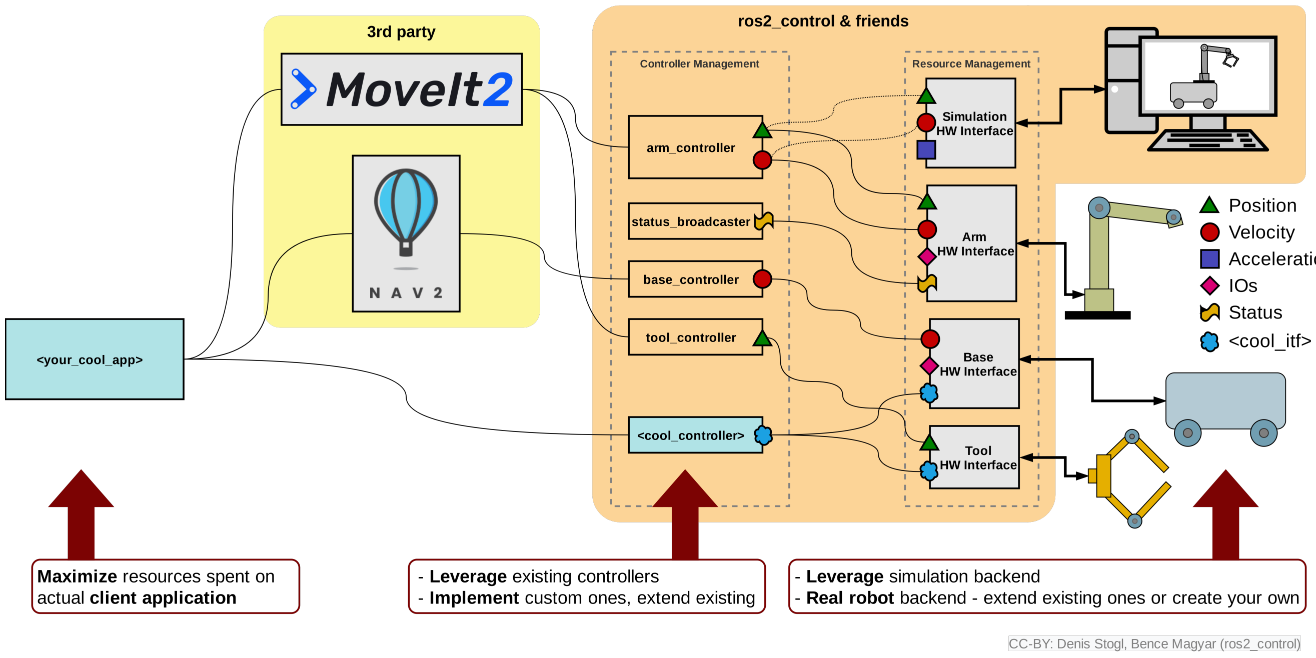

Fig. 3. Overview of *ros2_control*. Source: [2]

A. Current Design

First, a URDF was created for a simple two DoF planar arm, and the *MoveIt Setup Assistant* was used to create a complete description of this robot model, such that it was ready to be used by *MoveIt 2* (joint-space and Cartesian-space trajectory planning, simulated behavior, etc). From here, the runtime files could be updated to substitute custom *ros2_control* items as needed.

Aside from various utilities to collect and publish trajectory control telemetry, maintain controller and PMA segment configuration data, etc., the remainder of development went towards two custom *ros2_control* items, as described in the following subsections.

1) Custom Pressure-Trajectory Controller: The "pressure-trajectory" controller developed specifically for two DoF planar robot arms consisting of PMA segments uses the robot model and sliding-mode control algorithm described in [1] to maintain the joint-space trajectory. This is where the above equations, as well as the relevant ones found in [1], are implemented. Once the next desired pressures are computed, the dynamics model is used one step further to calculate the joint accelerations that result from this (Eqn. 3.8 in [1]), and these are what are commanded to the simulated hardware interface (see Section III-A2).

This *ROS2* node also maintains a number of parameters via the *ROS2* parameter server interface so the simulation is fully dynamically configurable. The parameters default to the "ideal" values as described in Section VI of [1], but are configurable on runtime so any desirable system configuration could be simulated. Their names and default values can be found in Tables I - IV.

Parameter Description	Default Value
Acceleration due to gravity ($\frac{m}{s^2}$)	9.81
k1 sliding-mode control design constant (unitless)	50.0
k2 sliding-mode control design constant (unitless)	50.0
Γ1 sliding-mode control design constant (unitless)	1.0
Γ2 sliding-mode control design constant (unitless)	1.0
μ1 sliding-mode control design constant (unitless)	5.0
μ2 sliding-mode control design constant (unitless)	5.0

TABLE I
CONFIGURABLE SIMULATION DATA FOR THE SLIDING-MODE CONTROLLER AND THEIR DEFAULT VALUES.

Parameter Description	Default Value
shoulder segment mass (kg)	10.0
shoulder segment pulley radius (m)	0.0762
shoulder segment overall length (m)	0.46
shoulder segment length to CoM (m)	0.23
shoulder segment antagonist PM pairs (count)	6
shoulder bicep nominal pressure (kPa)	310.3
shoulder tricep nominal pressure (kPa)	449.6

TABLE II
CONFIGURABLE SIMULATION DATA FOR THE ROBOT SHOULDER SEGMENT AND THEIR DEFAULT VALUES.

The design for this node was inspired by the joint trajectory controller offered by *ros2_control*, see those libraries for its implementation.

2) Custom Simulated PMA Robot Hardware Interface: This is a simple hardware interface which accepts commanded joint accelerations. The simulation is performed when "reading" the simulated acceleration by accepting the commanded value. The change in joint velocity and position is estimated given this change in acceleration, and the overall exposed state telemetry are the joint positions, velocities, and accelerations.

The design for this node was inspired by a demo interface

Parameter Description	Default Value
elbow segment mass (kg)	10.0
elbow segment pulley radius (m)	0.0508
elbow segment overall length (m)	0.46
elbow segment length to CoM (m)	0.23
elbow segment antagonist PM pairs (count)	3
elbow bicep nominal pressure (kPa)	310.5
elbow tricep nominal pressure (kPa)	310.3

TABLE III
CONFIGURABLE SIMULATION DATA FOR THE ROBOT ELBOW SEGMENT AND THEIR DEFAULT VALUES.

Parameter Description	Default Value
F_0 constant (unitless)	179.2
F_1 constant (unitless)	1.39
K_0 constant (unitless)	5.71
K_1 constant (unitless)	0.0307
B_0 constant (while inflating) (unitless)	1.01
B_1 constant (while inflating) (unitless)	0.00691
B_0 constant (while deflating) (unitless)	0.6
B_1 constant (while deflating) (unitless)	0.000803

TABLE IV
CONFIGURABLE SIMULATION DATA FOR EACH PM (SHOULDER BICEP, SHOULDER TRICEP, ELBOW BICEP, AND ELBOW TRICEP) AND THEIR DEFAULT VALUES.

offered by *ros2_control*.

B. Attempted Design

The initial design for this simulation suite also consisted of a custom trajectory controller and a custom hardware interface, but the state and command interface "connections" were different. One of the improvements made while designing the *ros2_control* utilities was that custom types of hardware interfaces could be defined. Its toolbox still comes with out-of-the-box solutions for common hardware interfaces (joint positions, velocities, and accelerations), as some combination of these are usually all that is needed for trajectory control for standard manipulators. But the ability to easily insert custom interfaces is also supported for outlier cases.

For this project, it would have been more appropriate if this custom trajectory controller had a single command interface of type "pressure", in units of e.g. *kPa*, for each joint in the PMA robot arm. Then, the simulated hardware interface could simulate the system's response to such, and output joint positions, velocities, and accelerations according to its dynamics model. This leaves the "pressure trajectory controller" agnostic to the hardware interface that it is commanding pressures to, so a theoretical hardware interface for a physical version of a robot manipulator of this type could be swapped in for the simulated counterpart. The simulated hardware interface can simulate the state telemetry, while a physical hardware interface would measure it.

There were issues with this approach that made implementation difficult on a timeline with a tight deadline. This is due largely to the way that robot hardware interfaces in *ros2_control* are not a node-based class and have limited communication with the *ROS2* network, and simulating a response in this hardware interface without access to the additional metrics produced by the trajectory controller did not seem feasible.

C. Future Development

Future development would ideally be to revisit the original approach from Section III-B and continuing that route for the custom trajectory controller and simulated hardware interface. Next step would be to find a solution to command Cartesian-space trajectories into *MoveIt 2* that the custom trajectory controller can understand (i.e. *MoveIt 2* automatically samples the trajectory in Cartesian-space and creates a corresponding one in joint-space, or the custom controller also caters to *ROS2* Cartesian trajectory control and does the transformation to joint-space trajectories itself). From there, a simple node will listen to the telemetry output by the trajectory controller that is specific to PMA robot models and plot the robot's trajectory in joint-space and "pressure-space".

IV. DISCUSSION AND CONCLUSION

The theory and implementation described in [1] was analyzed and summarized, and possible sources of human error throughout that article were discussed here. The model for a two DoF planar robot arm consisting of segments actuated entirely by PM's was derived in a way that a sliding-mode trajectory controller could be designed for it. A simulation suite that utilizes *ros2_control* and *MoveIt 2* was designed with some initial success, but has requires for development for a complete solution. A custom trajectory controller was developed specifically for a robot with the model discussed here, and a hardware interface was developed for the sake of simulating feedback. Other utilities were developed to support the full simulation pipeline.

V. LIST OF ACRONYMS

DoF	degree of freedom	1
PM	pneumatic muscle	1
PMA	pneumatic muscle actuator	1

REFERENCES

[1] J. H. Lilly and P. M. Quesada, "A Two-Input Sliding-Mode Controller for a Planar Arm Actuated by Four Pneumatic Muscle Groups," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 12, no. 3, pp. 349–359, September 2004.
[2] ros2_control Maintainers, https://control.ros.org/master/doc/resources/resources.html, 2022.