1-2018

Modeling of a Dynamic McKibben Style Muscle System Using Material Properties

Alexander G. Arcus
aga2952@rit.edu

Modeling of a Dynamic McKibben Style Muscle System Using Material Properties

By Alexander G. Arcus

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering

Supervised by

Dr. Kathleen Lamkin-Kennard

Kate Gleason College of Engineering

Department of Mechanical Engineering

Rochester Institute of Technology

Rochester, NY

January 2018

Approved by:

_____

Dr. Kathleen Lamkin-Kennard
Thesis Advisor, Department of Mechanical Engineering

_____

Dr. Agamemnon Crassidis
Committee Member, Department of Mechanical Engineering

_____

Dr. Jason Kolodziej
Committee Member, Department of Mechanical Engineering

_____

Dr. Michael Schrlau
Department Representative, Department of Mechanical Engineering

Acknowledgements

Thanks to my advisor Dr. Kathleen Lamkin-Kennard for helping to make my vision for this project a reality. One day I came to her office with an idea for what I believed to be a good idea for a thesis. With her help that idea has flourished and finally been completed.

Special thanks to Dr. Steven Day for his help in better understanding the McKibben style muscle as a physical system. Thank you to Dr. Agamemnon Crassidis for providing the code for PID tuning.

Abstract

The McKibben style artificial muscle is a type of pneumatic artificial muscle. When the muscle is pressurized a length contraction and contractive force occur. Modeling a McKibben style muscle presents many challenges. The physical dynamics of the muscle are highly nonlinear, which makes accurate modeling difficult. Modeling using experimentally determined coefficients enables the creation of an equation that ignores the nonlinear nature of the system, but it can have a limited range or accuracy and cannot adjust for changing system properties. Modeling using the system properties requires the consideration of the nonlinear dynamics of the muscle; however, it provides the ability to run numeric simulations before constructing muscles. This thesis focused on dynamic modeling of a two muscle system using system properties.

In order to develop a two muscle dynamic system model, the following steps were taken. Models using physical properties for individual static muscles were examined. Two muscles were linked around a pulley to form a two muscle system with an output of angular displacement and torque. Finally, the effect of valves was added to allow for modeling of transient system response.

In order to validate the model, an experimental system was constructed. The muscle system was simulated using MATLAB and outputs were compared to experimental results. Good agreement between theoretical and experiment results was obtained. A PID controller was then implemented on the new model to demonstrate the feasibility of using the model for control of a two muscle system. The controller was run through an optimization routine to determine the gains which gave the least position error.

This work is the first to provide a dynamic model for a system of two of opposed McKibben style muscles based on the physical properties of the muscle system.

Table of Contents

Nomenclature

F = force

P = pressure

L = muscle length

L_0 = initial muscle length

n, N = number of braid wraps on muscle

b = braid strand length

r, R_p = radius of pulley (opposed muscle system)

α, θ = angle of rotational displacement around pulley (opposed muscle system)

T = torque

y = muscle length

m, M = mass

B = damping effect coefficient

K = spring element coefficient

F_{cc} = contractile force

g = acceleration due to gravity

D = muscle diameter

D_0 = initial muscle diameter

ϵ = contraction ratio

a, A = experimentally determined parameter

b, B = experimentally determined parameter

C = experimentally determined parameter

D = experimentally determined parameter

Δp = driving pressure

θ = braid angle

D_0 = muscle diameter at braid angle = 90^o

t_k = braid thickness

D_s = diameter of braid srand

E = elastic modulus of the braid material

E_R = elastic modulus of the muscle bladder

V_b = volume of muscle

t = thickness of tube

V = volume

R = specific gas constant

T = absolute temperature

PID = Proportional Integral and Derivative

PWM = Pulse Width Modulation

Table of Figures

Introduction

The McKibben style muscle was first developed in 1950 by Joseph L. McKibben [1]. The artificial muscle was originally intended for orthopedic use to aid weakened muscles in the upper arm. The muscle uses a large amount of compressed air for actuation, requiring a gas tank of significant size to accompany the user. This form of utilization and the muscle itself fell out of favor as electric motors became smaller and easier to use. Electric motors can use light weight batteries instead of a bulky air tank. Recent works have turned back to the McKibbben style muscle for use as a biomimetic actuator [19]. Klute et al. show the ability of the muscle to mimic the properties of an organic muscle through the addition of a damping element [2]. Dzahir et al. have returned to the study of the McKibben muscle for applications in orthotics, now focusing on gait training rather than the original use in hand muscle strengthening [3].

Figure 1 shows a typical construction of a McKibben style muscle, which consists of an elastic bladder (rubber tube in this case) with a surrounding braided sheath [4]. One end of muscle has a fitting to allow for the input and exhaust of air, while the other end is sealed. The actuation of the McKibben style muscle comes from an increase in internal pressure, as seen in Figure 2. As the internal pressure increases, the muscle radius increases. The radial increase creates a tension in the braid, causing a length contraction and contractile force.

The basic equation for modeling a McKibben style muscle comes from an analysis of the system using virtual work. The concept takes all the energy provided by the input pressure and transforms it into the length contraction and force production [5] such that

$$F = \frac{P}{4n^2\pi}(3L^2 - b^2) \tag{1}$$

where F is force, P is gauge pressure and L is muscle length. In Equation (1), n and b are muscle constants that refer to number of wraps and braid strand length respectively. The constants are associated with the specific braid material chosen for use in the muscle.

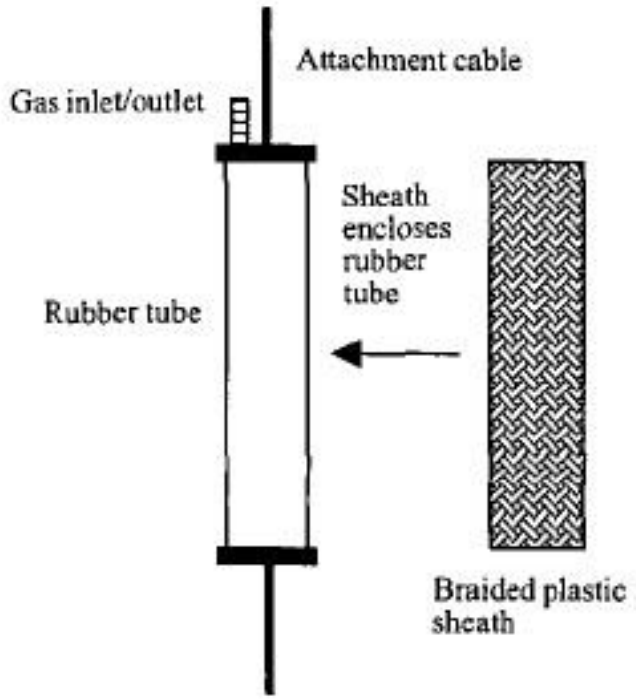


Figure 1: Construction of the McKibben artificial muscle [6].

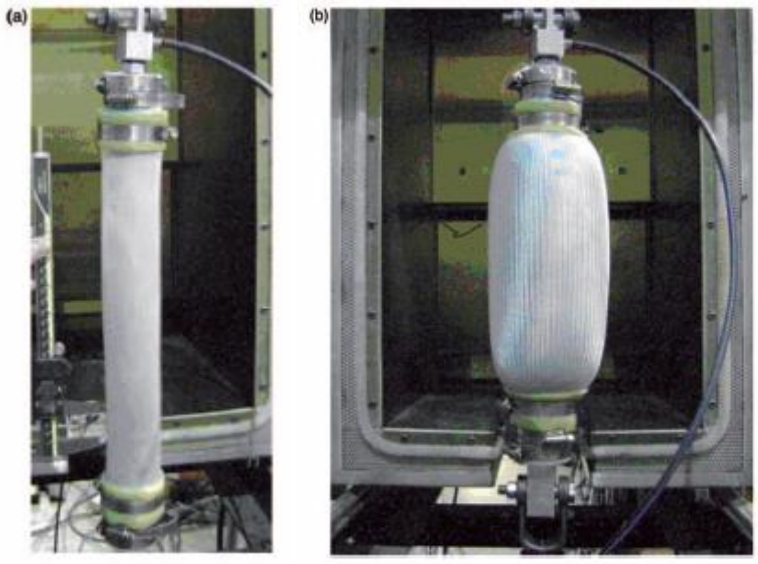


Figure 2: (a) McKibben style muscle at rest and (b) Pressurized muscle [3].

This thesis examines the mathematical modeling of a dynamic system consisting of an opposed McKibben style muscle pair for robotic applications. The setup mimics the human form so it lends itself to use in robotic armatures.

Problem Statement

Current models of McKibben style muscles do not address the needs for modeling a system containing a pair of opposed McKibben style muscles, as seen in Figure 3. The opposed muscle system requires a model which can predict its dynamic nature, but currently no adequate model exists. Most previously developed models are for a single, static muscle [7-8]. The models perform well under static conditions, but have significantly different behavior when given a dynamic excitation. A limited number of dynamic models of the McKibben style muscles have been developed using experimentally determined parameters to identify the dynamic effects not considered in the static models [7-8]. The experimentally developed models show good agreement with experimental data; however, the models do not have the sizing and scaling capabilities that a model derived from physical properties would have to adequately predict system dynamics.

The goal of this work was to develop a dynamic model based on the physical characteristics of the muscle and the system. These characteristics include the muscle length, muscle radius and the materials used for the muscle and braid. The model should provide a good estimate of muscle output (force, length) to allow for sizing and scaling before muscle creation. Another benefit of having the opposed muscle model is the ability to assist in automation. To be successful in automation, the model must be small in size so that it is able to be run concurrently with real time sensor readings. Running concurrently allows the model to provide interpolation between sensor readings and to detect malfunctions in the sensors. A successful model should provide good correlation with experimental results using as few terms as possible.

The objectives of this thesis were thus:

- Creation of a dynamic muscle model
- Integration of the muscle model into a system model
- Validation of the system model using an experimental apparatus
- Implementation of a simple PID controller in simulation to demonstrate the feasibility of controlling a muscle system with respect to a desired angle

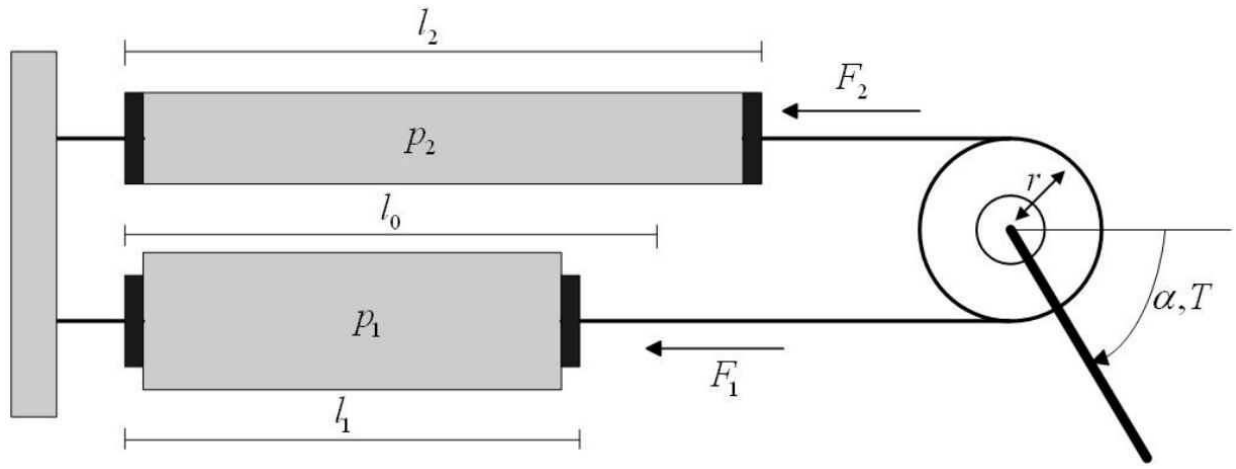


Figure 3: Opposed McKibben style muscle pair in a system generating torque and rotational displacement [9].

Modeling the McKibben Style Muscles

Equation (1) mathematically represents an idealized form of a McKibben style muscle based on energy balance. A real system has inefficiencies which cause an error in predicted force produced of up to 30% when compared to data generated from Equation (1) [5]. To reduce the error, there are two major ways to adjust the model. The first method uses experimental results from actual muscles to determine coefficients for a desired equation. The second method utilizes additional corrective terms based on the physical properties of the system. These terms are added to Equation (1). This thesis focuses on the second modeling approach. The second method will generate a model which allows for manipulation of the effects of the physical properties of the system. A key benefit of a physical properties model is the ability to run numerical simulations of a muscle's expected response before construction. A discussion of both methods is included below.

Modeling with Experimental Coefficients (Method 1)

One of the major ways that the problem of muscle inefficiency is addressed computationally is through experimentally determining the model coefficients. The least complex of these models simulates the model as a linear three-element system [10] seen in Figure 4.

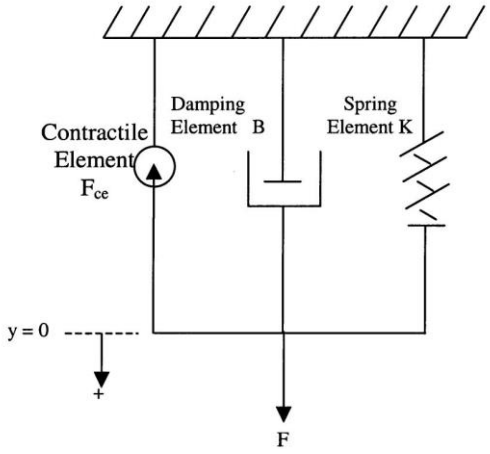


Figure 4: Physical representation of the second order linear equivalent system [10].

The model takes the form of a linear, second order differential equation with a forcing function, F_{cc}, and attached mass, such that

$$M\ddot{y} + B\dot{y} + Ky = F_{cc} - Mg \tag{2}$$

where the coefficients of the derivatives are the experimentally determined mass, M, the damping, B, and the spring constant, K. The other values are acceleration due to gravity, g, length, y, and its time derivatives, $\dot{y}$ and $\ddot{y}$. The values for B, K and F_{cc} are determined experimentally. Working with a linear model is well documented, making this an appealing choice, particularly when determining system response and applying control techniques. The method is limited by its complete dependence on the experimental coefficients. The linear estimation that Equation (2) creates will diverge from the highly nonlinear model as the system moves further from the regime used to estimate the coefficients.

Another way that the muscle can be modeled is by using finite element analysis. Bertetto et al. used a finite element mesh to create a model for the McKibben style muscle [11]. The model proposed by Bertetto et al. uses a mesh to identify positions (nodes) on the muscle. After applying a force to the system, the model evaluates the displacements caused at the set nodes. This technique allows for a better tracking of the deformation, which is large for this type of system. Calculating the deformation function, and thereby the stress function, creates a model in good agreement with experimental data obtained from the muscle. The limitation of a finite element mesh, however, is the number of elements required to produce valid results. Modeling a system with a high strain like the McKibben style muscle

requires many elements, which increases the simulation run time and chance of error in correct element definition, especially around the endcaps of the muscle.

Schroder et al. modeled an opposed muscle system using a hybrid equation developed from physical properties and experimentally determined coefficients [9]. The system is generated from the Bridgestone model of the energy balance

$$F = D_0^2 p[a(1-\epsilon)^2 - b] \tag{3}$$

where F is the force generated, P is the gauge pressure of the muscle, D_0 is initial muscle diameter and ϵ is the contraction ratio. Here *a* and *b* are parameters determined through experimentation. The model is then transformed into a torque model to describe the system seen in Figure 3 where the torque T is calculated using

$$T = \frac{2D_0^2 ar^3}{l_0^2}\Delta p\alpha^2 - \frac{2D_0^2(p_1+p_2)ar^2}{l_0}\alpha + D_0^2 r(2a-2b)\Delta p \tag{4}$$

Equation (4), developed by Schroder et al., however, represents a static model for a symmetric, opposed muscle system. To determine a dynamic representation, a function dependent on the angular velocity must be added. In Equation (4), the angular displacement is assumed to be small so the first term with alpha squared is assumed to be too small to have a significant impact on the equation. Schroder et al. consolidated the coefficients in Equation (4) and added a correction function producing

$$T = Ar^2\alpha + Br\Delta p + f(\dot{\alpha}) \tag{5}$$

Experimentation determined that the correction function has the shape of a third order polynomial centered around the origin. Schroder et al. [9] assigned experimentally determined coefficients to the correction functions producing

$$T = Ar^2\alpha + Br\Delta p + C\dot{\alpha}^3 + D\dot{\alpha} \tag{6}$$

Here, A, B, C and D are consolidations of experimentally determined and physically derived coefficients. The driven wheel is defined by the torque, T, the radius, r, and the rotation angle, α. The driving pressure is given by Δp. Results from the study by Schroder et al. [9] demonstrate that it is possible to have a model with good agreement with data from a controlled muscle pair. However, while Equation (6) shows good agreement with a torque model, it does not fulfill the requirements of the model being

developed in this thesis. Equation (6) cannot provide scaling, a property of equations based purely on the physical properties of the system.

Modeling Using Muscle Properties to Determine Coefficients (Method 2)

The earliest form of the virtual work model of the McKibben style muscle is [4], [12], [13]

$$F = \frac{\pi D_0^2 P}{4}(3cos^2\theta - 1) \tag{7}$$

In this equation, F represents the force generated by the system and P is the internal pressure. Here, θ is the braid angle and D_0 is the diameter of the muscle at a braid angle of 90°. Braid angle is difficult to measure with a sensor so Equation (1) is preferable over Equation (7) for experimental work and numerical simulations. Terms can be added to this model to account for losses in the system, such as from friction, endcap effects, braid extension and strain effects [5], [12], [14].

The use of the physical properties of the muscle in the model equations allows for numerical modeling of the muscle system before construction. Physical property based equations make quick changes to the muscle design simple to implement. The challenge associated with this modeling method is a sufficient knowledge of the dynamics and properties of the physical system.

Equation (1) is the simplest to implement in a numeric tool, but has an error of 20% to 30% due to inefficiencies in the muscle that are ignored [5]. Ching-Ping Chou et al. [5] further develop the static model to account for friction between the bladder and braid and the volume lost to tube thickness [15]. This leads to the addition of a correction term to Equation (7) to yield

$$F = \frac{\pi D_0^2 P}{4}(3cos^2\theta - 1) + \pi P\left[D_0 t_k\left(2sin\theta - \frac{1}{sin\theta}\right) - t_k^2\right] \tag{8}$$

where D_0 is the initial diameter of the muscle, θ is the braid angle and t_k is the braid thickness. Equation (8) offers an improvement to the previous model, but still produces errors of up to 15% in some cases [5].

Another source of loss in force is due to the extension of the braid. The tension force created during actuation in the braid causes a strain in the braid strands. Davis et al. [5] found that the strain in braid strand length can be found by including the effect of the force and pressure on the braid, such that

$$b = \frac{Kb_{min}+\sqrt{(Kb_{min})^2+12L^2(K+1)}}{2(K+1)} + \frac{2Pb_{min}^2 sin^2\theta}{ED_s n\pi^2} \quad (9)$$

$$\text{where} \quad K = \frac{4n^2\pi EA}{Pb_{min}L}$$

In Equation (9), b represents the braid length and b_{min} represents the braid length at rest. D_s is the diameter of a braid strand and E represents the elastic modulus of the braid material. Consideration of braid extension, while more accurate than Equation (7), can show too much reduction in the output force produced. The reduction in force is expected as part of the induced force is causing the braid length to extend.

Consideration of the stresses generated on the bladder material lead to two additional terms which were added to Equation (1) by Kothera et al.[7]. Addition of terms to account for stresses yields

$$F = \frac{P}{4n^2\pi}(3L^2 - b^2) - \frac{E_R t L^2}{2n^2\pi}\left(\frac{1}{R} - \frac{1}{R_0}\right) + E_R V_b \left(\frac{1}{L_0} - \frac{1}{L}\right) \quad (10)$$

In this equation, V_b is the volume of the muscle, R is the radius, and t is the tube thickness. E_R is the elastic modulus of the muscle bladder. Equation (10) assumes that the muscle has a cylindrical shape and a constant bladder thickness and internal volume throughout actuation. Testing against a muscle in free contraction showed an error of over 10%. To reduce the error, the assumption of constant thickness was removed to yield

$$F = \frac{P}{4N^2\pi}(3L^2 - b^2) - P\left(\frac{V_b}{L} - \frac{tL^2}{\pi DN^2}\right) - E_r V_b \left(\frac{1}{L_0} - \frac{1}{L}\right) - \frac{E_R L}{\pi DN^2}(tL - t_0 L_0) \quad (11)$$

Equation (11), when used with a length adjustment to account for tip effects, provides the most accurate prediction of muscle behavior of current models with an error of less than 10%. While Equation (11) model is effective for reducing the error in the force output, it is lengthy and only predicts the output of a single muscle. For computational efficiency, a new model is required which has fewer terms and can be adapted to a two muscle system.

Creation of a Dynamic Muscle Model

Creation of a new model begins with consideration of the previous work. The previous equations need to be manipulated in order to create a suitable model for the two muscle system discussed in this thesis.

The new model will be numerically simulated to test its ability to predict the response of the muscle system.

Physical System Components

Table 1: Muscle properties

	Muscle 1	Muscle 2
Initial length	14 cm	14 cm
Initial Radius	0.5 cm	0.5 cm
Bladder thickness	0.1 cm	0.1 cm
Initial braid angle	20 deg	20 deg
Bladder elastic modulus	$0.001*10^9$ Pa	$0.001*10^9$ Pa

The basic system consists of two pressure valves and two McKibben style muscles connected by a cord which is wrapped around a pulley as seen in Figure 5. The muscles used for the experimental system have the properties displayed in Table 1. Before initial pressurization there is some slack in the connecting cord allowing the muscle to contract slightly. The initial contraction is what allows pressurized muscle to be compliant during actuation.

Figure 5: McKibben style, opposed muscle system including pulley and valves.

In Figure 5, P1 and P2 are the pressures of muscle 1 and muscle 2, respectively. F1 and F2 are the forces generated by muscle 1 and muscle 2, respectively. Consideration of the properties of the valves in the dynamic model is required for use in a real system. The valve effects determine the profile of the

pressure input which drives the muscle, making it necessary to consider in a transient system. Other works do not have the explicit inclusion of valves because valve effects can be ignored in static models and become a part of lumped experimental coefficients in experimental models.

Muscle Model Foundation

The energy balance Equation (1) is derived from

$$F = -P\frac{dv}{dL} \tag{12}$$

where P is pressure, F is force, v is volume and L is length. Equation (12) comes from the work done by a control volume.

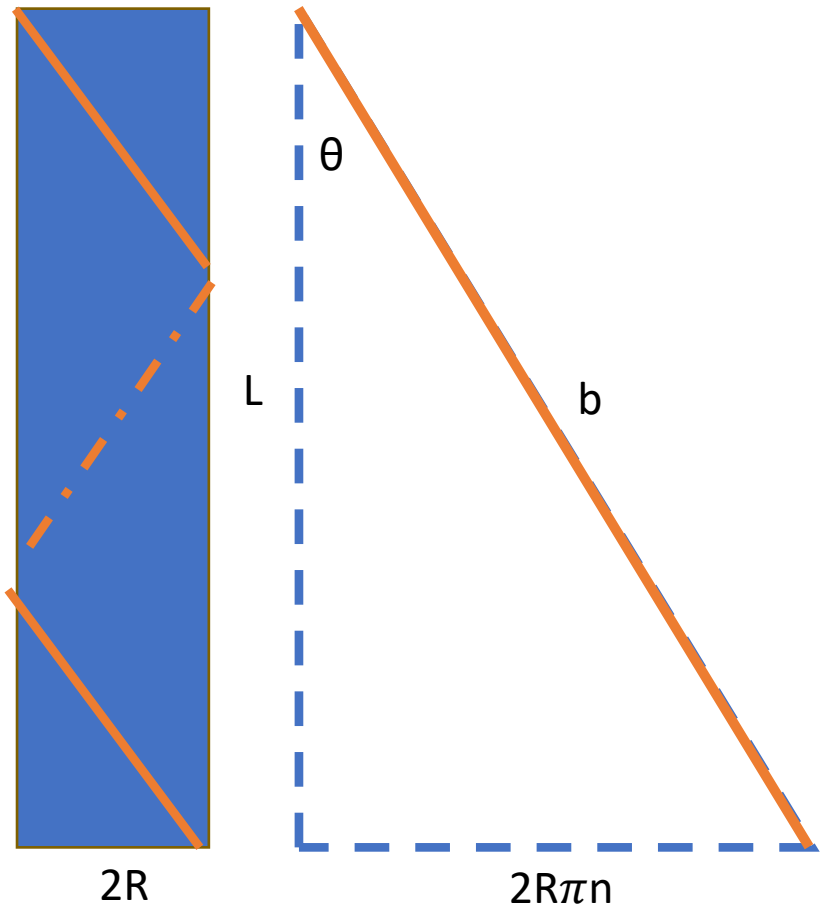


Figure 6: Braid geometry in a McKibben style muscle. (Left) A single strand of braid wrapped around the muscle. (Right) The same braid strand if cut away from the muscle and laid flat for observation.

Figure 6 shows the physical constants associated with the braid strand. Here b is the braid length and n is the number of wraps around the McKibben style muscle. The assumption of the volume of the muscle having the form of a perfect cylinder gives

$$V = \pi R^2 L \tag{13}$$

Here V is the volume, R is the muscle radius and L is the muscle length. The muscle length, L, can be found from Figure 6 to be

$$L = b\cos\theta \tag{14}$$

The muscle radius can be derived as

$$R = \frac{b\sin\theta}{2\pi n} \tag{15}$$

Substituting Equations (13-15), into Equation (12) gives

$$F = \frac{P}{4n^2\pi}(3L^2 - b^2) \tag{1}$$

Equation (1) is the form that the energy balance equation takes when the dynamics of the muscle are substituted into the equation. Caldwell et al. [16] used this method of "opening" the muscle to help determine the characteristics inherent in the muscle design.

Exploring Dynamic Use of a Static Muscle Model

Determining the dynamic characteristics of a McKibben style muscle is necessary for developing an appropriate model. The dynamic characteristics can be found by observing the behavior of the derivatives of the function. From the energy balance equation, Equation (1), three variables have a time derivative: force, pressure and length. Taking the derivative of Equation 1 yields

$$\dot{F} = \frac{\dot{P}}{4n^2\pi}(3L^2 - b^2) + \frac{6PL\dot{L}}{4n^2\pi} \tag{16}$$

However, Equation (16) cannot be implemented in simulation in its current form because it contains one input variable, pressure, and two interdependent output variables, force and length. A second equation is needed to define another relationship between the variables. The equation chosen for this in other studies [17] is the ideal gas law, which states,

$$PV = mRT \tag{17}$$

where P is the absolute pressure, V is the control volume, m is the mass, R is the dry air specific gas constant and T is the absolute temperature. Limited success was achieved with this method in modeling of length vs. pressure [17]. As a result, a new method for establishing the relationship between variables must be found.

Modeling Considerations

A number of factors must be considered when modeling a McKibben style muscle. The following define the range of operating conditions where the model is valid for a single McKibben style muscle.

Zero Internal Pressure Condition

Modeling the pre-pressurized condition of the McKibben style artificial muscle is difficult for a simple model. Various factors contribute to this difficulty, such as, any gap between the inner bladder and the braided sheath or the conditions occurring at zero pressure. The gap between the bladder and sheath creates a “dead zone” in the muscle response. The extra pressure needed for the bladder to expand to touch the braid does no useful work and is difficult to account for when developing equations.

The zero pressure condition also causes problems. The basic Equation (1) for the muscle is zero at zero pressure. Equation (10) shows that the force output is dependent on the muscle volume, muscle length and bladder elastic modulus at zero pressure. The force applied to a muscle at zero pressure produces little strain. The effect is caused by the braid angle reaching a minimum, adding the braid elasticity to the bladder elasticity. This results in the muscle having a high ability to contract, but almost no ability to expand.

Maximum Braid Angle

The energy balance muscle model, Equation (1), shows an effective maximum braid angle. The maximum occurs at one of the zeros for force in the equation, such that

$$0 = (3L^2 - b^2) \tag{18}$$

Here L is the muscle length and b is the braid length. The maximum braid angle condition can be seen as a parallel to a motor in free run where no force is produced and increasing pressure causes no increase in length. The maximum strain of the muscle is thus determined by the maximum braid angle which is limited by the braid chosen for muscle construction.

Decomposition of the Force Term

The starting point for the numeric simulation of a single muscle was the energy balance Equation (1). In Equation (19) the force term from Equation (1) is broken down to include an axial load and axial acceleration applied to the muscle to transform the equation into a differential equation, such that

$$F_{load} - M\ddot{L} = \frac{P}{4N^2\pi}(3L^2 - b^2) \tag{19}$$

Here, M is the mass attached to the muscle and $\ddot{L}$ is the second time derivative of the length. The transformation changes the force from a dependent variable into a function of the independent variable, L, and a known function, F_{load}. The transformation allows for the use of a single equation to define the motion of a single muscle.

MATLAB simulation results show that Equation (19) produces an undamped spring-like behavior in response to a step change in pressure as shown in Figure 7. The displacement results are expected, since Equation (1) takes a form similar to the fundamental equation for a linear spring and no damping is present.

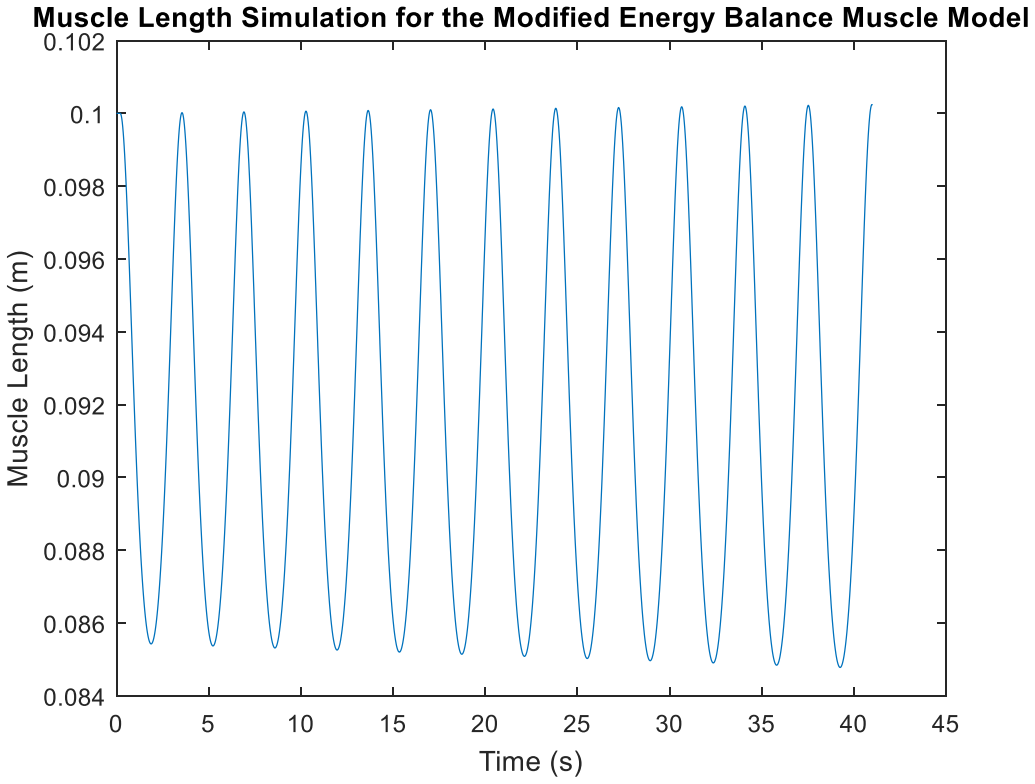


Figure 7: Simulation of a Muscle length vs. Time in response to step increase in pressure in a 10 centimeter McKibben style muscle using Equation (19).

Integration of Individual Muscle Models into a System Model

Successful creation of a single muscle model enabled the two muscle system model to be derived. Additional assumptions were needed to integrate the single muscle model into the two muscle system.

Assumptions to Simplify the System Model Equation

As previously described, the basic system model consists of two opposed muscles connected around a pulley. By adding and subtracting an equal pressure from the opposed muscles, the effective spring constant of the muscles can be approximated as a constant. The result of a constant effective spring constant is a constant tension applied to the connecting wire during actuation. Figure 8 shows the effects of the change in pressure on the system.

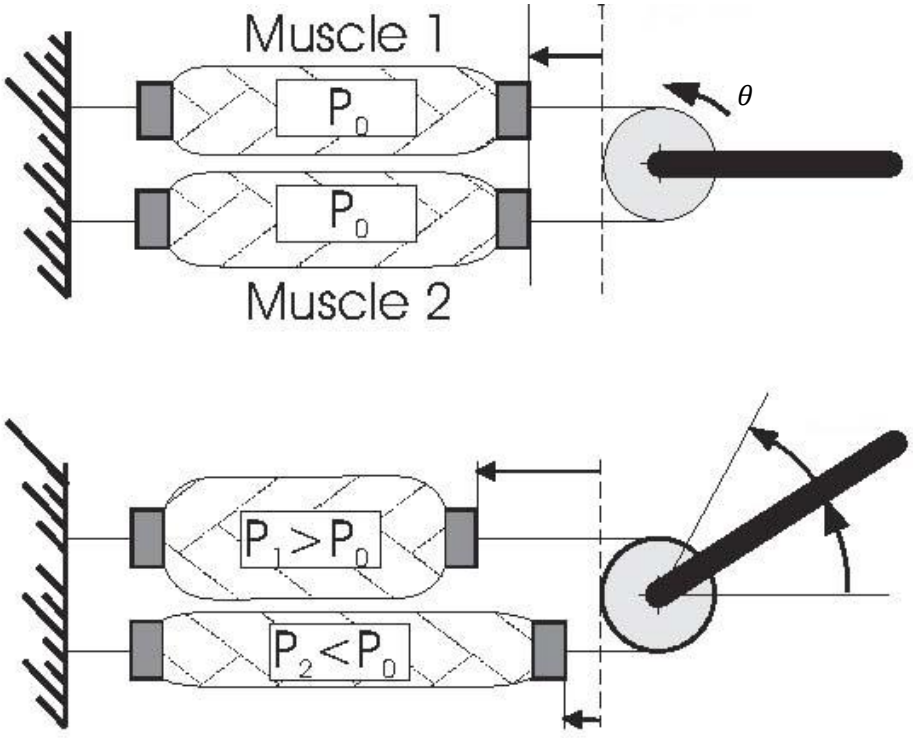


Figure 8: Effects of pressure changes in muscles 1 and 2 on the system [18].

The muscle model uses a constant value for b, the braid length. The assumption is made to reduce the required size and complexity of the muscle. If needed, Equation (9) would be used to determine braid extension.

The muscle model assumes that the muscle pressure during operation will be greater than zero. Avoiding zero pressure prevents potential negative effects in Equation (1) and Equation (10), as

previously mentioned in the muscle modeling section. These negative effects include the include any gap occurring between the braid and bladder of the muscle.

A constant muscle internal volume assumption is made to greatly simplify Equation (10). If the volume is considered to change along with the muscle, then determining the volume during operation requires an additional equation, increasing required processor time. Since the strains in the system are small and the radius increases along with a decrease in length, a constant value based on the initial volume is a reasonable simplification.

Incorporating Valves

The inclusion of the effects of valves on the system is a major factor that is required for numeric modeling of the real physical system. The consideration of the transfer of air is important because the flow rate into the muscle will not be equal to the flow rate out of the opposing muscle. In this work, valves are assumed to produce a steep ramp response of pressure when actuated.

For the system model, the simulation is started at a pressurized equilibrium state to eliminate any complications from the zero-pressure condition. Other assumptions include a constant volume in the bladder and a constant tension in the connecting strand after initial pressurization. The constant volume assumption can be used because, after initial pressurization, the muscle will maintain a similar shape due to the forces generated. This assumption greatly simplifies the dynamic model, allowing the mass flow rate to be directly related to the pressure.

As can be seen in Figure 8, without damping, the system is not useful for the prediction of the steady state system length response. To find a simple system equation with damping, Equations (1), (8) and (10) were considered. Equation (8) was discarded because braid angle was determined to be difficult to use a sensor to detect. The best length response was produced by the simulation of Equation (10) derived by Kothera et al. [7]. Equation (10) has an effective damping that is not present in Equation (1) and provides a response that is much closer to the real system. The Kothera et al. equation is tied into the muscle system through a force equivalence assumption around the muscles.

Force Linking of the Muscles

The numeric simulation of the system was accomplished by equating the forces in the opposing muscles, muscle 1 and muscle 2 in Figure 8. Linking the equations allows for the calculation of the acceleration caused by the pressure difference between the two muscles. Using this method on Equation (19), where the subscripts refer to the muscle number, gives

$$M\ddot{L}_1 + \frac{P_1}{4n^2\pi}(3L_1^2 - b^2) = M\ddot{L}_2 + \frac{P_2}{4n^2\pi}(3L_2^2 - b^2) \tag{20}$$

To incorporate damping, Equation (10) is manipulated in the same fashion as equation (20) to yield the final system model where

$$M_{eq}\ddot{L}_1 + \frac{P_1}{4n^2\pi}(3L_1^2 - b^2) - \frac{E_R t L_1^2}{2n^2\pi}\left(\frac{1}{R_1} - \frac{1}{R_{01}}\right) + E_R V_b\left(\frac{1}{L_{01}} - \frac{1}{L_1}\right) \tag{21}$$
$$= M_{eq}\ddot{L}_2 + \frac{P_2}{4n^2\pi}(3L_2^2 - b^2) - \frac{E_R t L_2^2}{2n^2\pi}\left(\frac{1}{R_2} - \frac{1}{R_{02}}\right) + E_R V_b\left(\frac{1}{L_{02}} - \frac{1}{L_2}\right)$$

The valve input is represented using a steep ramp function that represents a change from an initial pressure to the desired pressure. The slope was determined experimentally, using a linear estimation of pressure input to muscles of the experimental apparatus during operation.

Numeric Simulation of the System Model

The open loop response of the coupled muscle system to a pressure input was simulating using Simulink and MATLAB. All code and diagrams can be found in the appendixes.

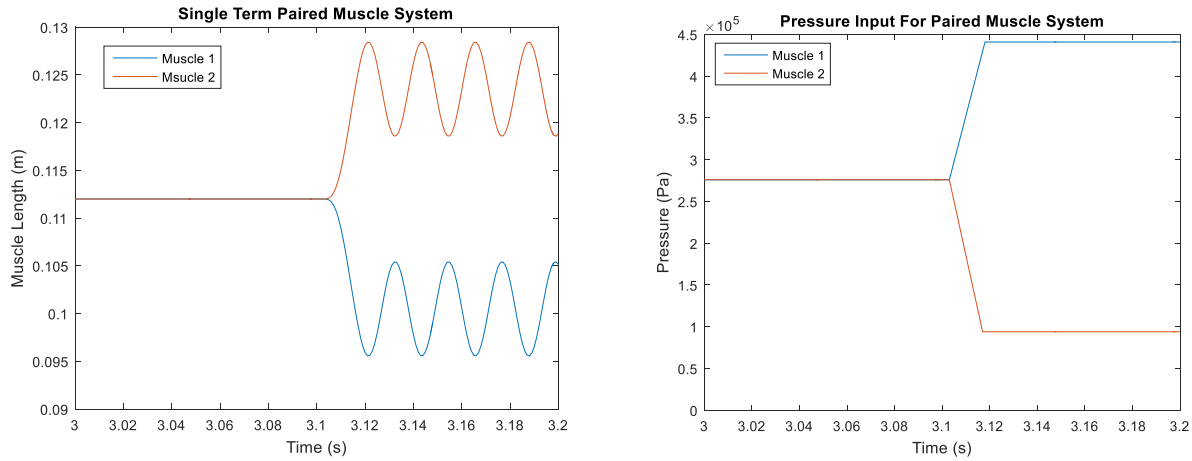


Figure 9: Open loop response of Equation (20) (left) to a steep ramp pressure input (right).

Figure 9 shows the simulated changes in muscle lengths using Equation (20) with no damping in response to a change in pressure. The graph axis has been changed to enlarge the region of induced motion. In this case, the valve dynamics were estimated using a steep ramp to go from the initial pressure to the final pressure. The oscillation that the graph depicts is expected as it mirrors the purely oscillatory output from a single muscle as seen in Figure 8.

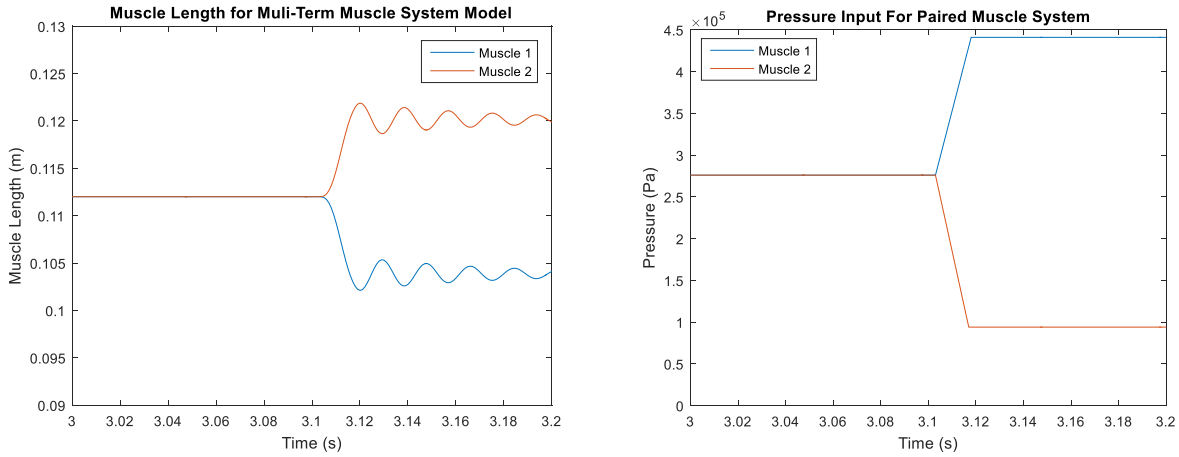


Figure 10: Open loop response of Equation (21) (left) to a steep ramp pressure input (right).

Figure 10 shows the response of Equation (21) to the same steep ramp pressure input shown in Figure 9. The left side of Figure 10 shows the expected damped length response to the pressure input.

Experimental Apparatus for Validation

A test apparatus that mimics the setup shown in Figure 5 was fabricated to validate the simulation results.

Hardware Setup

The experimental system consists of mechanical, electrical, and pneumatic systems as shown in Figures 11 and 12. The mechanical system consists of the pair of McKibben style muscles, a pulley and an actuated arm. The muscles provide the actuation for the system when pressurized. The pulley receives the actuation and moves the actuated arm (part A), seen in Figure 11. In Figure 12, parts B and C are muscle 1 and muscle 2, respectively. Part D is the pulley and part E is the potentiometer.

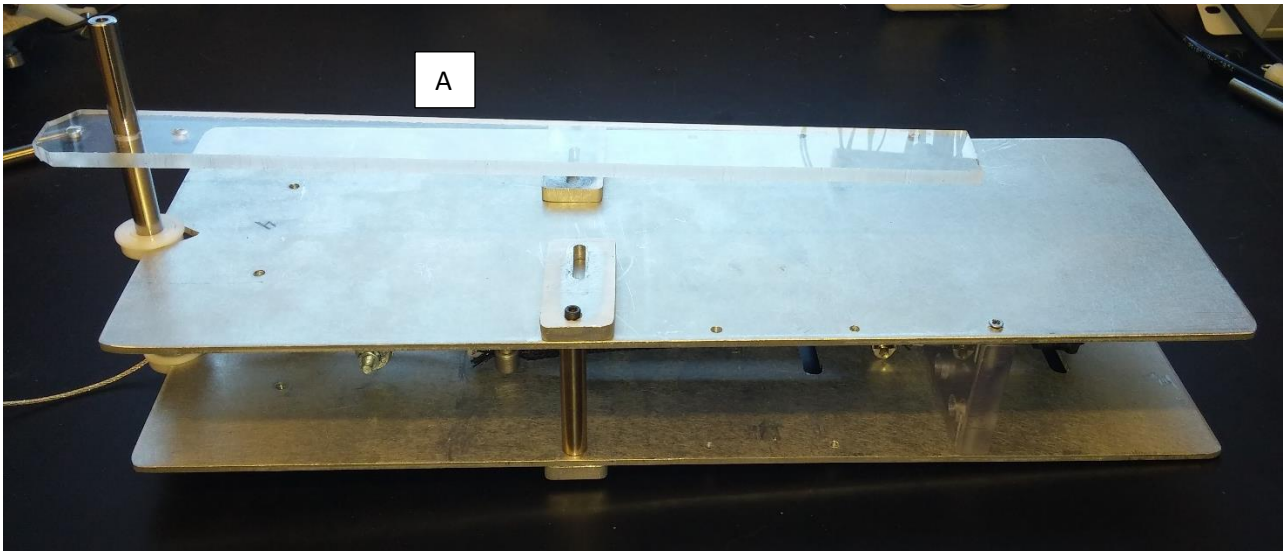


Figure 11: Experimental system setup consisting of the McKibben muscle pair and actuated arm.

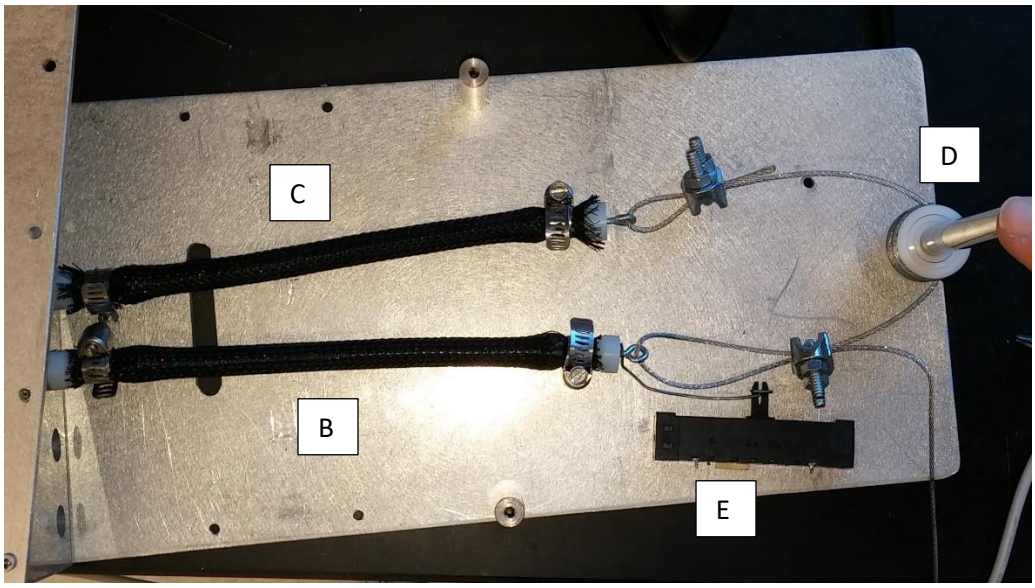


Figure 12: Internal setup of the experimental system showing the McKibben style muscles, pulley and potentiometer.

The electrical system consists of a microcontroller, battery, valves and sensors. The Arduino microcontroller passes commands to the valves and reads data from the sensors. The battery powers the system. The valves open and close based on signals passed from the microcontroller. The pressure sensors are attached to the muscles and the high-pressure tank. The linear potentiometer measures the muscle length. In the described hardware setup, a linear potentiometer is used because the wire connecting the two muscles can slip.

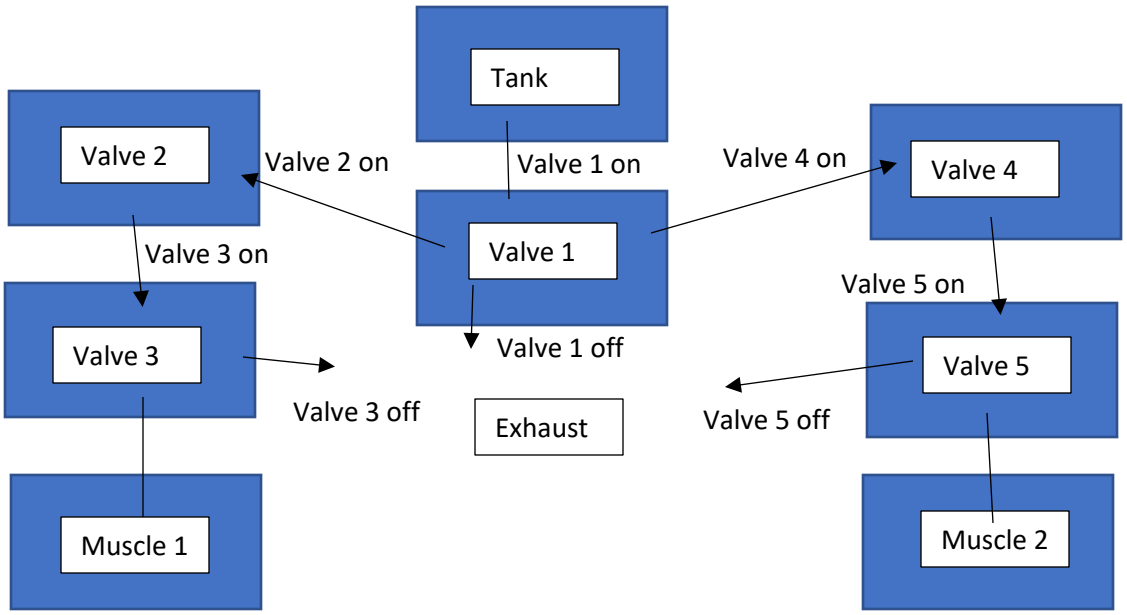


Figure 13: Pneumatic setup used for operation of the experimental apparatus. Describe more in detail what the arrows show.

The pneumatic system is shown in Figure 13 and consists of an air tank, valves and muscles. The air tank is the high-pressure source used to actuate the muscles. The valves connect the muscles to the high-pressure air source or the room pressure to control muscle actuation. The muscles actuate when pressure is increased via a connection to the high-pressure source.

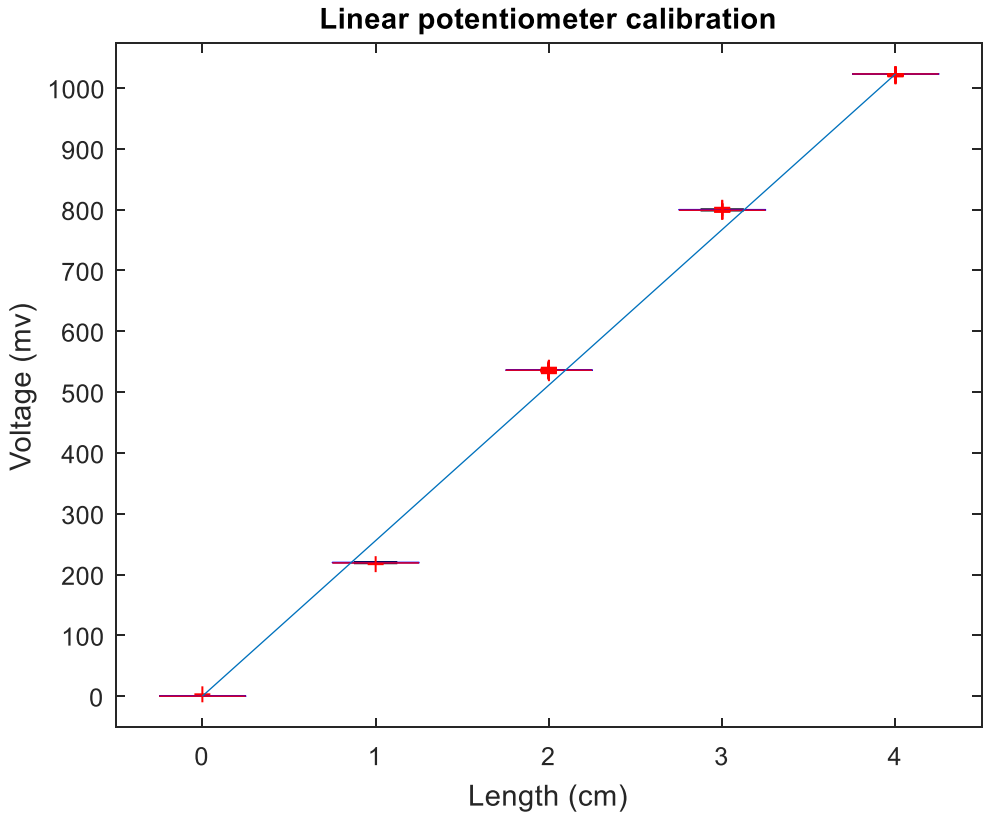


Figure 14: Linear Potentiometer calibration with error bars representing the variance of the potentiometer readings.

Figure 14 shows the calibration curve used to relate the voltage readings from the potentiometer to changes in muscle length. The points were generated by moving the potentiometer to the desired distance and holding position there. An error plot of 100 representative points can be seen in Figure 14. The curve generated is a linear estimation through the endpoints. It can be seen to have a good correlation with the potentiometer readings. The potentiometer can be seen in Figure 12 attached to muscle 1.

Experimental Apparatus Operation

The experimental system was designed to cycle through a series of desired output angles as shown in Figure 8, defined by the output from the linear potentiometer. During the setup of the system, both muscles are pressurized to a starting pressure P_0. The output angle is determined through a conversion from the output of the potentiometer attached to muscle 1. When the desired angle moves from zero to a positive angle, valve 2 opens to let pressure build in muscle 1, and valve 5 closes to let pressure release from muscle 2. When the desired angle is reached, valve 2 closes and valve 5 opens. If the

system overshoots the desired angle, valve 3 is closed and valve 4 is opened. When the system achieves the desired angle, valve 3 is opened and valve 4 is closed. The process is repeated until the system is within a specified target range around the desired angle.

If the desired angle is negative, valve 3 is closed and valve 4 is opened. When the system reaches the desired negative angle valve 3 is opened and valve 4 is closed. If the desired angle is overshot, then valve 2 is opened and valve 5 is closed. Then when the desired angle is reached, valve 2 is closed and valve 5 is opened. This process continues until the output angle is within an acceptable boundary layer of the desired angle.

In order to test the code for changing muscle length, a sequence of desired lengths was input into the program, seen in Table 2. The sequence was chosen to imitate an expected work cycle for the system. One muscle length sequence takes place every 0.7 seconds and consists of three steps. The chosen lengths provide the system with a small change, a large change and a return to the neutral position. The ability to perform these motions shows that the muscle system model is capable of prediction for more complex motion patterns.

The sequence of desired muscle lengths is shown in Table 2. The hold time for the upper length was chosen to be longer because the system undergoes a larger length change. This longer duration gives the system a longer time to stabilize before the next desired length is input. Figure 15 shows the experimental muscle length output overlaid with the desired sequence of lengths as a function of time. Figure 15 starts at 3 seconds to display an enlarged view of the best representative cycle captured by the experimental setup. The graph shows that the code is effective in controlling the valves such that the muscle reaches the desired lengths.

Table 2: Input sequence for changing muscle length.

Step	Pot Val	Muscle Length (meters)	Duration (seconds)
1	410	0.08	0.3
2	200	0.071	0.15
3	600	0.087	0.25

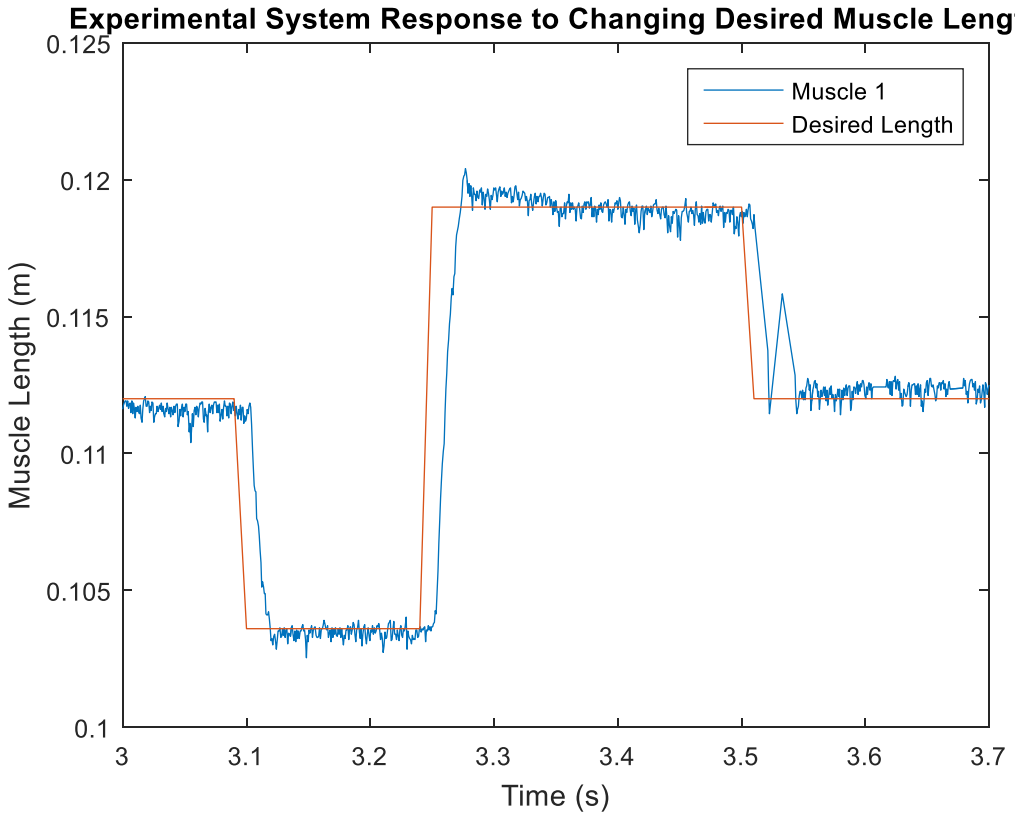


Figure 15: Response of experimental system overlaid with desired length.

The code for the Arduino microcontroller can be found in Appendix D. The sampling step size for the Arduino varies between 0.0006 and 0.0008 seconds.

The experimentally measured pressures in each muscle during actuation are shown in Figure 16. The Interpolation of the pressure changes were used to derive the steep ramp functions used as inputs to Equations (20) and (21).

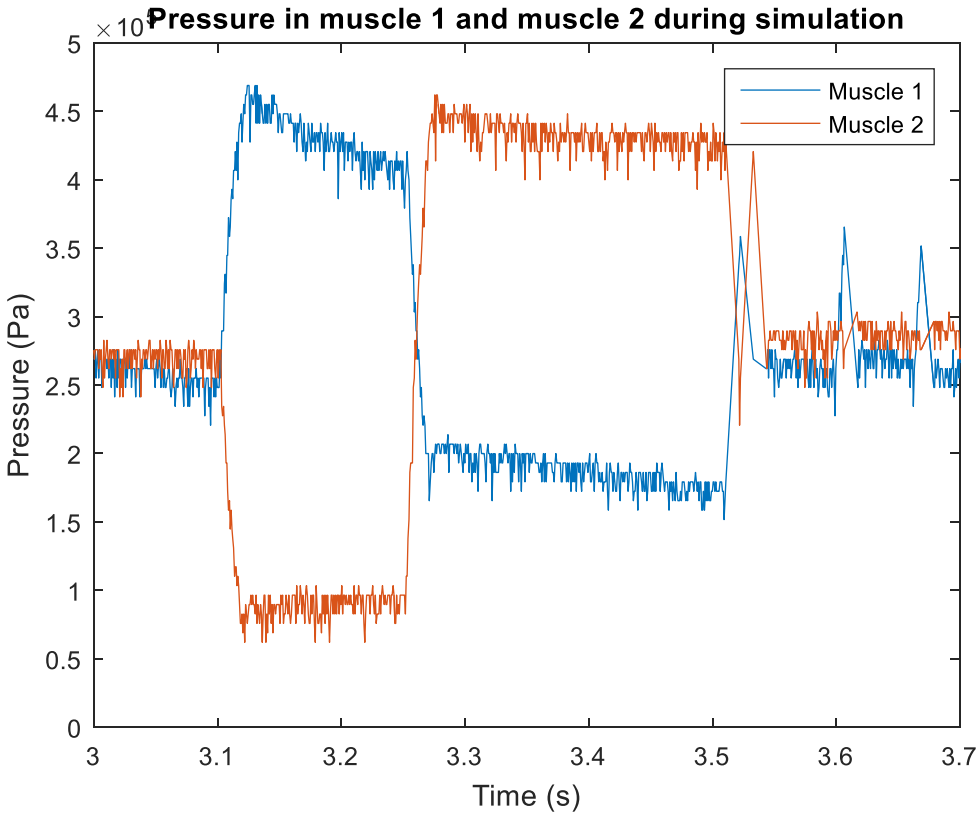


Figure 16: Measured pressure outputs from muscles 1 and 2 during the sequence of desired lengths.

Results

The simulated results generated using Equation (21) are compared to experimental results in Figure 17. The simulation undershoots the experimental system and has a slight ringing before achieving steady state. The experimental data can be seen to have some slight noise originating from the analog reading of the potentiometer.

Figure 18 shows the experimentally reported pressure outputs with the pressure interpolation for both P1 and P2. The pressure is interpolated so that the simulation and experimental sampling rates are identical. The interpolation of input pressure is used to drive the simulation using Equation (21) to produce Figure 17. The output of the experimental system and simulated system are overlaid to determine the system error.

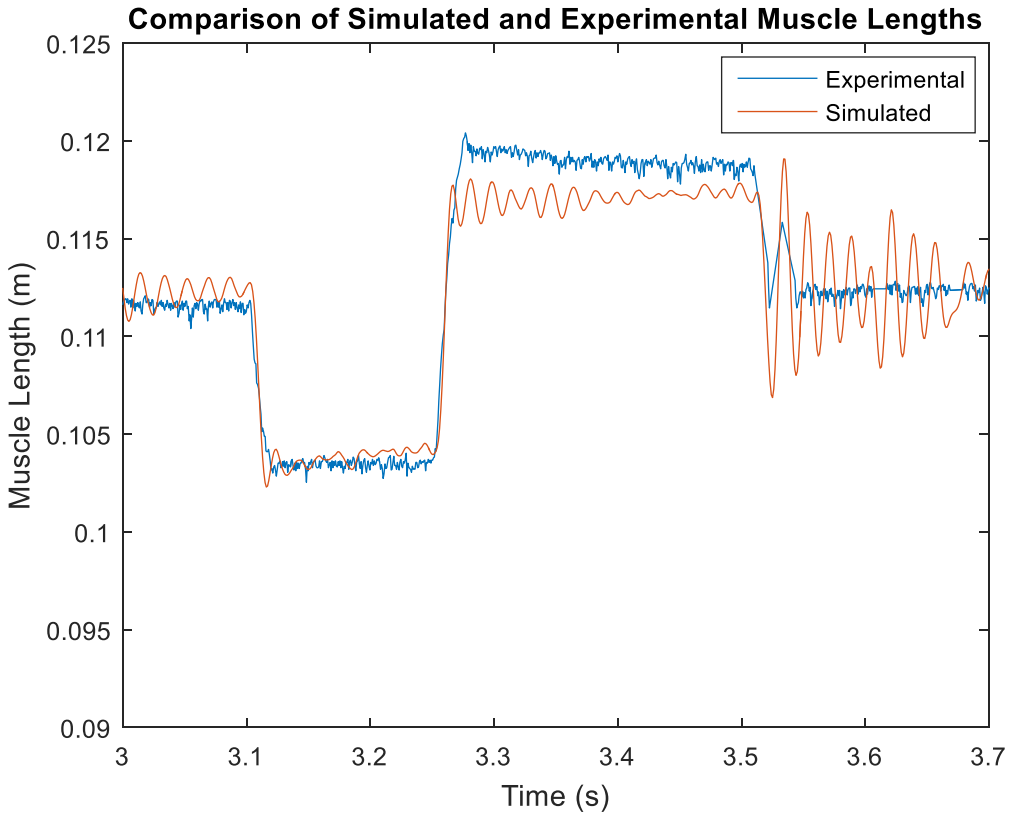


Figure 17: Open loop numeric simulation response overlaid with experimental system response.

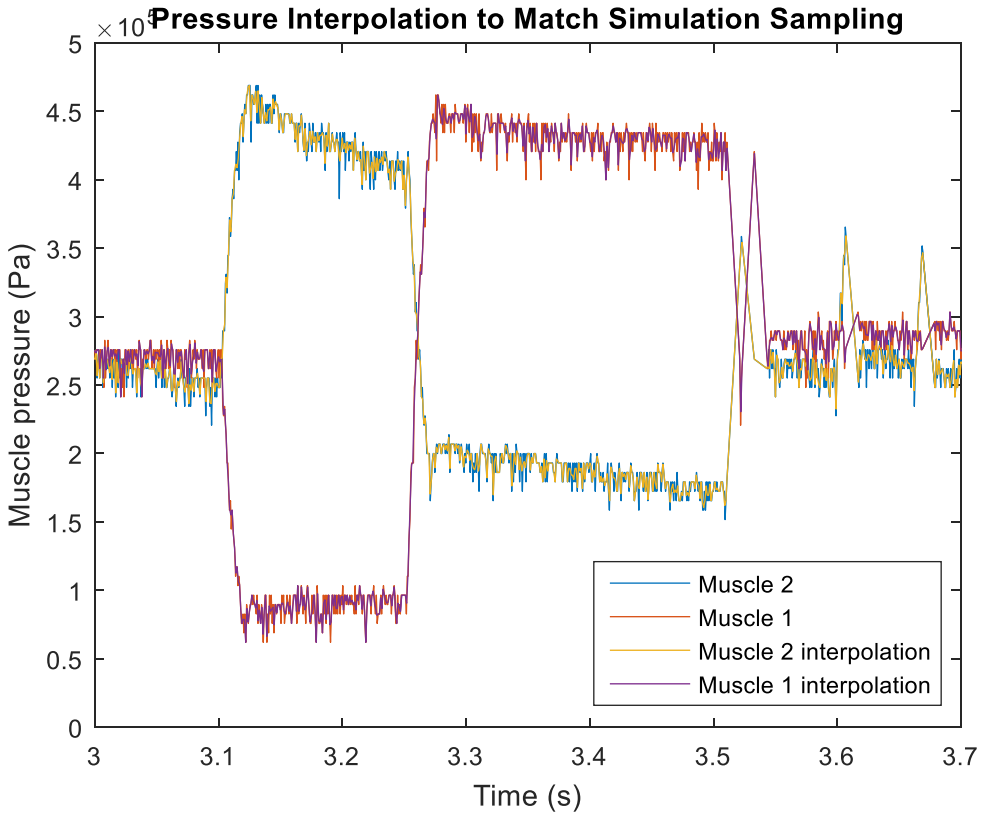


Figure 18: The pressure readings from muscle 1 and muscle 2 from the experimental work are interpolated so that the sampling rate matches the simulation sampling rate.

The differences between the experimental and simulation results are within 5% of the total muscle length. The sampling step size for the model 0.001 seconds.

State Space Model Form

To achieve a state space model system, Equation (21) can be transformed using the assumption of a fixed total system length. The assumption gives

$$L_1 = L_0 - R_p\theta \tag{22}$$

and

$$L_2 = L_0 + R_p\theta \tag{23}$$

where θ is the angle of the pulley and R_p is the pulley radius. With these assumptions, the system only needs one potentiometer to define system position. By combining Equations (21-23) the system can be rewritten as

$$J\ddot{\theta} = R_p\left[\frac{P_1}{4n^2\pi}\left(3\left[L_0 - R_p\theta\right]^2 - b^2\right) - \frac{E_R t(L_0 - R_p\theta)^2}{2n^2\pi}\left(\frac{2n\pi}{\sqrt{b^2 - (L_0 - R_p\theta)^2}} - \frac{1}{R_0}\right)\right.$$
$$\left. + E_R V_b\left(\frac{1}{L_0} - \frac{1}{L_0 - R_p\theta}\right)\right]$$
$$- R_p\left[\frac{P_2}{4n^2\pi}\left(3\left[L_0 + R_p\theta\right]^2 - b^2\right) - \frac{E_r t(L_0 + R_p\theta)^2}{2n^2\pi}\left(\frac{2n\pi}{\sqrt{b^2 - (L_0 + R_p\theta)^2}} - \frac{1}{R_0}\right)\right.$$
$$\left. + E_R V_b\left(\frac{1}{L_0} - \frac{1}{L_0 + R_p\theta}\right)\right] \quad (24)$$

where J is the mass moment of inertia of the actuated arm. J is the physical equivalent of the sum of the M_{eq} terms found in Equation (21). Equation (24) depicts the system in state space form. The utility of the state space form is well documented making further manipulation and control of Equation (24) more straightforward in its implementation.

Development of Controls for Simulation

The proof of controllability is important for nonlinear system models. Many nonlinear models cannot be controlled by a simple controller like a PID controller. Equation (24) is controlled through a feedback loop created around the length error. To create the loop, the length of the muscle is compared to the desired length of the muscle. The difference between the desired and actual length is the error. The error is converted into the duty cycle of a PWM signal for the 4 solenoid valves (2, 3, 4 and 5) of Figure 13. The valves cycle between on and off, creating the choppy line seen in Figure 19. The value of J used is 0.1 Kg*m^2 for this simulation. J is the moment of inertia of the pulley and is estimated for the purpose of simulation.

A PID controller is implemented on the error feedback. The controller modifies the signal passed to the valves to help reduce error. The modification is accomplished by changing the duty cycle that is delivered to the solenoid valves. The larger magnitude of the error, the greater the duty cycle given to the valve. An optimization routine is used on the PID controller gains to find the minimum error. The optimization routine takes a numeric integral of the error and then varies the given parameters, here the PID gains, in order to find the minimum.

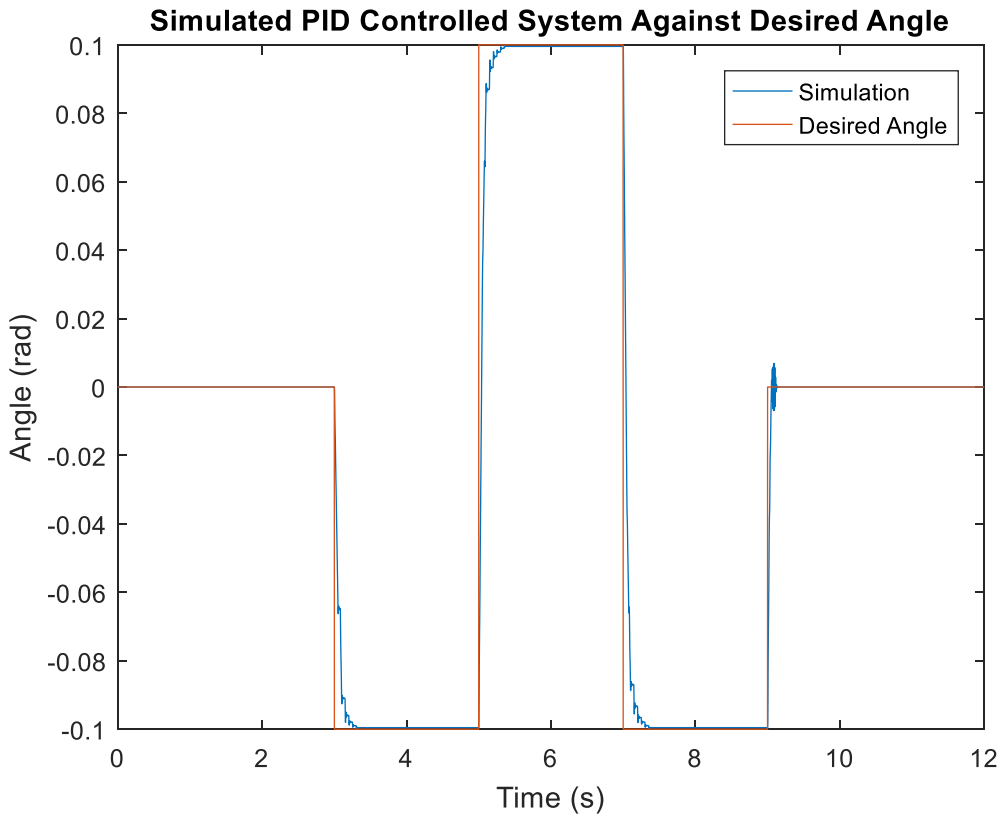


Figure 19: PID controlled model overlaid on the desired response.

The controlled system, shown in Figure 19, has a half second response with no overshoot. The results demonstrate controllability.

Valve estimation

To estimate the valve response, a complex signal mixing is used alongside the PID controllers.

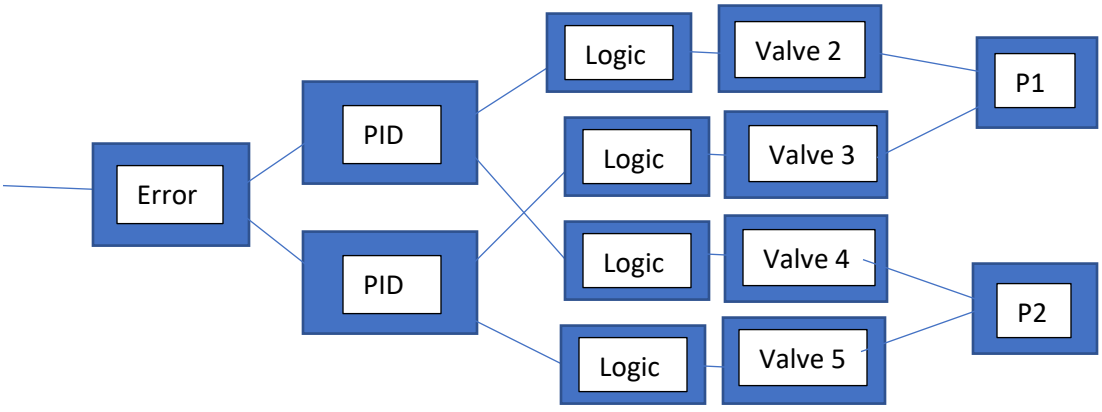


Figure 20: Signal mixing for the PID control system. Angle error is translated into pressure inputs.

Figure 20 shows the signal mixing responsible for translating the angular position error into a pressure signal that the model can use. Two PID controllers are implemented. One PID is used for the valves that increase muscle pressure while the other is used for the valves that decrease muscle pressure. The logic block before each valve determines the sign, positive or negative, of the signal from the PID controller. The sign determines which valve is turned on in the pair controlled by each PID controller. The logic then converts the PID signal into a PWM signal that is sent to the valves for actuation.

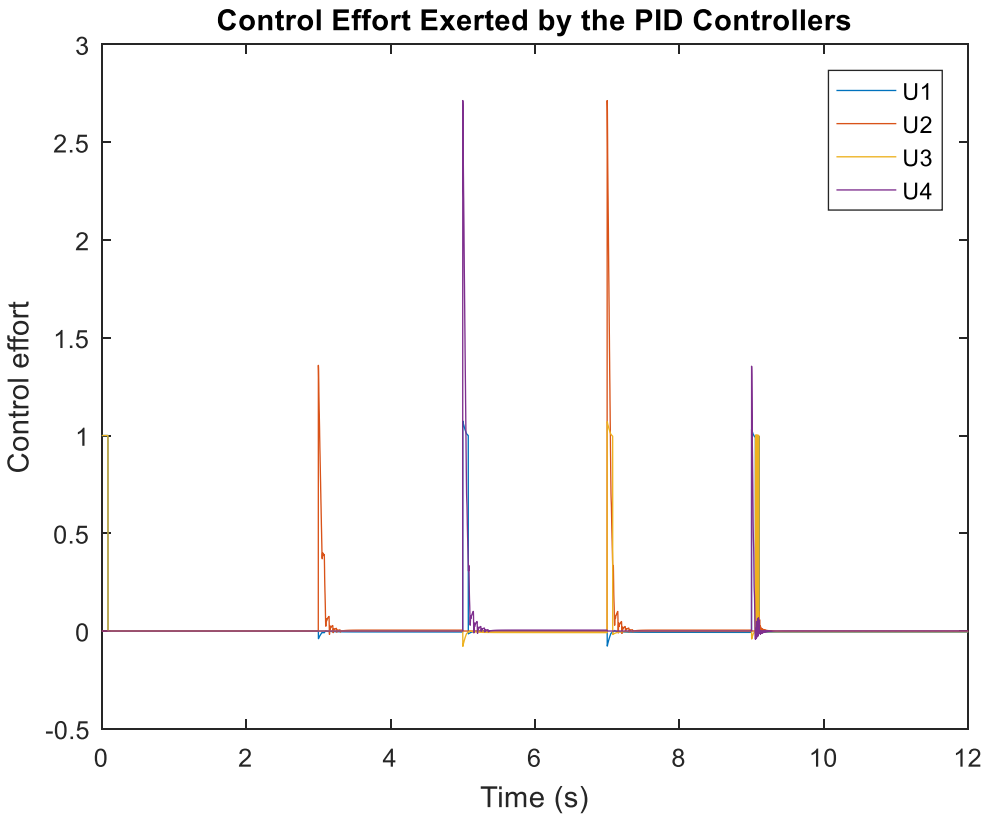


Figure 21: Control effort exerted by PID controllers on valves 1-4.

Figure 21 shows the control efforts, U1-4, that is exerted by the PID controllers that is passed to the logic preceding the valves as seen in Figure 20. The logic converts all signals over the value of one to one and all negative signals to zero.

Discussion

The model developed in Equation (24) is the first dynamic model to be developed for a two muscle system using the physical characteristics of the system. Previous works [9], [18] used experimental based modeling techniques on two muscle systems to develop models which were considered easy to implement and manipulate. These systems are limited, however, by the need to build and test the system before the model coefficients can be determined. Equation (24) is an equally simple model for a two muscle system that shows good agreement with experimental data and incorporates the physical characteristic necessary for scaling and extension to similar muscles.

Extension of the Model

The derived model showed good agreement with experimental results obtained for a muscle of 14 cm length and a 0.5 cm radius as shown in Figure 19. For general use, the model should be able to predict the behavior of similar muscles with different physical properties.

Figure 22 shows the length vs. time plot for a muscle 25 cm in length and with a 0.5 cm radius. The graph shows that a larger displacement is predicted than for a 14 cm muscle in response to the same pressure change. The strain in the muscle, however, is decreased relative to the 14 cm muscle due to the increase in initial muscle volume, which provides no work during actuation. This is expected as the increase in initial volume with the same pressure change should have a smaller strain.

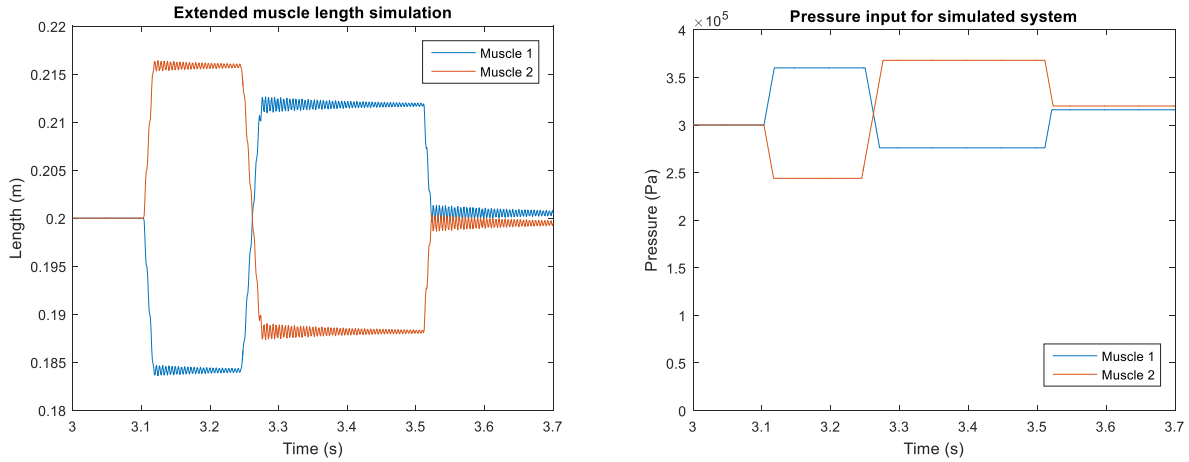


Figure 22: Simulated changed in muscle length an increased initial muscle length (Left) and the pressure input to the simulation model (right). –

Figure 23 shows the results of an increase in initial muscle radius from 0.5 cm to 2 cm in radius with a 10 cm length. This is a large change in the radius parameter, but, for the same pressure input, the length output from the model decreases. The decrease in length is expected since the increase in internal volume decreases the amount of pressure that provides useful work in the muscle.

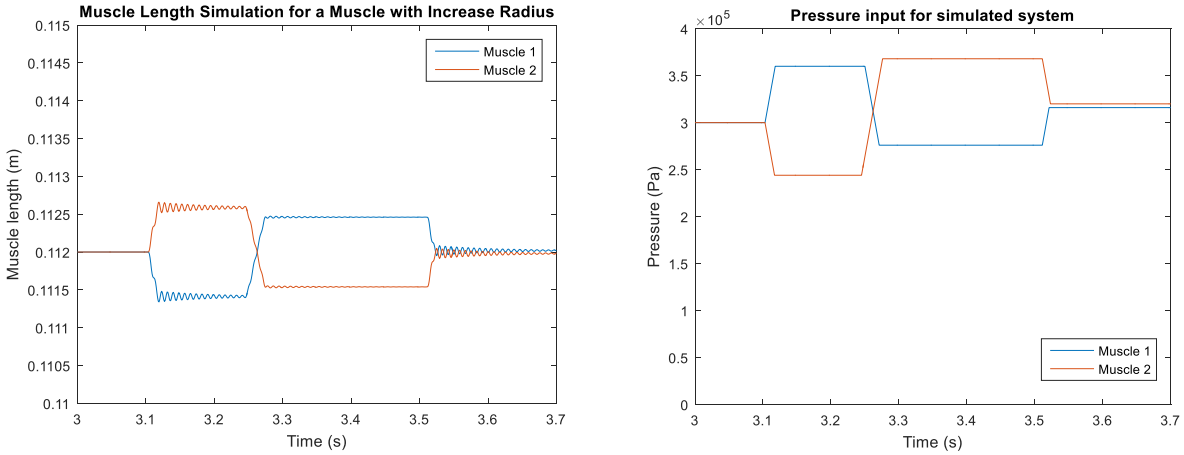


Figure 23: Simulated change in muscle length with an increased initial muscle radius (left) and the pressure input to the simulation model (right).

Figure 24 shows the predicted change in length for a simulated muscle with an alternate bladder material which has a higher elastic modulus. For this simulated muscle, the elastic modulus is increased from 0.001 GPa to 0.008 GPa. As expected, the increase in bladder stiffness causes a decrease in output displacement when stimulated with the same pressure input.

Figures 22-24 illustrate how having a model based on the physical characteristics of the system can be used to predict responses to a range of system configurations. Increasing the muscle length, radius, or the Young's modulus of the bladder material for the same pressure inputs decreases the amount of useful work being done on the system. The overall changes in length were less than for the 14 cm muscle as expected, thus these tests confirm that the model is useful for prediction.

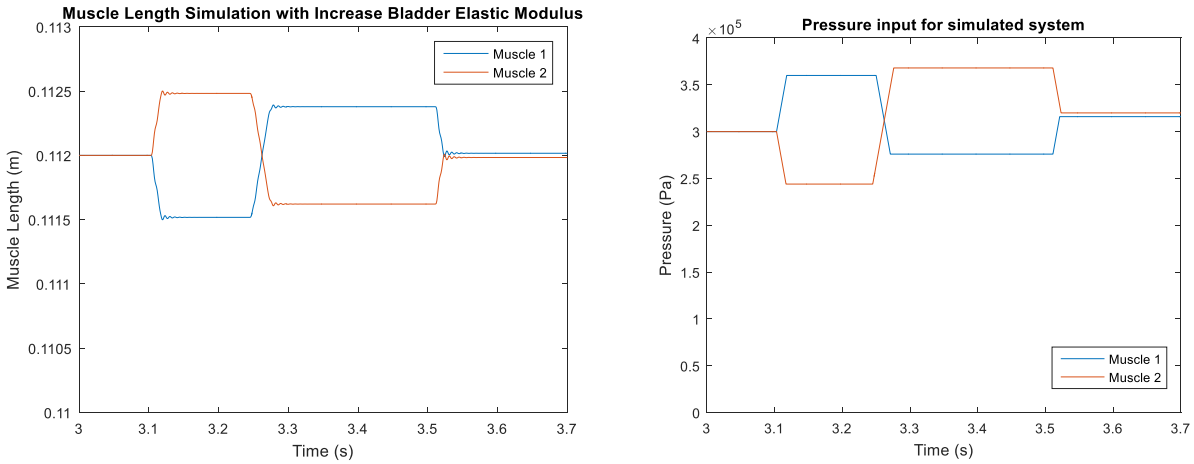


Figure 24: Simulated change in muscle using an increased Young's modulus for the bladder material (left) and the pressure input to the simulated muscle (right).

Controlling the Muscle System

Implementation of a simple control algorithm was done as a proof of concept to demonstrate that the muscle system could be controlled through a simple control system like PID. The results showed a basic level of control could be achieved, however, the controls work was based around the use of solenoid valves. The use of solenoid valves is not optimal for precise control of the two muscle system, but these were readily available in the lab. Using proportional valves in the future would allow for gradual changes in flow rate through the valves, producing smoother outputs to the muscles. Using gradual changes would allow for more accurate control without the choppy response, seen in Figure 19, that the solenoid valves produce. Further work on control of the muscle system should consider the use of proportional valves.

A goal of this thesis was to implement a simple control algorithm to prove controllability. The PID controller used is a simple control method that was chosen to make the code easy to implement on a real system. While another control method may give a faster response, additional complexity in the controller could slow down the simulation times. Future work could focus on investigating the tradeoffs between simulation times and the speed of the controller response.

Strengths and Weaknesses of the New System Model

The new model is effective at predicting the motion of the two muscle system. The error for the steady state response of the system is small, while using a model containing few terms. These abilities fulfil the requirements for the desired model for a two muscle system.

The open loop form of this model has a larger error in length than a well developed experimental model will likely have since it is based on the specific response of the muscle. In addition, the model does not account for hysteresis in the muscle, which would lead to the eventual reduction of angular and torque outputs from the muscle. Future work should consider the addition of a damping effect and the implementation of the model onto a microcontroller.

Future work

The model developed here is meant to act as a safety check for an error based controller on an autonomous system. Through comparison with the developed model, inconsistencies with the sensor readings can be discovered and fixed before catastrophic failure. The developed model needs to give meaningful data when run with different sizes and material constraints on the system, to be able to assist in construction requirements of the muscles.

While damping present in Equation (24) brings the model closer to actual system response, it may not be enough to represent the physical system. To reduce error further, additional manipulation of the terms may be required. Future work may consider using a complex term for the elastic modulus, simulating viscoelastic damping, to further reduce error.

Conclusions

After examining many models derived for a single McKibben style muscle, a simple model was selected which showed good agreement with muscle behavior. In order to describe a symmetric opposed muscle pair, two muscle models were linked together. To be able to model transient muscle response, valve dynamics were added. This inclusion is new for models that are based on physical properties of the system. The consideration of complications due to valves has only been studied in models using experimentally determined coefficients to date.

Comparison to experimental results showed that the newly developed model is effective at predicting the response of a two muscle system. Using this model allows for the simulation of a proposed system before muscles have been built. These benefits make the new model developed here a valuable addition to the family of models that currently exist.

Demonstrating the controllability of the new muscle model is important for its continued use. Proving controllability confirms that the muscle is well defined and converges. The proof was done in simulation with a PID controller implemented on the angle error of the system model.

References

[1] Tondu, Bertrand. Lopez, Pierre "The McKibben muscle and its use in actuating robot-arms showing similarities with human arm behavior." Industrial Robot Volume 24 pp. 432-439 Number 6 1997

[2] Klute, Glenn K., Joseph M. Czerniecki, and Blake Hannaford. "McKibben Artificial Muscles: Pneumatic Actuators with Biomechanical Intelligence." IEEE/ASME International Conference on Advanced Intelligent Mechatronics (1999) 221-226.

[3] Dzahir, Mohd, Azuwan Mat, and Shin-ichiroh Yamamoto. "Recent Trends in Lower-Limb Robotic Rehabilitaton Orthosis: Control Scheme and Strategy for Pneumatic Muscle Actuated Gait Trainers." Robotics (2014) 120-148.

[4] Caldwell, Darwin G., N. Tsagarakis, and G.A. Medorano-Cerda. "Bio-mimetic actuators: polymeric Pseudo Muscular Actuators and pneumatic Muscle Actuators for biological emulation." Mechatronics 10 (2000) 499-530.

[5] Davis, Steve, N. Tsagarakis, J. Canderle, and Darwinn G. Caldwell. "Enhanced Modelling and Performance in Braided Pneumatic Muscle Actuators." The International Journal of Robotics Research Vol 22. (2003) 213-227.

[6] Chan, S. W., Lilly, John H. Repperger, Daniel W., Berlin, James E. "Fuzzy PD+I Learning Control for a Pneumatic Muscle." The IEEE Conference on Fuzzy Systems (2003) 278-283

[7] Kothera, Curt S., Mamta Jangid, Jayant Sirohi, and Norman M. Wereley. "Experimental Characterization and Static Modeling of McKibben Actuators." Journal of Mechanical Design Vol. 131 (2009)

[8] Davis, S., J. Canderle, P. Artrit, N. Tsagarakis, and Darwin G. Caldwell. "Enhanced Dynamic Performance in Pneumatic Muscle Actuators." IEEE International Conference on Robotics & Automation (2002) 2836-2841.

[9] Schroder, Joachim, Duygun Erol, Kazuhiko Kawamura, Ph.D., and Rudiger Dillmann, Dr.-Ing. "Dynamic Pneumatic Actuator Model for a Model-Based Torque Controller." IEEE International Symposium on Computational Intelligence in Robotics and Automation (2003) 342-347.

[10] Reynolds, D. B., D. W. Repperger, C. A. Phillips, and G. Bandry. "Modeling the Dynamic Characteristics of Pneumatic Muscle." Annals of Biomedical Engineering Vol. 31 (2003): 310-317.

[11] Bertetto, A. Manuello, and M. Ruggiu. "Characterization and modeling of air muscles." Mechanics Research Communications 31 (2004) 185-194.

[12] Tondu, Bertrand, and Pierre Lopez. "Modeling and Control of McKibben Artificial Muscle Robot Actuators." IEEE Control Systems Magazine (April 2000) 15-38.

[13] Zhang, Zhiye, Michael Philen. "Pressurized artificial muscles." Journal of Intelligent Material Systems and Structures 23 (2011) 255-268.

[14] Parandyk, Wiktor, Michal Ludwicki, Bartlomeij Zagrodny, and Jan Awrejcewicz. “The Positioning of Systems Powered by McKibben Type Muscles.” Mechatronics: Ideas for Industrial Applications, Advances in Intelligent Systems and Computing. 317 (2015) 133-140.

[15] Chou, Ching-Ping, and Blake Hannaford. “Measurement and Modeling of McKibben Pneumatic Artificial Muscles.” IEEE Transactions on Robotics and Automation, Vol 12. (1996) 90-102.

[16] Caldwell, Darwin G., Gustavo A. Medrano-Cerda, and Mike Goodwin. “Control of Pneumatic Muscle Actuators.” IEEE Control Systems (1995) 40-48.

[17] Phatak, Aniruddha Sudhir. “Relationship Between Velocity of Contraction and Force Applied on Air Muscles.” Thesis/Dissertation Collections at RIT Scholar Works

[18] Tondu, B., S. Ippolito, J. Guiochet, and A. Daidie. “A Seven-degrees-of-freedom Robot-arm Driven by Pneumatic Artificial Muscles for Humanoid Robots.” The International Journal of Robotics Research Vol. 24. (2005) 257-274.

[19] Robinson, Tyan M., Curt S. Kothera, Benjamin K. S. Woods, Robert D. Vocke III, and Norman M. Wereley. “High Specific Power Actuators for Robotic Manipulators.” Journal of Intelligent Material Systems and Structures Vol 22 (2011) 1501-1511.

Appendixes

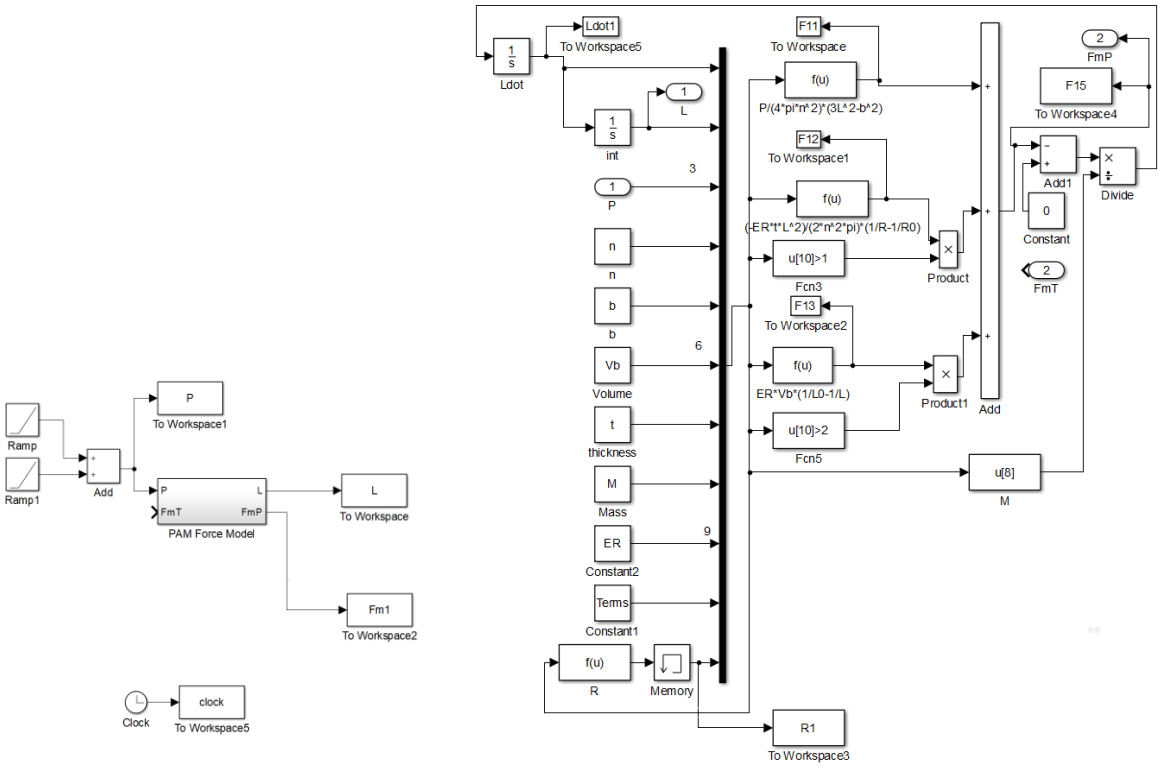
Appendix A: MATLAB Code

```
%% Tgraphsm.m

%Muscle constant

L0=0.5;

R0=.005;
t=.0015;
Vb=R0*R0*pi*L0;
ER=1e6;

Pddot=10e4;
b=L0/cosd(30);
n=b*sind(30)/(2*R0*pi);

Rp=.04;
%Simulation constant

Rair=287.058;
M=1;
Patm=101325;
tf=20;
Ldot0=0;
T=297;

Vtank=pi*.1*.1*.3;
P_Hi0=3e5;

Cd=1;

%% Simulation

Terms=1;
sim('Tgraphs')

%% store terms

L1=L;
P1=P;
Fm11=Fm1;

%% Simulation

Terms=3;
sim('Tgraphs')

%% store terms
```

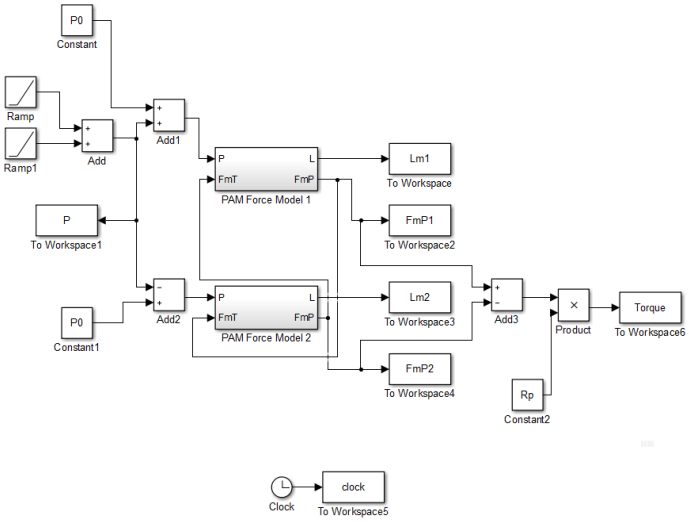
```
L3=L;
P3=P;
Fm13=Fm1;

%% plots

figure(1)
plot(tout,L1,tout,L3)

figure(2)
plot(tout,P1,tout,P3)

figure(3)
plot(tout,Fm11,tout,Fm13)

%% Torque time
L01=L0;
L0=L01*.9;
F=20;
P0=F*4*n*n*pi/(3*L0*L0-b*b);

%% Simulation

Terms=1;
sim('TgraphsT')

%% Term storage

Lm11=Lm1;
Lm21=Lm2;
P1=P;
FmP11=FmP1;
FmP21=FmP2;
Torque1=Torque;

%% Simulation

Terms=3;
sim('TgraphsT')

%% Term Storage

Lm13=Lm1;
Lm23=Lm2;
P3=P;
FmP13=FmP1;
FmP23=FmP2;
Torque3=Torque;

%% Plots

figure(21)
plot(tout,Lm11,tout,Lm21)
```

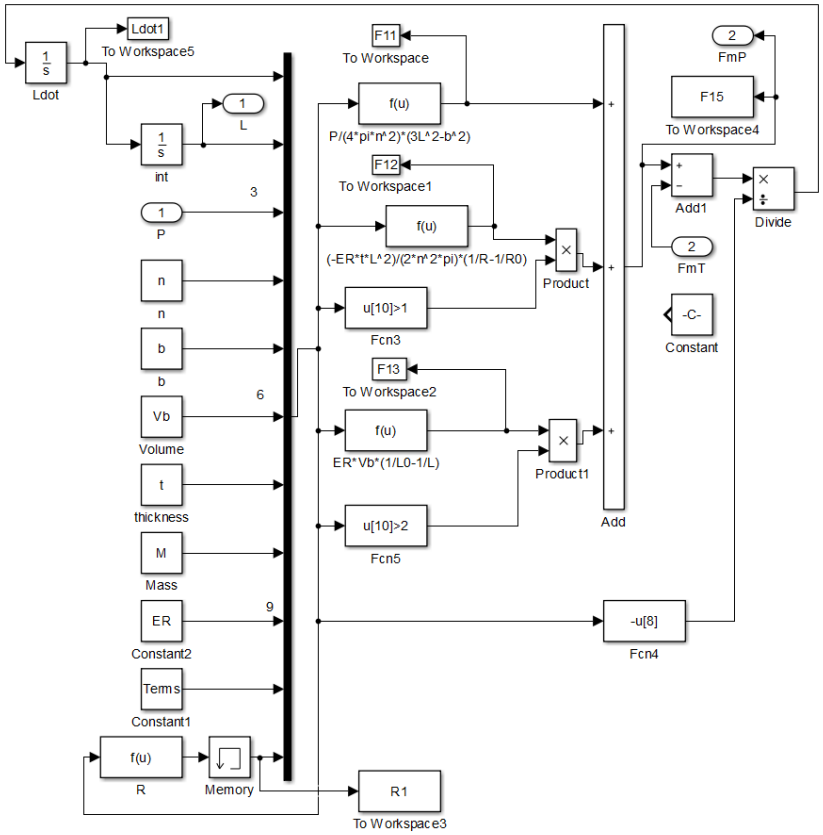
```
figure(23)
plot(tout,Lm13,tout,Lm23)

figure(24)
plot(tout,FmP11,tout,FmP21)

figure(26)
plot(tout,FmP13,tout,FmP23)

figure(27)
plot(tout,P0+P,tout,P0-P)

figure(28)
plot(tout,Torque1,tout,Torque3)
```

Appendix B: Simulink Tgraphs.slx



Appendix C: Simulink TgraphsT.slx





Appendix D: Arduino Code

```
/*
 Example sketch for the Xbox Wireless Reciver library - developed by Kristian Lauszus
 It supports up to four controllers wirelessly
 For more information see the blog post: http://blog.tkjelectronics.dk/2012/12/xbox-360-receiver-
added-to-the-usb-host-library/
 This open source sketch was used for the Xbox receiver connection
 */

#include <XBOXRECV.h>
#include "definitions.h"

USB Usb;
XBOXRECV Xbox(&Usb);

void setup() {
  Serial.begin(115200);

  if (Usb.Init() == -1) {
    Serial.print(("\r\nOSC did not start"));
    while (1); //halt
  }
  Serial.print(("\r\nXbox Wireless Receiver Library Started"));
  pins();
}

void loop() {
 //Check to see if the battery on the robot is within its appropriate range, if not it exits the while loop
 while (digitalRead(batteryLevelHealth) == 1){
    Usb.Task();
    //confirms Xbox remote control connection
    if (Xbox.XboxReceiverConnected) {
      //Keeps V1 on at all times unless deflating the bones and muscles
      //when all the air from the tank is also released to atmosphere
      digitalWrite(V1, turnOn);
      Serial.println("");
      pressureSensor();
      compressor();
      digitalWrite(V2, turnOn);
      walkingStart();
      inflateMuscleResevoir();
      deflateMuscleResevoir();

      //checks for button presses
      for (uint8_t i = 0; i < 4; i++) {
        if (Xbox.Xbox360Connected[i]) {
```

```
        if (Xbox.getButtonClick(B, i)){
          Serial.println(F("B"));
          //for walking() function
          walkingCommand = !walkingCommand;
          //Serial.println(millis());
        }

        if (Xbox.getButtonClick(X, i)){
          //Serial.println(F("X"));
          //for compressor() function
          compressorCommand = !compressorCommand;
          //Serial.print("Compressor switched state");
        }
      }

      //report the battery level of the Xbox controller
      if (Xbox.getButtonClick(XBOX, i)) {
        Xbox.setLedMode(ROTATING, i);
        Serial.print(F("Xbox (Battery: "));
        Serial.print(Xbox.getBatteryLevel(i)); // The battery level of the Xbox controller in the range 0-3
        Serial.println(F(")"));
      }
    }
  }
}
//Turn off everything except for arduino when the battery is dead
digitalWrite(compressorPin, turnOff);
compressorCommand = false;
 walkingCommand = false;
 digitalWrite(V1, turnOff);
 digitalWrite(V2, turnOff);
 digitalWrite(V3, turnOff);
 digitalWrite(V4, turnOff);
 digitalWrite(V5, turnOff);
 digitalWrite(V6, turnOff);
 digitalWrite(V7, turnOff);
 digitalWrite(V8, turnOff);
 digitalWrite(V9, turnOff);
 digitalWrite(V10, turnOff);
 digitalWrite(V11, turnOff);
 digitalWrite(V12, turnOff);
 digitalWrite(V13, turnOff);
 digitalWrite(V14, turnOff);
 digitalWrite(V15, turnOff);
 //turn off everything for the maximum delay allowed
 delay(32767);
}
```

```
void compressor(){
  if (compressorCommand == true){
    //convert the output to PSI to read the pressure of air tank
    airTankPressure = ((analogRead(airTankPressureSensor)*0.0049)-0.5)/0.04;
    if (airTankPressure < 60){
      digitalWrite(compressorPin, turnOn);
      }
    else if (airTankPressure > 80){
      digitalWrite(compressorPin, turnOff);
    }
  }
  else if (compressorCommand == false){
    digitalWrite(compressorPin, turnOff);
  }
}



//The purpose of this pressure function is to constantly measure the
//air pressure of the air tank, the muscles airflow, and the bone airflow

void pressureSensor(){
  //Measure air tank pressre and convert to PSI using formula from datasheet
  Serial.print("Air tank (PSI) = \t");
  airTankPressure = ((analogRead(airTankPressureSensor)*0.0049)-0.5)/0.04;
  Serial.print(airTankPressure);
  Serial.print("\t");

  //Measure muscle pressre and convert to PSI using formula from datasheet
  Serial.print("Muscle (PSI) =\t");
  musclePressure = ((analogRead(musclePressureSensor)*0.0049)-0.5)/0.04;
  Serial.print(musclePressure);
  Serial.print("\t");

  //Measure bone pressre and convert to PSI using formula from datasheet
  Serial.print("Bone (PSI) =\t");
  bonePressure = ((analogRead(bonePressureSensor)*0.0049)-0.5)/0.04;
  Serial.print(bonePressure);
  Serial.print("\t");

  //Print potentiometer reading
  Serial.print("Mus Length =\t");
  musLength=analogRead(musLenghtPot);
  Serial.print(musLength);
  Serial.print("\t");
}
```

```
void walkingStart(){
  if (walkingCommand == true ){
    if(startWalking == false){
      //previousMillis records the time at which the startWalking if loop starts and starts the counter
      //it also starts the walkingCounter from zero
      previousMillis = millis();
      walkingCounter = 0;
    }
    //If startWalking stays true, keep the previously set previousMillis and walkingCounter
    startWalking = true;
  }
  else if (walkingCommand == false){
    startWalking = false;
      //Turing off this set of solenoids first releases all the air from the muscles
      digitalWrite(V10, turnOff);
      digitalWrite(V11, turnOff);
      digitalWrite(V12, turnOff);
      digitalWrite(V13, turnOff);
      digitalWrite(V14, turnOff);
      digitalWrite(V15, turnOff);

      digitalWrite(V4, turnOff);
      digitalWrite(V5, turnOff);
      digitalWrite(V6, turnOff);
      digitalWrite(V7, turnOff);
      digitalWrite(V8, turnOff);
      digitalWrite(V9, turnOff);
  }
Serial.print(millis()-previousMillis);
Serial.print("\t");

  if (startWalking){
       // initial pressurization
       if (millis() - previousMillis > 100 + walkingCounter && millis() - previousMillis < 2000 +
walkingCounter && musclePressure < 35) {
         digitalWrite(V10, turnOn); // to muscle
         digitalWrite(V8, turnOn); // to valve
         Serial.print("inflate");
       }

       if (millis() - previousMillis > 100 + walkingCounter && millis() - previousMillis < 2000 +
walkingCounter && bonePressure < 35) {
         digitalWrite(V11, turnOn); // to muscle
         digitalWrite(V9, turnOn); // to valve
         //Serial.println(millis()-previousMillis);
         Serial.print("Inflate muscles");
       }
```

```
      // Hold muscle inflation
      if (millis() - previousMillis > 100 + walkingCounter && millis() - previousMillis < 2000 +
walkingCounter && musclePressure > 50){
        digitalWrite(V8, turnOff);
        digitalWrite(V10, turnOff);
        Serial.print("Deflate");
        delay(100);
        digitalWrite(V10, turnOn);
      }

      if (millis() - previousMillis > 100 + walkingCounter && millis() - previousMillis < 2000 +
walkingCounter && bonePressure > 50){
        digitalWrite(V9, turnOff);
        digitalWrite(V11, turnOff);
        //Serial.println(millis()-previousMillis);
        Serial.print("Deflate muscles");
        delay(100);
        digitalWrite(V11, turnOn);
      }

      //Right
      if (millis() - previousMillis > 3000 + walkingCounter && millis() - previousMillis < 4000 +
walkingCounter && musLength > 700){
        digitalWrite(V11, turnOff);
        digitalWrite(V8, turnOn);
        //Serial.println(millis()-previousMillis);
        Serial.print("Right");
      }

      //Hold
      if (millis() - previousMillis > 3200 + walkingCounter && millis() - previousMillis < 4500 +
walkingCounter){
        digitalWrite(V8, turnOff);
        digitalWrite(V11, turnOn);
        //Serial.println(millis()-previousMillis);
        Serial.print("Hold");
      }

      //Left
      if (millis() - previousMillis > 4500 + walkingCounter && millis() - previousMillis < 6500 +
walkingCounter && musLength < 900) {
        digitalWrite(V10, turnOff);
        digitalWrite(V9, turnOn);
        //Serial.println(millis()-previousMillis);
        Serial.print("Left");
      }

      //Hold
```

```
      if (millis() - previousMillis > 4700 + walkingCounter && millis() - previousMillis < 7000 +
walkingCounter){
        digitalWrite(V9, turnOff);
        digitalWrite(V10, turnOn);
        //Serial.println(millis()-previousMillis);
        Serial.print("Hold");
      }

      //Deflate S

      //counter allows the previous two sequence to continue until stopped by a button press
      if (millis() - previousMillis > 7000 + walkingCounter){
        walkingCounter = walkingCounter + 7000;
      }
  }
}
```

```
#define turnOn HIGH
#define turnOff LOW

//pins that work for digital: 22, 24, 26, 28, 30, 31, 32, 33, 34, 35, 36, 37, 40, 42, 44, 45, 46, 47, 48, 49
#define V1 22 //Solenoid valve 1
#define V2 24 //Solenoid valve 2
#define V3 26 //Solenoid valve 3
#define V4 28 //Solenoid valve 4
#define V5 30 //Solenoid valve 5
#define V6 31 //Solenoid valve 6
#define V7 32 //Solenoid valve 7
#define V8 33 //Solenoid valve 8
#define V9 34 //Solenoid valve 9
#define V10 35 //Solenoid valve 10 35
#define V11 36 //Solenoid valve 11 36
#define V12 37 //Solenoid valve 12 37
#define V13 40 //Solenoid valve 13 40
#define V14 42 //Solenoid valve 14 42
#define V15 44 //Solenoid valve 15 44

#define compressorPin 45 //Compressor
#define batteryLevelHealth 46 //Digital read of whether battery is in operating levels

#define musLenghtPot A13 //Length of muscle
#define airTankPressureSensor A12 //Pressure input of air tank A12
#define musclePressureSensor A11 //Pressure input of air to muscle A11
#define bonePressureSensor A10 //Pressure input of air to bone A10

int airTankPressure = 0; //PSI value of air tank
int musclePressure = 0;//PSI value of muscles
int bonePressure = 0;//PSI value of bones
int musLength = 0;//Pot muscle length
int batteryLevelValue = 0;

boolean compressorCommand = false;
boolean inflateBonesCommand = false;
boolean deflateBonesCommand = false;
boolean bendMuscleCommand = false;
boolean deflateMuscleCommand = false;
boolean startWalking = false;
boolean walkingCommand = false;
boolean musclePressureInRange = false;
boolean inflateMuscleResevoirCommand = false;
boolean deflateMuscleResevoirCommand = false;
boolean standCommand = false;

unsigned long previousMillis = millis();
unsigned long walkingCounter = 0;
```

Appendix E: Table of simulation values

Table 3: Physical properties used for numeric simulation

Physical parameter	Value
L0	0.1 m
R0	0.005 m
Meq	0.01 Kg
ER	0.001e9 GPa
t	0.0015 m
P0	2.76e5 Pa