MODELING CELLULAR ACTUATOR ARRAYS

A Thesis
Presented to
The Academic Faculty

by

David L. MacNair

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
The George Woodruff School of Mechanical Engineering

Georgia Institute of Technology
December 2013

MODELING CELLULAR ACTUATOR ARRAYS

Approved by:

Dr. Jun Ueda, Advisor
Georgia W Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Kok-Meng Lee
Georgia W Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Harvey Lipkin
Georgia W Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Mike Stilman
School of Interactive Computing
Georgia Institute of Technology

Dr. Jeannette Yen
School of Biology
Georgia Institute of Technology

Date Approved: August 2013

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. ECCS-0932208

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

I^n	Force carried across Hill-Type Model cell n for the examples used in the Modeling chapter.
M	Total number of masses in an array.
N	Total number of cells in an array.
Q	Total number of internal variables.
V	Total array control inputs in an array.
$\mathbf{\Phi}$	Mass matrix. Diagonal matrix containing the inverse of ϕ_m (mass of mass m) for each entry (m, m).
$\mathbf{\Theta}$	Array control inputs.
$\boldsymbol{\alpha}$	Effects of mass locations on array masses and endpoint forces.
$\boldsymbol{\beta}$	Effects of mass velocities on array masses and endpoint forces.
$\boldsymbol{\delta}$	Effects of array control inputs on array masses and endpoint forces.
$\boldsymbol{\eta}$	Effects of mass velocities on internal cell states.
$\boldsymbol{\gamma}$	Effects of cell internal states on array masses and endpoint forces.
$\boldsymbol{\iota}$	Effects of cell internal states on internal cell states.
$\boldsymbol{\kappa}$	Effects of array control inputs on internal cell states.
$\mathbf{B}_{\mathbf{\Theta}}$	Array control inputs component of the input matrix.
$\mathbf{D}_{\mathbf{\Theta}}$	Array control inputs component of the feed-forward matrix.
$\boldsymbol{\mu}$	Column vector of length N with 1's in every element.
$\boldsymbol{\zeta}$	Effects of mass locations on internal cell states.
$\dot{p}_L^n$	Velocity of the left endpoint of cell n.

$\dot{p}_R^n$	Velocity of the right endpoint of cell n.
$\dot{p}_m$	Velocity of mass m.
$\dot{z}_L$	Left array endpoint velocity.
$\dot{z}_R$	Right array endpoint velocity.
ℓ_L^m	List of cells connected to the left of mass m.
ℓ_R^m	List of cells connected to the right of mass m.
$\mathbf{A}$	System matrix.
$\mathbf{B_Z}$	Array endpoint positions and velocities component of the input matrix.
$\mathbf{B}$	Input matrix.
$\mathbf{C}$	Output matrix.
$\mathbf{D_Z}$	Array endpoint positions and velocities component of the feed-forward matrix.
$\mathbf{D}$	Feed-forward matrix.
$\mathbf{F}$	Array Endpoint Forces.
$\mathbf{J}$	Internal state variables for all of the cells in an array.
$\mathbf{J}_q$	Internal cell states for internal variable number q. There is one entry for ever cell per internal variable number.
$\mathbf{P_v}$	Positions and velocities of the $M-2$ masses [p_2 ... $p_{(M-1)}$] internal to an array.
$\mathbf{P}$	Positions of the $M-2$ masses [p_2 ... $p_{(M-1)}$] internal to an array.
$\mathbf{U}$	Input vector.
$\mathbf{X}$	State vector.
$\mathbf{Z}$	Array endpoint positions.
$\dot{\mathbf{P}}$	Velocities of the $M-2$ masses [p_2 ... $p_{(M-1)}$] internal to an array.

$\dot{\mathbf{Z}}$	Array endpoint velocities.
ω_1^n	Internal variable 1 of cell n. In Hill-Type Model cells, this is the length of cell n's contractile element.
ω_q^n	Internal variable q of cell n.
ϕ_m	Mass of mass m. ϕ_1 and ϕ_M equal 1 since they are used in determining array endpoint force and not mass acceleration changes.
$\text{coe}(y)_L$	Left side force coefficient matrix function for variable y.
$\text{coe}(y)_R$	Right side force coefficient matrix function for variable y.
$\text{cow}(y)_q$	Internal variable q coefficient matrix function for variable y.
θ^v	Control input number v. In Hill-Type Model cells, this is the pure force due to activating cell n where $n = v$.
b_{de}	Damping coefficient in Hill-Type Model cells.
k_{ae}	Actuator spring constant in Hill-Type Model cells.
k_{pe}	Parallel elastic spring constant in Hill-Type Model cells.
k_{se}	Series elastic spring constant in Hill-Type Model cells.
m	Mass number index.
n	Cell number index.
p_L^n	Position of the left endpoint of cell n.
p_R^n	Position of the right endpoint of cell n.
p_m	Position of mass m.
q	Internal variable number index.
v	Array control input number index.
z_L	Left array endpoint position.
z_R	Right array endpoint position.

SUMMARY

This work explores the representations and mathematical modeling of biologically-inspired robotic muscles called Cellular Actuator Arrays. These actuator arrays are made of many small interconnected actuation units which work together to provide force, displacement, robustness and other properties beyond the original actuator's capability. The arrays can also exhibit properties generally associated with biological muscle and can thus provide test bed for research into the interrelated nature of the nervous system and muscles, kinematics/dynamics experiments to understand balance and synergies, and building full-strength, safe muscles for prosthetics, rehabilitation, human force amplification, and humanoid robotics.

This thesis focuses on the mathematical tools needed bridge the gap between the conceptual idea of the cellular actuator array and the engineering design processes needed to build physical robotic muscles. The work explores the representation and notation needed to express complex actuator array topologies, the mathematical modeling needed to represent the complex dynamics of the arrays, and properties to guide the selection of arrays for engineering purposes. The approach is designed to aid automation and simulation of actuator arrays and provide an intuitive base for future controls and physiology work. The work is validated through numerical results using MatLab's SimMechanics dynamic modeling system and with three physical actuator arrays built using solenoids and shape memory alloy actuators.

- Chapter 1 will present the motivation behind using actuator arrays, review existing literature related to actuator arrays benefits and modeling, and clarify the goals of the research.
- Chapter 2 will provide methods for representing, automatically generating, and determining the similarity between complex array topologies.
- Chapter 3 will develop dynamic modeling techniques needed to simulate and control actuator arrays related network systems.
- Chapter 4 will demonstrate how outside dynamics can be applied to model complex systems

including outside loads, multiple arrays, and hierarchical arrays.

- Chapter 5 will show static properties, methods for controlling redundant actuation usings, and non-linear extensions to the actuator array modeling.
- Chapter 6 will provide experimental results validating the actuator array modeling techniques presented in this thesis.
- Chapter 7 will summarize the results and provide future directions for the research.

CHAPTER I

INTRODUCTION AND LITERATURE SURVEY

1.1 Overview

This thesis studies the representation and mathematical modeling of biologically-inspired robotic muscles called actuator arrays (Fig. 1). Actuator arrays are made of many small interconnected actuation units (cells) which are combined to gain greater force, displacement, robustness, and other properties the original actuators could not achieve. The cells can be built from many traditional actuators like motors, hydraulics, pneumatics, etc. and the resulting arrays can gain many of the properties associated with biological muscle like compliance and robustness.

This work focuses on bridging the gap between the conceptual idea of the cellular actuator array and the engineering design process needed to select and build physical robotic muscles. The work explores the representation and notation needed to express complex cellular actuator array topologies, the requirements for cells used in arrays, the mathematics and graph theory for finding all possible array topologies given particular constraints, the modeling process for deriving dynamic

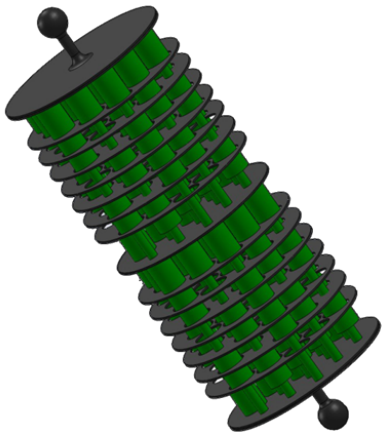


Figure 1: Example actuator array (robotic muscle). An actuator array is made from many small interconnected traditional actuation units (cells) which can combine to carry greater loads, achieve greater displacement, and exhibit higher robustness than their constituent actuators.

equations of motion and properties, and techniques for simplifying the control of the highly redundant actuators in an array. This work introduces the fingerprint, a compact representation of complex actuator array topologies, and an analysis method based on graph theoretic modeling to provide a systematic and efficient approach to generating system dynamics and static properties. The approach is designed to aid automation and simulation of cellular actuator arrays, allows for fast recalculation of different cell array topologies, and provides an intuitive base for future controls and physiology work on robotic muscles. While the dynamics representing a given cell array actuator could be generated using other means, such as Dymola, SimScape, SimMechanics, or other computational methods, the presented method allows the dynamics to be calculated with less human effort, less computational effort, and with greater speed. The method greatly outperforms other methods when needing to test many different actuator arrays as part of a design and engineering method as it allows for efficient recalculation given different array topologies.

Three physical cell array actuator systems have been constructed to validate the presented theory. The first is a static experiment utilizing solenoid actuation to study static strength and stochastic control methods of the actuators acting together in a large array. The second experiment tested the dynamic aspects of the theory using cells based on Miga NanoMuscle 704 SMA actuators and lightly damped pen springs, and the third setup is a more practical silicone-based SMA actuator array design inspired by the M-lines and Z-lines of sarcomeres in biological muscle. Finally examples and numerical results were used to show the ability of the presented theory to represent large and complex cellular actuator arrays, complex hierarchical outside dynamics interacting with arrays, and limited non-linear internal cell dynamics.

1.2 Motivation

1.2.1 Traditional Actuation Drawbacks

Many humanoid robots can be said to be biologically inspired in terms of morphology, but they are not biologically inspired in terms of actuation. In the field of robotics, the electromagnetic motor reins king. These motors come in all shapes and sizes from large industrial grade alternating current (AC) motors to small and accurate servo-motors, though most of these motors have common concerns which limit their use as the field of robotics continues to grow.

Figure 2: Issues with traditional motor-driven robotic devices include lack of flexibility which can lead to damage or injury.

A typical AC or DC motor, for instance, maintains highest efficiency while being run continuously at high speed, but loses this efficiency as speeds are constantly changing or when the motor needs to apply a force to an object without moving. The high speed aspect of this drawback can be countered by gearing down motor outputs - allowing the motor to run at high speed while driving a slower moving component - but this can create backlash, adding non-linear dynamic components which complicate system modeling and control. In order to reduce backlash and simplify control, the gearboxes are typically built to be stiff, which in turn can limit the ability to work safely around humans and fragile equipment, as shown in Fig. 2. Control methods can counteract this drawback by adding virtual flexibility; however this is not a property of the motor itself and can lead to dangerous consequences when sensors or control systems fail.

Due to the stiff nature of most robotic motors and actuators, robotic devices typically use a direct position control scheme. This works quite well in the controlled pick-up, bash-in, and move-along environment of industrial robotics, but as robotic applications expand they become less structured and require more control over the forces the robotic devices apply. This has led to the development of series elastic elements being combined with motors [64] [104] as shown in Fig. 3 and the development of more novel forms of actuation including shape memory alloy (SMA) [79] [24], piezoelectric (PZT) [90] [11] [56] [7] [18] [19] [27] [35] [55] [58], and many others [68] [6] [30] [63] [73]. Many of these newer systems, however, still suffer from concerns such as small force or displacement capability and thus have limited applicability on their own.

Figure 3: Series elastic actuator example.

Current actuators also usually lack robustness. Due to their stiffness and mass, typically only a single motor controls a single joint and if that motor fails the joint either freezes - becoming rigid and losing a degree of freedom - or breaks completely leaving a limp and useless device. This creates issues when working in a hostile environment or carrying a delicate load like a human. As an example, if a robotic arm hammering a component in a hostile environment (radioactive, war zone, etc.) has a single actuator fail, the arm and hammer can fall onto sensitive equipment (reactor core, bomb) and would subsequently require a second device or a human to enter the hostile environment to retrieve the robot risking further damage or injury. A robot with redundant actuation, however, may still lose the ability to continue hammering but would be able to secure the area, return the hammer properly, and move itself to safety on its own power.

Applications in robotics and mechatronics do combine multiple actuators together to accomplish a task; however the actuators generally work toward different degrees of freedom or different magnitudes of movement. Fig. 4 shows several examples. Reconfigurable robots [103] (left image) directly apply the idea of redundancy to adapt their form factors to changing challenges, however lack of flexibility of motor drivers prevents them from working in parallel to share loads. In parallel robotic devices, actuators can share heavy carrying loads. In these applications, the displacement capabilities are significantly reduced and the failure of a single unit can render the system unable to operate [12]. In micro-macro robotics, such as the arm of a hard drive [29] (right image), accuracy and speed can be gained by using a fast but inaccurate macro position device and then using a micro positioning device to hone in on the goal. Both actuators act in series on the same degree of freedom. This gives large displacement capabilities without sacrificing a small actuator's accuracy but still suffers from a lack of robustness and the inability of the actuators to share loads. The actuators arrays introduced and mathematically modeled in this thesis allow actuators to share loads, gain

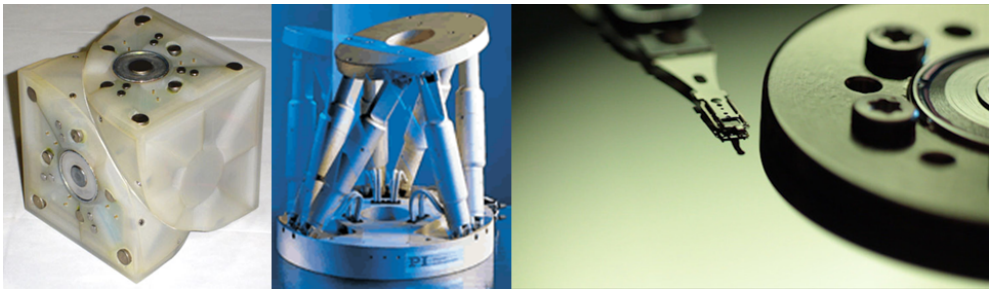


Figure 4: Multi-actuator examples. Left: Redundant actuation in reconfigurable robotics. Center: Increased load capacity in parallel robotics applications. Right: Increased displacement without sacrificing accuracy with micro-macro actuators.

greater displacement, and have greater robustness.

1.2.2 Biological Advantages

Multi-celled organisms have specialized muscle cells that move their body parts by elastic contraction in response to signals from the central nervous system. Rarely does a single muscle cell stand alone in an organism, but rather they are grouped together in huge quantities to form muscles. All of the muscle cells then work in concert to produce contractile motion, enabling movement [23]. Two lessons can be learned from this. First, the make-up of biological muscle reduces the need for an organism to produce extremely powerful single cells for all actuation while at the same time increasing robustness against individual cell failure. When creating robotic muscles, the same can be done. Many smaller actuators can be combined together in actuator arrays to produce large-scale motion and force while also gaining actuator array robustness against failure. The second lesson regards the importance of elasticity in the contraction of biological muscle cells. When rigid actuators attempt to work in concert, discrepancies in the magnitude of their motion can create internal forces which significantly reduce the overall effectiveness of the actuators. In some cases the actuators can even tear themselves apart. It is important, therefore, to ensure each actuator is either inherently flexible or is combined with a flexible element in order to gain positive overall effects.

Biological muscles are non-continuous, non-uniform, and consist of several types of muscle tissues, each with different levels of contractile speed and fatigability [23]. Motor units are bundles of muscle fibers with specific force, speed, and displacement capabilities stimulated by the nerve

impulses of a single motor neuron. A single muscle consisting of many motor units is thus structurally quantized in terms of force generation. It is believed that this is an important component to the generation of natural motions. Furthermore, actuation of muscle cells is a stochastic process. When nerve impulses stimulate a motor unit calcium ions are released. These are subsequently captured by the muscle cells causing them to contract. The probability of a single cell contracting is dependent on the concentration of calcium ions in the muscle, thus the brain and motor neurons do not directly control exactly which cells activate [23] [102]. This can yield insight into the design of control systems for actuator arrays. By selecting non-uniform actuators (from the standpoint of force) and arranging actuators into non-uniform structures we can create force discretization profiles which should allow systems to produce natural motions. By controlling redundant actuators with a stochastic process, control effort and wiring complexity can be reduced allowing a cheaper robotic muscle to be produced.

The primary way to endow robots with biological abilities is to equip them with biologically inspired actuation. This work uses the term 'biologically-inspired' in regard to actuation in robotics, namely, that the actuators themselves have structural and operational characteristics in common with muscles. Specifically, the actuator systems of interest will 1) have a modular structure: the actuator selectively activates distinct units (recruitment), and 2) possess elasticity allowing impulse signals to produce a smooth contraction. The modular architecture may be non-uniform, or hybrid, consisting of different materials and actuation technologies. Different sizes within a particular actuator technology can also be used, such as varying the diameter of shape memory alloy actuators; a trade-off between speed and force.

1.2.3 Natural Motion

For decades researchers in physiology have been trying to model and generate natural motions (the movements created by biological systems) in order to both gain a greater understanding of biological muscle and produce motions similar to muscle. A key step in understanding these motions is producing a physical, well understood test platform with a dynamic model closely resembling biological muscle. This test bed can then serve as the basis for controls experiments to better understand the interrelated nature of the nervous system and the muscles, for kinematics/dynamics

experiments to understand balance and synergies, and for building viable, full-strength, and safe muscles for prosthetics, rehabilitation, human force amplification, and humanoid robotic systems. Electromagnetic motors, however, do not typically have dynamic models similar to biological muscle and thus cannot fill this need.

Developing the ability to produce natural motions benefits therapy and rehabilitation for patients who have suffered muscle damage or degradation, allowing patients to function more easily in society and, in the best case, make the degradation or loss of their original limb a non-issue in ordinary life. The advances also aid in the development of naturally moving prosthetic devices and facilitate the development of human force amplification exoskeletons. Finally, robotic devices powered by standard motors tend to move "like robots" instead of like biological organisms, which causes uncertainty and hesitation in humans working nearby. By understanding natural motion - and developing actuators and control systems that mechanically duplicate that motion - we can create robotic systems that are safer to work near, cause less hesitation and fear, and are better able to interact with the human environments and tools of the world. This yields more life-like and capable humanoid robotic systems, and is essential as robots continue to integrate into human society.

1.2.4 Artificial Muscles for Rehabilitation and Quality-of-Life

Stroke and Amyotrophic Lateral Sclerosis (ALS) give case examples of how human-like cellular actuator arrays can be very helpful in improving quality of life.

Stroke is a condition where blood flow to portions of the brain is impeded by an interruption of blood supply (ischemic stroke) or hemorrhaging (hemorrhagic stroke), killing many of the brain cells including those associated with movement [77] [20]. Over time, the brain is able to remap the movement commands to working cells allowing the patient to regain a significant portion of his or her pre-stroke mobility.

Task-oriented training, one of the methods involved in rehabilitation of stroke victims, involves moving the stroke patient's limbs while the patient visualizes the movement. Over time the association becomes an active control and the patient is able to initiate movement; weakly at first but with growing strength [83] [46]. Patients are generally released once they reach a level of strength and mobility at which they become self-dependent in everyday tasks.

The devices used in therapy of the stroke patients generally consist of frames constraining motion the doctor initiates manually or use motor-driven robotic systems [101] [46] [47] [86] [43] [88]; neither of which have human-like properties. Furthermore, the current therapy devices have a sole use in the task-oriented training portion of the rehabilitation process, but cannot be easily transitioned into assistive devices due to the rigid nature of the motors.

With a more human-like cellular actuator arrays, the devices can move as the original limb moved during the associative learning phase (improving therapy results) and can be scaled between a pure task-oriented rehabilitation device and a force-assistive amplification device, aiding the patient in becoming self-sufficient.

Those who suffer from Amyotrophic Lateral Sclerosis (ALS) or Lou Gehrig's disease lose muscle control slowly over time due to a breakdown of the motor neurons that control voluntary muscle movement [67] [53] [39]. Though ALS is fatal, efforts can be made to extend its victims quality-of-life as the disease takes hold. Currently these efforts center on pain control, respiratory management, and extending mobility and self-sufficiency [4] [22] [39]. Extension of mobility and self-sufficiency are currently done through ongoing physical support (from canes, to ankle-foot-orthotics, to powered mobile chairs), independence devices (special eating implements, toilet seats, bath-tub modifications, etc.) and assistive communication devices. These allow a user to interact with and move through the world, but still allow only limited increases to the quality-of-life.

Human-like cellular actuator arrays have the potential to serve as force amplification devices which can be scaled up over time as the disease progresses, essentially allowing the actuator array to become the patient's muscles. This allows sufferers significantly increased self-reliance and quality-of-life. Since the muscle degradation generally occurs over the course of months, a human-like actuator would be able to learn its user's motions and be able to use this knowledge and its internal dynamics to accurately reproduce those motions even as the user's neurons degrade.

While this work does not focus on patient studies or the design of muscles specific to rehabilitation and quality-of-life improvements, the modeling methods and design guidelines presented in this work enable the engineering of patient specific artificial muscles (actuator arrays) from existing actuator technologies. This engineering then allows researchers to develop improved devices and test those devices with the patients. Separately, the mathematics introduced in this work can aid

physiology researchers in better understanding how human muscles work, allowing them to develop focused treatment techniques. Both of these are critical steps to improving future rehabilitation and quality-of-life techniques.

1.3 Background

1.3.1 Natural Motion

Generation of natural movements, or the movements created by biological systems including humans and animals, has been one of the biggest scientific questions discussed in physiology for decades. Based on visual perception of biological motion humans can easily distinguish between motions created by artificial systems such as robots and those of humans [36] [87], and human natural movements are stereotypical [13] [99] [98] [5] [65] [16] [57]. Excluding small individual differences, motions with the same objective, such as arm-reaching or walking, tend to be very similar among different people. Stereotypical adaptation processes are also observed in neurological patients [14] [44] [66]. Together, these observations imply the existence of general rules for coordination of multiple muscles to generate natural movements.

Humans can easily distinguish natural motions from artificially created motions, even from limited visual information such as the trajectories of light points attached to moving limbs, and the natural motions are stereotypical [36] [87]. A robotic limb which 'moves like a robot' gives a feeling trepidation to those working nearby. The definition of what causes a natural motion is still an open subject of research however one potential explanation is signal dependent noise [37]. As robotic movements become more natural, people interacting with the robot are easier able to predict the robot's movements, even subconsciously, and feel more comfortable close to the robot. Such perceived realism of movement has been an interest in Computer Graphics [26], Psychology [32], and Social Robotics [61] [17].

1.3.2 Muscle Optimality Criteria

Many research efforts searching for these underlying rules of natural movement focus on the variability of motor system timing spike generation which results in variability of muscle forces [31] [10] [37] [80] [85] [84] [28] [59]. Harris and Wolpert showed that the standard deviation (SD) of the commanded signal varies proportionally with the mean of the command signal [31]. They

argued that this signal-dependent noise plays a central role in motor control and that movements are organized to minimize the variance at the endpoint. When taking a point-to-point motion, for example, Harris and Wolpert argue that the optimal strategy is to reach the desired endpoint with the minimum error due to the noise in the motor signals. The trajectory, velocity, and acceleration for the motion are determined by solving this optimization problem. They discovered that the calculated trajectories according to this criterion are similar to the stereotypical human motions, or natural movements [31]. Todorov et al. [85] suggested that this optimization strategy is equivalent to $\min(\sum(\sigma^k))$, the optimality principle in muscle force generation [13] [98] [5] [65]; that is, a minimization of the sum of muscular stress, σ, raised to a power, k, which is subject to the force/torque constraints of a given task. Although physiologically-based cost functions with this structure have been used in biomechanics and physiology to predict redundant muscle forces, its physiological meaning was not clear. Simmons et al. has applied this concept to optimal control of a standard 2-DOF manipulator with non-redundant planar rotary joints [76], however signal-dependent noise was computed and merely added to the control signal to mimic a natural system, but this noise never exists in AC/DC rotary motors. This is the beginning of a bottom-up approach, but large gaps still remain. The work presented by Harris and Wolpert is intuitively understandable but more effort is needed to reach fully conclusive results.

Cellular actuator arrays and the mathematical models presented in this thesis provide a well understood physical platform for further research understanding muscle optimality criteria and the control signals used to control human muscles. The signal distribution methods presented here demonstrate the same characteristics as the signal-dependent noise observed in biological muscle, and developing optimal control methods for the cellular actuator arrays will likely lead to greater understanding of the brain's control over biological muscle. Furthermore, since optimality criteria like signal dependent noise is believed to lead to natural motion, actuator arrays which follow this optimality criteria should be capable of achieving natural motion.

1.3.3 Biological Muscle Structure

The detailed mechanism of biological muscles is still unknown; however, biological muscles suggest important design guidelines for new robot actuators. Muscles are dynamic systems with

high degrees of internal freedom and relatively few inputs and outputs. A muscle is composed of numerous sarcomeres, small functional units which contract to provide varying levels of displacement and stiffness, and internal force, velocity, and displacement receptors, i.e. Golgi tendon organs and muscle spindles [50] [34] [33] [89] [82]. In vertebrates, impulses from the brain are carried by the spinal cord to motor neurons which in turn distribute the impulses to the muscle fibers, via synaptic terminals at neuromuscular junctions, and activate the sarcomeres. It is known that the activation of sarcomeres is a stochastic process, rather than a deterministic control, because of the diffusion of calcium ions at these junctions [40]. The brain does not know or control the displacement or force of any given sarcomere, only the overall muscle. This is part of the inspiration being using a stochastic process to control the actuator arrays. Due to the large redundancy of sarcomeres, Golgi tendon organs, muscle spindles, motor neurons, etc., even if some of the components are not functional the rest of the functional components can compensate to maintain total functionality. This is critical for biological systems, and very useful when reproduced in robotic actuators.

The contrast between the actuators used in biological systems (i.e., muscles) and robotic systems (e.g., rotary DC motors) is another possible cause of the difference between movements of humans and machines. Biological muscles are completely different from their electromagnetic counterparts, both in constitution and function. [50] [34] [102] [42] [33] [89] [82]. They are energy efficient, compact, lightweight, naturally compliant, and silent. As new robotic actuator technologies are studied, one of the goals is to invent artificial muscle actuators with these properties. In recent years, great progress in the development of new actuators has been presented using, e.g., shape memory alloys [79] [24], pneumatic rubber actuators [68] [6], conductive polymers [30] [63] [73], and piezoelectric materials [90] [11] [56] [7] [18] [19] [27] [35] [55] [58]. These novel actuators are particularly useful in human assistive technologies [100] [1] [43] [48] [62] [86] [88] [93] and biomedical robotic applications. Beyond the obvious engineering benefits, use of these new actuator technologies in robotic systems gives researchers an experimental platform to explore the basis behind human motion and robotic duplication of this motion.

1.3.4 Stochastic Effects

It is known from prior literature that the activation of sarcomeres is not governed by deterministic control, but is instead affected by a stochastic process due to the diffusion of calcium ions [40]. It is unknown to what extent this or other properties affect whether a movement is perceived to be natural, but developing artificial muscles with stochastic properties will hopefully shed more light on this active research area. The stochasticity in networks has been attracting increased attention from broad academic areas such as computer science (robust computer networks), biology (robustness of human immune systems), and urban engineering (robust transportation networks) [75] [38] [2] [81] [54] [78], all of which serve as references when designing and analyzing the stochastic systems.

A vast number of mechanical actuator units can be controlled using the same stochastic process as observed in biological muscle [94]. Instead of wiring many control lines to each individual cell, each cellular actuator has a stochastic local control unit that receives a broadcasted signal from a central control unit and changes its state in a simple ON-OFF manner [72]. This stochastic coordination, named stochastic recruitment, was inspired by the calcium diffusion process in the signal transduction of muscles [40] and drastically improves wiring and addressing issues. The ON-OFF control does not require high-performance actuator driver circuits and it resolves the problem associated with hysteresis of the actuator material. The stochastic cellular actuator array also has a high robustness against failure of the actuator units. Stochastic control theory [45] [21] proves the system stability, and despite having no deterministic coordination, the ensemble of the cellular actuators robustly tracks a given trajectory even if, for example, 30% of the cells fail [94].

The idea of the Stochastic Cellular Actuator proposed a new architecture for robot actuators inspired by muscle properties, which in turn has a potential to be a novel approach to synthesize biologically-inspired actuators [95] [96] [93] [97] [94] [91]. This concept connects small actuator units (PZT in this case) in series or in parallel to compose a macro-size actuator array. A new structure called a nested rhombus multi-layer mechanism [96] [93] [97] was proposed to amplify the displacement of piezoelectric ceramic actuators, such as lead zirconate titanate (PZT), in order to create a novel actuator unit with 20-30% effective strain; which is comparable to natural skeletal muscles [50] [34]. This mechanism drastically mitigates the drawbacks of PZT, i.e. its extremely

small strain of only 0.1 % [90]. In addition to the strain amplification, the mechanisms are sufficient to produce muscle-like compliance in the actuator unit [96] [93] [97].

Although the characteristics of broadcast control have been investigated [94], this used only a simple array topology where actuator cells were connected in series and physical interaction among cells was ignored. The current work presents a method to generate and analyze complex actuator array topologies where actuator cells are connected in series, in parallel, or a mixture of both, interacting through mechanical compliance. Additionally, the restrictions to only using PZT or simple structures are lifted allowing for a diverse set of actuators ranging from shape memory alloy to PZT to even pneumatic and hydraulic technologies.

1.3.5 Graph Theoretic Modeling

One of the principle challenges in developing cellular actuator arrays for the above applications involves efficient dynamic modeling of the arrays and the systems they connect to. While these dynamics can be developed from base Newtonian or Lagrangian principles, this process is computationally expensive and difficult to automate [52]. Many methods for graphically representing complex multi-body systems exist for various domains (mechanical, electrical, hydraulic, etc.), though many are based on bond graphs and graphic theoretic modeling (GTM) [69].

Bond graphs focus on power relationships between components in an interconnected system. They are non-domain specific and can accommodate multiple domains in a single system (mechanical, electrical, hydraulic, etc.). Power, the multiplication of effort (force, voltage, pressure) and flow (displacement, current, mass flow), through the elements is represented by bi-directional connectivity of the elements which define the dynamic relationships in system. Bond graph methods provide simplified notation compared to other methods, however the complexity of developing the dynamic equations of motion does not scale to large systems as well as GTM.

GTM presents an organized and scalable technique for developing dynamic models of interconnected physical systems using the basis of linear graph theory [3]. Linear graph theory is a subset of topology, the study of how things are connected (which is itself a discipline of mathematical combinatorics). GTM combines matrix functions and edge/node network modeling from linear

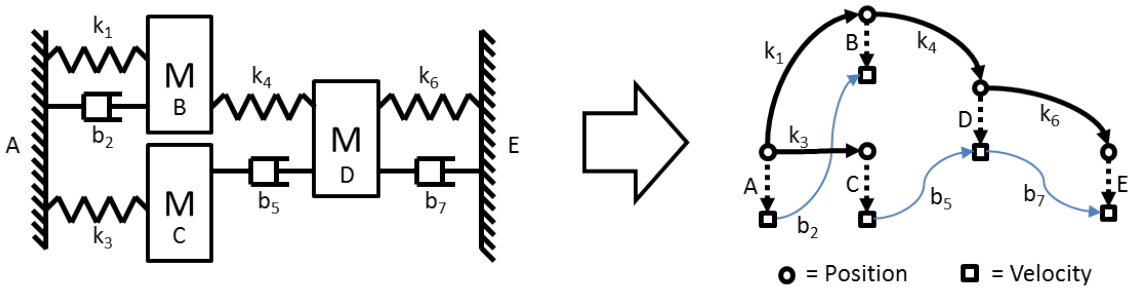


Figure 5: Graph Theoretic modeling Example.

graph theory and combines it with physical engineering components to develop algorithms for automatically generating differential equations modeling the dynamics of a physical systems [15] [25]. The theoretical basis for GTM came from Leonhard Euler in the 1700s and was combined into a unifying system dynamics theory for multi-domain engineering applications by a text by Koenig, Tokad, and Kesavan [41]. This has be subsequently expanded into different fields and applications ranging throughout mechanical, electrical, hydraulic, and other domains [9] [51] and integrated into commercial software packages [71].

McPhee has presented a series of publications on the applications of linear graph theory to flexible multi-body systems [52] [69] [70] [74]. The key concept is to introduce a matrix, named 'incidence matrix', to represent a complex topology of a multi-body system. This works by viewing physical components of a system as separate edges of a graph connecting to common node points. An edge of the graph - generally an element like a spring or damper in a mechanical system or a resistor, capacitor, inductor, or driver in an electrical system - then connects to terminal nodes where the effects of the edges on the node are governed by physical laws (Force equilibrium in mechanical systems, Kirchhoff's Current Law in electrical systems). The incidence matrix, I defines which nodes are connected to which edges through a mapping, where a 1 or -1 signifies a physical connection (reference direction determines sign) and a 0 signifies a lack of a connection. Fig. 5 shows an example system with its associated linear graph and (1) gives the incidence matrix for the example. The position and velocity are separated in the example for the sake of clarity, though they are treated together with GTM.

$$
[I] = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}
\begin{array}{c}
\begin{array}{ccccccc} k_1 & b_2 & k_3 & k_4 & b_5 & k_6 & b_7 \end{array} \\
\left[\begin{array}{ccccccc}
1 & 1 & 1 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & -1 & -1
\end{array}\right]
\end{array}
\tag{1}
$$

A linear graph can be viewed as a data flow map through the electromechanical system. By choosing a starting position, such as node A in the example, paths can be found through the connecting edges to each of the other nodes in the graph. Each edge is given a "through variable" - a variable that must be measured in series with the edge, like force - and each node is given an "across" variable - a variable that must be measured as a difference between two nodes, like displacement. By following the graph from node through an edge to a connecting node, the relationship between the node across variables can be determined. For example, in Fig. 5 node A connects to node B through spring k_1. If the force (through variable) in the spring is known, the displacement between A and B can be determined. The incidence matrix provides this mapping between variables.

When systems have multiple paths between components, the process gets trickier. In Fig. 5, the relationship between A and B can be determined either through spring k_1 or through damper b_2. This redundancy creates linear dependence between the nodes and prevents an analytical solution for the variables in the system. Preventing this issue taps into the idea of cutsets from the study of topology. By choosing the correct elements to drop from the system, the graph becomes a tree where only a single path exists between any two nodes. Once this tree is generated, a linear independent set of equations (based on across variables) can be determined. In Fig. 5 each of the dampers (b_2, b_5, and b_7) can be dropped leading to a reduced incidence matrix given by (2).

$$[I_{simp}] = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{cccc} k_1 & k_3 & k_4 & k_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} \end{array} \tag{2}$$

The equations/elements that are part of this simplified form are called the 'cutset' equations for the system because cutting any one of them would yield two separate and unrelated systems. Determining the cutset equations can be done through manual selection to maintain physical significance of the system variables, or it can be done by applying Gauss-Jordan elimination to obtain the row-reduced echelon form followed by eliminating the unneeded columns.

Once the cutset equations are found, final dynamics are determined through applying 'terminal' equations at each node to determine through variable values. For rigid body dynamics, the terminal equations for each node proportionally relate the sum of the forces (through variables) to the acceleration of the node (across variable). This is shown in (3) for node B in Fig. 5.

$$m_b \cdot \ddot{x}_b = -\sum F_b = k_1 \cdot (x_b - x_a) + b_2 \cdot (\dot{x}_b - \dot{x}_a) - k_4 \cdot (x_d - x_b) \tag{3}$$

The through variables are solved for in terms of the node across variables, and then the result is plugged into the cutset equations to determine the final dynamic equations of motion for the system in terms of the through variables. [51] provides an in-depth overview of the graph theoretic modeling as well as many examples for electromechanical systems, and readers are encouraged to look here for additional details.

While the GTM method does generate the equations of motion for the dynamic systems of interest to the current work, they require careful manual selection of state variables - or 'cutsets' - to have physical meaning. While an automatic method for cutset selection using Gauss-Jordan elimination, this results in meaningless state variables which make the interpretation of modeling/simulation results more difficult.

The 'incidence matrix' in GTM consists of a list of connections between all elements in the system (which must be provided as an input). Since cellular actuator array cells generally have similar or identical internal dynamics, the dynamic equations do not need to be separately generated for each cell. Once the internal dynamics are known, the self-contained cell structures should separate the internal cell dynamics from the topology dynamics. This allows each cell to be treated as a single edge of the linear graph, thus greatly shrinking the 'incidence matrix'.

[8] presents a similar simplifying method called Newton-Raphson Mixed Nodal Tableau by separating internal dynamics of photovoltaic cells and then 'stamping' these repeated dynamics into a larger system as a single element and using GTM to generate the final equations of motion. This allowed for a simpler process for generating the dynamic equations of motion and allowed for the non-linear dynamics of the photovoltaic system while treating the rest of the system as linear. This method is similar to the method pursued in the current work where repeating internal cell dynamics with a possible non-linear element are connected, or 'stamped', into the topology of an actuator array. While the general idea is similar, [8] focused on electrical and specifically photovoltaic power systems while the current work deals with mechanical structures and specifically bio-inspired muscle systems.

1.4 Research Goals

This research aims to support the goals of creating natural motion, overcoming the limits of traditional actuation, and creating human-safe actuators by introducing cellular actuator arrays with redundant and biologically inspired actuation and the mathematical modeling methods needed to represent and design them. Fig. 6 shows challenges the current work solves as well as their relationship to the overarching goals. The challenges are specifically laid out below.

- To introduce a method to leverage current actuator technologies to build biologically inspired human-like artificial muscles called cellular actuator arrays
- To represent complex cellular actuator array topologies and develop a deterministic method for generating and exploring possible topologies given a number of cells
- To generate the dynamic equations of motion for varied cellular actuator array topologies

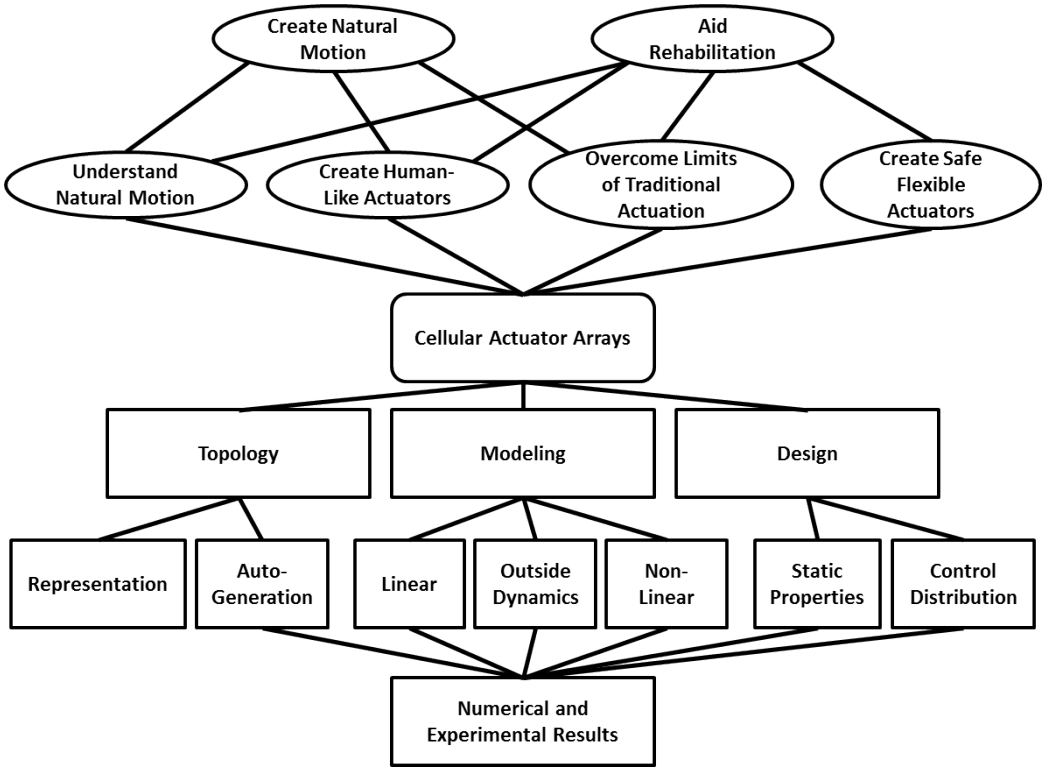


Figure 6: Research Goals. Motivations are highlighted by circles while items this work covers are highlighted with rectangles.

with a method that accommodates any valid linear internal cell dynamics, well-structured non-linear internal cell dynamics, and outside dynamic structure

- To determine static properties and control signal distribution methods as criteria to aid in actuator array design and selection
- To numerically and experimentally validate results using physical hardware and numerical simulation

CHAPTER II

ARRAY STRUCTURE

This chapter addresses the challenges of representing complex cellular actuator array topologies, exploring possible topologies given certain criteria, and determining the similarity between related topologies. The chapter begins by first addressing the main components of an array, the cell and the mass, and restrictions on what types of actuators can be used in each cell of an array. The discussion then moves into representing arrays using a modified incidence matrix from graph theoretic modeling and introduces a reduced representation of well-ordered topologies called a fingerprint. The fingerprint is then used to deterministically identify possible array topologies so that they can be evaluated. The chapter closes by presenting a method for determining similarity between topologies which future work can use to search for topologies with desirable properties.

2.1 Cell

A cell consists of a modular set of dynamic elements including a flexible linear actuator - or a non-flexible linear actuator connected in series with a flexible element(s) - and any elements needed to properly model the modular component of the array. Cell flexibility is needed to mitigate the differences in length between cells that are active and those that are not. The actuator can be based on any linear actuation technology so long as 1) the actuator can be represented as a spring, stiff or flexible, with a pure force preloading the spring; 2) the actuator's pure force follows a known force versus time function, $\theta(t)$, when activated; 3) that function is minimally dependent on external cell load and therefore the interaction is negligible. The first criteria is generally true of most actuators, whether they are mostly stiff (like a motor) or flexible (like shape memory alloy). For stiff actuators, a large force acts on a stiff spring to represent the actuator's displacement. The second criteria is generally easy to achieve through measurement of the force generated in the actuator's flexible element when the actuator is fully blocked. For stiff actuators, estimating the stiffness of the actuator and measuring the displacement is usually sufficient to generate the force function. The final criteria implies that the external effects on the cell, whether displacement or force, should have

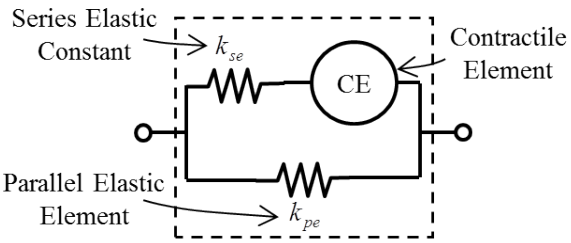


Figure 7: Hill-Type model.

little effect on the actuator's ability to reach the commanded position. This can be met through design, for example by either limiting outside force or choosing a low stiffness flexible element to combine in series with an actuator, ensuring the actuator can move through its full displacement. Actuator technologies such as piezoelectrics, ultrasonic motors, linear stepper motors, hydraulics, pneumatics, and shape memory alloy (SMA) can all meet the three criteria.

While the theory presented in this work holds for any internal cell model meeting the above criteria, this work focuses on the Hill-Type cell model from studies in physiology. This is done to work towards the goal of natural motion and to provide a consistent example for explanation purposes. Different cellular actuator array applications may have goals requiring vastly different cell dynamics than is provided by this model. The Hill-Type cell model consists of a parallel elastic element connected in parallel with the combination of a series elastic element and a contractile element, which are themselves connected in series. This is shown in Fig. 7. In most accepted Hill-Type model literature, the parallel elastic and series elastic elements are modeled as springs while the contractile element does not have consistently defined dynamics. In many cases the contractile element is simply treated as a function of force versus time. In physiology this is done due to lack of knowledge of the dynamics of the contractile element, an open area of research, however the current work uses it as a method of model simplification. In this work the actuator, or contractile element in the hill-type model, is treated as a spring acted on by a pure force preload. The result is a shortening of the neutral length of the actuator's 'spring' by the amount of the displacement of the actuator.

The series elastic element represents the connective material between actuators and can be anything from stiff metal to springs, silicone, or even cloth. With fragile actuators the series elastic element should have lower stiffness to protect the actuator, however this weakens the overall force output of the system. The Hill-Type cell model can be modified to add in an additional damper in

parallel with the series elastic element or the contractile element to represent the damping effects of many actuator or material choices. This allows for additional design options. For the purpose of theory explanation the internal cell dynamics are linear to allow for representation by a linear model, though chapter 5 extends the theory to include some non-linearities (like creep and hysteresis). The parallel elastic element represents spring-like forces carried across the cell but not seen by the series elastic and contractile elements. These are generally kept small to avoid internal compressive forces but are maintained through the theory to allow for greater design flexibility. The methods presented in the following theory are generalized to accept any internal dynamics as long as the above criteria are met. They do not require the internal structure of the cell to follow a Hill-Type cell model.

While also not required, this work focuses on operating each cell in an fully-on or fully-off fashion. This allows actuator hysteresis to be avoided, can simplify modeling, control, and physical wiring and electrical requirements. With a large number of cells, proper discretization of the overall array output force/displacement is still maintained.

In many cases, such as with SMA actuators, this also allows non-linear effects to be decoupled from linear dynamics. The non-linear effects can then be modeled using a forward-loop input shaping instead of a dynamic effect, simplifying modeling and simulation. Specifically in the case of SMA actuators, the actuator can be modeled as a spring with a pure force input acting across it while heating and cooling are accounted for by adding a non-linear delay to the input force signal.

2.2 Connecting Structures (Masses)

Cell array actuators are collections of cells connected in various arrangements, or topologies, to provide the large-scale motion required of a muscle system. The topology of the array is critical to determining the array's properties. Having more cells in series tends to give an array more displacement, more in parallel gives more force and higher robustness, and having a non-uniform structure can give higher force discretization and more fine-tuned control.

If the above cell criteria is met, internal dynamics of the cell can be decoupled from the dynamics of the array. This is done by treating the cell as an input-output function, taking in the positions and velocities of the cell endpoints (right side and left side) and outputting the force at those endpoints. The array itself consists of the cells and the connecting structures between the cells. The connecting

structures are assumed to have mass and are called masses. All cells connect directly to masses and all masses connect directly to cells or the endpoints. No mass is connected directly to any other mass and likewise for cells.

2.3 Incidence Matrices

The methods presented in this work build on the concepts of graph theoretic modeling (GTM) presented in chapter 1. In GTM, incidence matrices are used to represent the connections between cells (topology) and then mathematical operations are carried out on these incidence matrices to develop the dynamic equations of motion. This work follows the same concept, but defines the incidence matrices differently to better suit the application to actuator arrays.

The driving simplification behind modeling actuator arrays comes in the ability to separate the dynamics about each cell, essentially treating cells as input-output functions relating endpoint motion and inputs to output forces. For this reason, the incidence matrices for the current work are separated into incoming connections (connections to the right of a cell; incoming relative to masses) and outgoing connections (connections to the left of a cell; outgoing relative to masses) as shown in Fig. 8. The incoming connections matrix is $\mathbf{H}$ and the outgoing connections matrix is $\mathbf{G}$. $\mathbf{H}$ is an N-by-M matrix and $\mathbf{G}$ is an M-by-N matrix where N is the number of cells in the array and M is the number of masses. The elements of both $\mathbf{H}$ and $\mathbf{G}$ are either 1, representing a connection between the associated cell (row in $\mathbf{G}$, column in $\mathbf{H}$) and mass (column in $\mathbf{G}$, row in $\mathbf{H}$), or 0, representing no connection. This is shown in equation (4).

$$\begin{aligned}
\mathbf{G}_{m,n} &= \begin{cases} 0 & \text{if mass } m \text{ is not connected to the left of cell } n \\ 1 & \text{if mass } m \text{ is connected to the left of cell } n \end{cases} \\
\mathbf{H}_{n,m} &= \begin{cases} 0 & \text{if mass } m \text{ is not connected to the right of cell } n \\ 1 & \text{if mass } m \text{ is connected to the right of cell } n \end{cases}
\end{aligned} \tag{4}$$

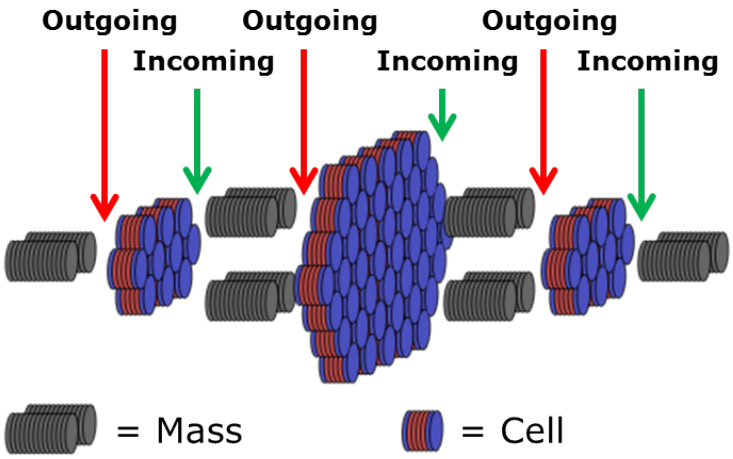


Figure 8: Explanation of incidence matrix components. Outgoing connections are represented by $\mathbf{G}$ and incoming connections are represented by $\mathbf{H}$.

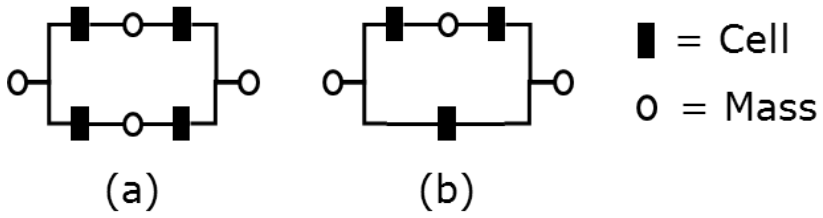


Figure 9: Examples of (a) a layer based actuator array approach and (b) a non-layer based actuator array. The layer based array has two cells on each path between array endpoints while the non-layer based array has one path with one and one with two. With identical cells, the non-layer based array would likely generate internal compressive forces.

2.4 Fingerprint

2.4.1 Fingerprint Definition

While not required, many intelligently designed cellular actuator array topologies will adopt a layer-based approach to keep all cells in tension. In a layer-based approach all cells have identical relaxed lengths, identical unloaded actuated lengths, and are arranged such that all topological paths between array endpoints contain the same number of cells (and masses). Fig. 9 shows an examples of (a) a layer based approach and (b) a non-layer based approach. This especially useful when using directionally non-linear components like wires or depending on tension forces to keep cell elements aligned.

These arrays can be represented using a 'fingerprint,' a layer based set of two row matrices encoding the connection information of the array. The fingerprint transcription consists of segmenting

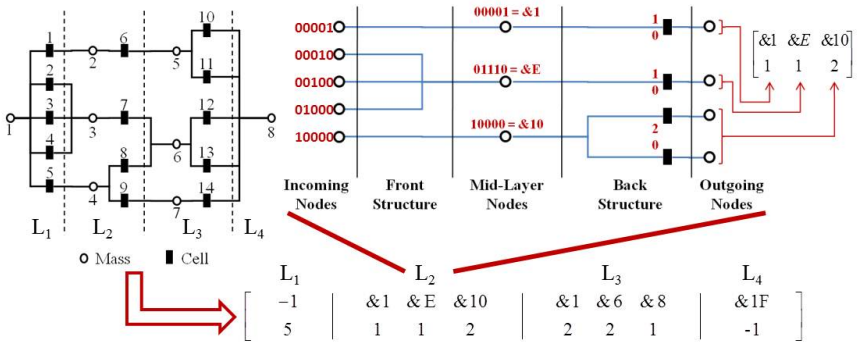


Figure 10: Example of building a fingerprint from an actuator array topology.

the array into layers just after the cells and before the masses those cells connect to. The layer can then be represented as an incoming structure and an outgoing structure relative to the layer's masses. For each mass, a hexadecimal number represents a binary encoding of the cells from the previous layer that the masses connects to. For example, in Fig. 10 the incoming structure of layer two would be represented by [&1, &E, &10] showing the first mass connects to the first cell of layer one (00001); the second mass connects to the next three cells of layer one (01110); and the third mass connects to layer one's final cell (10000). The back structure for each mass is represented by the number of cells connected to that mass. # is used to signify the beginning of the array as there is no previous layer to provide an incoming structure. Likewise, # is also used to signify the end of the array as there are no additional cells for the final layer of the array. Fig. 10 shows the complete transcription of a fourteen cell array with special focus given to the second layer for explanation purposes.

2.4.2 Fingerprint to Incidence Matrix Relationship

H can be populated automatically from the first row of the fingerprint. Column one is skipped as there are no left-connecting cells to the leftmost mass. Columns two through M are populated as shown in Fig. 11. Each subsequent layer begins one row down from the previous layer's lowest entry. **G** can be populated automatically from the second row of the fingerprint. The last row is skipped as there are no right-connecting cells to the rightmost mass. Rows one through $N - 1$ are populated as shown in Fig. 11. Each subsequent layer begins one column right of the previous layer's rightmost entry and contains a number of 1's corresponding to the number of outgoing cells

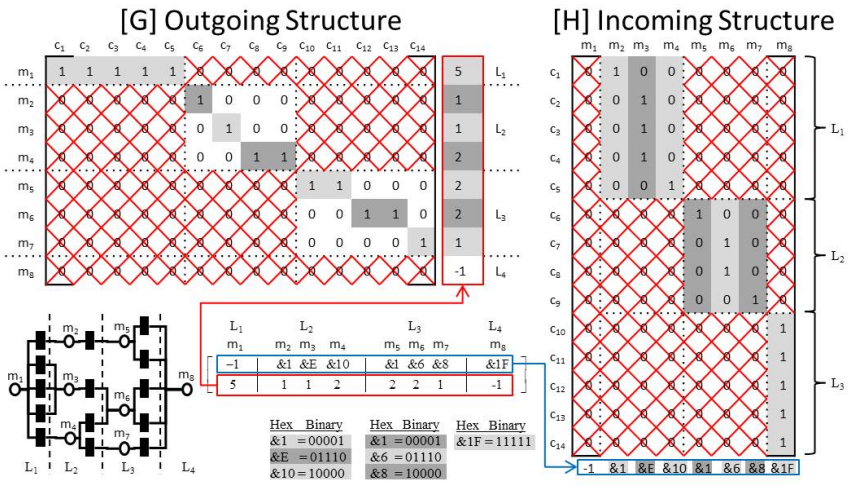


Figure 11: Example of building a fingerprint from an actuator array topology.

for each mass of the layer, or the elements of the second row of the fingerprint.

2.4.3 Fingerprint Autogeneration

In order to do an in-depth survey of actuator array properties, a method of generating all possible actuator array topologies given a fixed number of cells was needed. The auto-generation problem is broken into a layer by layer iterative approach. Each layer is then further broken down into its front section, which contains the information for bringing the incoming nodes down to a certain number of mid-layer nodes, and its back section, which contains the information for expanding each mid-layer node to a certain number of cells and outgoing nodes, as was shown in Fig. 10.

For each front section, all of the possible methods for connecting the incoming nodes are explored using a recursive function. This function takes in a binary input, a, showing which incoming nodes are not yet connected for the current test case. For example, in Fig. 12 incoming nodes 2 and 4 are not connected, thus the input would be 1010. The function iterates upward in binary from $s = 1$ to $s = 2^{\ell_a} - 1$ where ℓ_a is the length of a. Each s has 0's concatenated to the left side until it is the same length as a and is then subtracted from a element-wise. If there are any resulting -1's then s is tossed out, otherwise s is stored in hex in array $\boldsymbol{c}$ as c_l. This becomes front structure number as part of a fingerprint layer. To finish generating the front structure for each stored c_l, the function is called recursively with an input of $a_n = a - c_l$. No additional level of recursion is made for inputs of $a_n = 0$. The function returns the structure (Fig. 12) if $\boldsymbol{c}$ has L entries with $L > 1$.

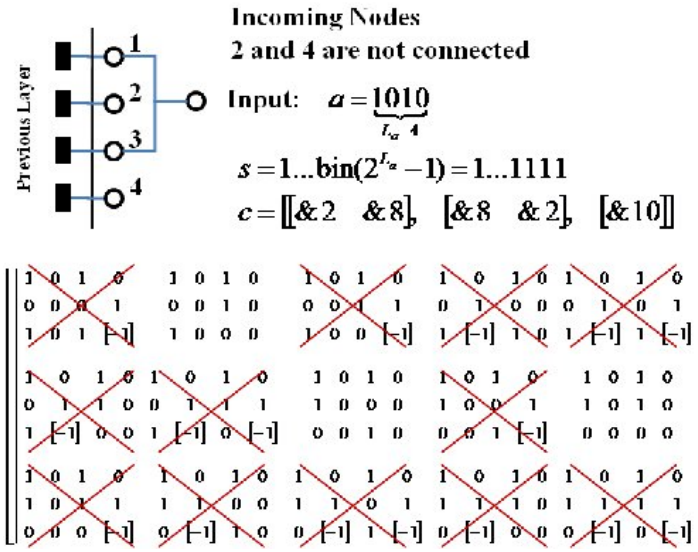


Figure 12: Front section autogeneration example.

$$[c_1, \text{recursion}]\,; \cdots ; [c_l, \text{recursion}]\,; \cdots ; [c_L] \tag{5}$$

From the example illustrated in Fig. 12, the first entry in $\boldsymbol{c}$ is $\&2$ representing a connection to only incoming node 2. The next layer of recursion then takes in an input of $1010 - 0010 = 1000$ and returns $\&8$. When the function finishes, the result is a structure of all of the frontal connection possibilities, each of which can have one or more mid-layer nodes. Since each mid-layer node must have at least one cells attached to it, those frontal connections which have more mid-layer nodes than remaining cells are tossed out.

Another recursive function is run to generate the back structure for each front structure. This function takes in the number of mid-layer nodes and the number of cells left to be placed, and returns all of the back connection possibilities. The function iterates from $g = 1$ to $g = z - m + 1$, where z is the number of remaining cells, and m is the number of remaining mid-layer nodes. The number of cells connected to the first mid-layer node is equal to current value of iteration, g. The upper limit of the iteration is due to the need for each mid-layer node to have at least one cell attached. The result function is then recursively called with w remaining cells determined by $w = z - g$ and o mid-layer nodes determined by $o = m - 1$.

Fig. 13 shows a process tree for how fingerprints were generated for arrays with 4 cells. All

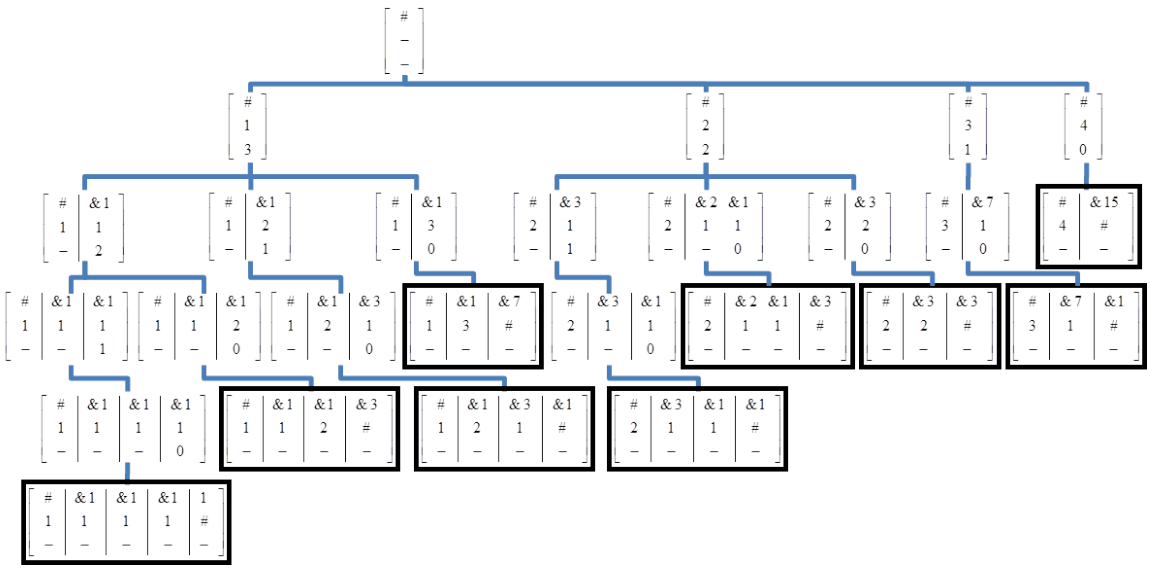


Figure 13: Autogeneration process tree for generating fingerprints for arrays with 4 cells. The third row in each representation shows unallocated cells remaining.

arrays begin with a single incoming node and a number of cells. The front end for layer 1 is always #, and the back end uses between 1 and N cells where N is the total number of cells. The second and subsequent layers take each of the previous layer's configurations and generate all of the possible configurations for that layer. When the function is called with no remaining cells, the last layer is always filled in to connect all mid-layer nodes to a single output node and put #'s for the number of cells.

Fig. 14 shows the 23 topologies for arrays with 5 cells, and Fig. 15 shows the number of connection possibilities for arrays up to 10 cells and the associated computational effort to generate the possible array structures.

The number of topologies for 2 – 8 cells are: 2 topologies for 2 cells, 4 topologies for 3 cells, 9 topologies for 4 cells, 23 topologies for 5 cells, 65 topologies for 6 cells, 199 topologies for 7 cells, 653 topologies for 8 cells, 2283 topologies for 9 cells, and 8467 topologies for 10 cells. For example, the computational time for 10 cells was 324 [s] by MATLAB running on a QuadCore 2.83GHz processor. The number of topologies and the computation time for generating all of the topologies increases exponentially with the number of cells.

A similar process can be used to autogenerate topologies based on the incidence matrices which

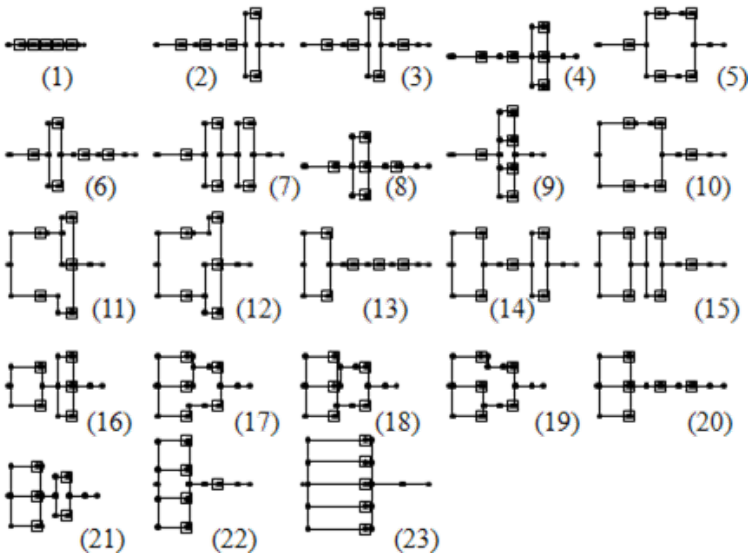


Figure 14: Automatically generated 23 topologies for 5 cells.

is not restricted to the layer based arrays, however that method produces many more topologies. The layer based array structure gives good coverage of possible structures while limiting results to arrays that generally avoid compressive internal forces. For this reason, the incidence matrix autogeneration process was not specifically covered in this thesis.

2.5 Incidence Matrix Identity and Similarity Transforms

When comparing two actuator array topologies, whether automatically generated or otherwise, it can be helpful to have a criteria to determine if two topologies are identical. If two topologies are not identical, it can be likewise helpful to determine how similar they are from a topological perspective. Identical topologies are defined as those with the same topological configuration but different numerical labels for the masses and cells. While a mirrored actuator array will contain the same static properties as the original, the forces at the endpoints can different dynamically thus the arrays are not considered identical. The same is true for mirrored sub-sections of an array. While two identical arrays may share the same graphical layout, the incidence matrices are different which complicates the identification. This section gives a method for identifying identical arrays.

This section also develops several transformations which can be used to judge the similarity of two array topologies. Unfortunately, due to the discontinuous nature of the solution space for

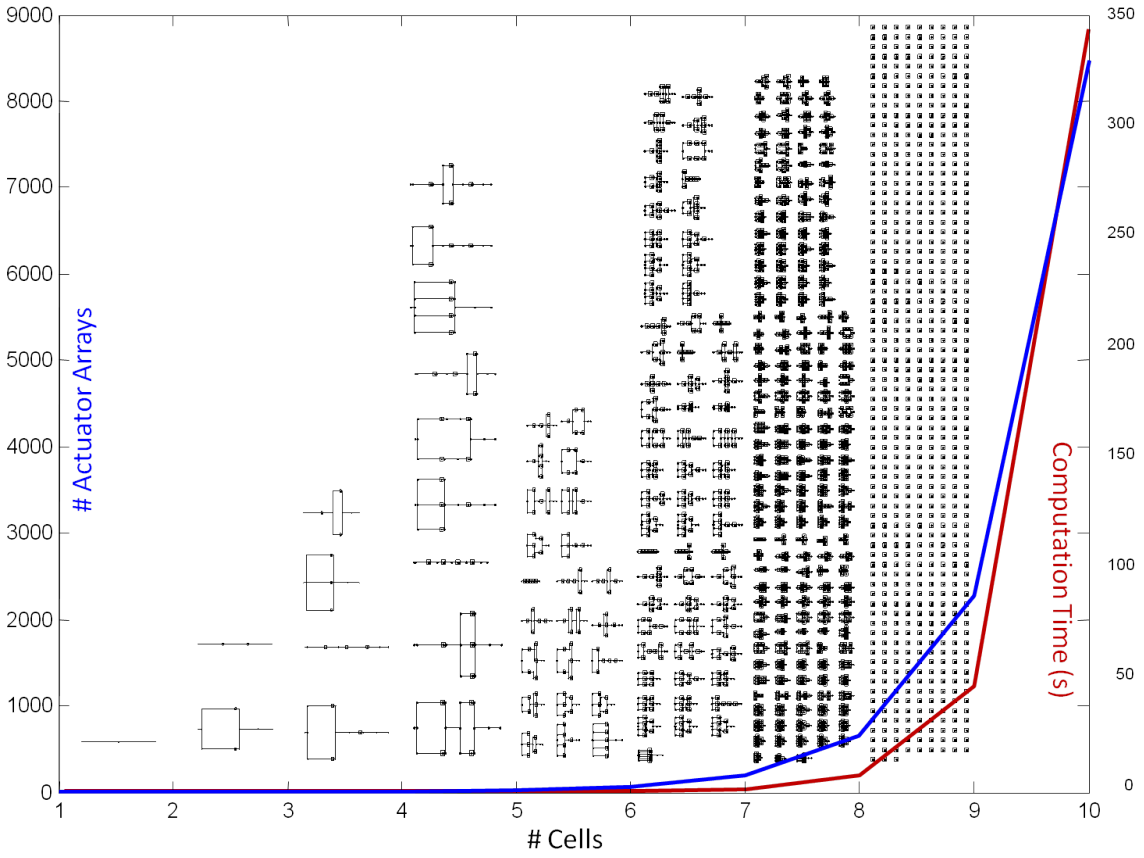


Figure 15: Automatically generated topologies and computational effort for 1-9 cells.

properties of arrays, these criteria do not always show similarity of the properties of the arrays. This is a limitation of the current methods and could be addressed in future work to develop methods of judging similarity of properties from the incidence matrices directly.

2.5.1 Incidence Matrix Identity Transforms

Two identical arrays will always contain the same number of cells and masses, thus their G (and likewise H) incidence matrices must have identical dimensions. If this criteria is not met, it is not possible to transform one into the other. Assuming the criteria is met, two transforms are available which when applied to any $\mathbf{G}$ and $\mathbf{H}$ incidence matrix pair will yield an identical topology.

Transformation 1 (T_m) switches mass a and mass b by switching rows a and b in $\mathbf{G}$ and columns a and b in $\mathbf{H}$ as shown in (6). This works due because of the mapping $\mathbf{G}$ make from masses to cells and that $\mathbf{H}$ makes from cells to masses. The multiplication of $\mathbf{G}$ and $\mathbf{H}$, $\mathbf{GH}$, has both its rows and

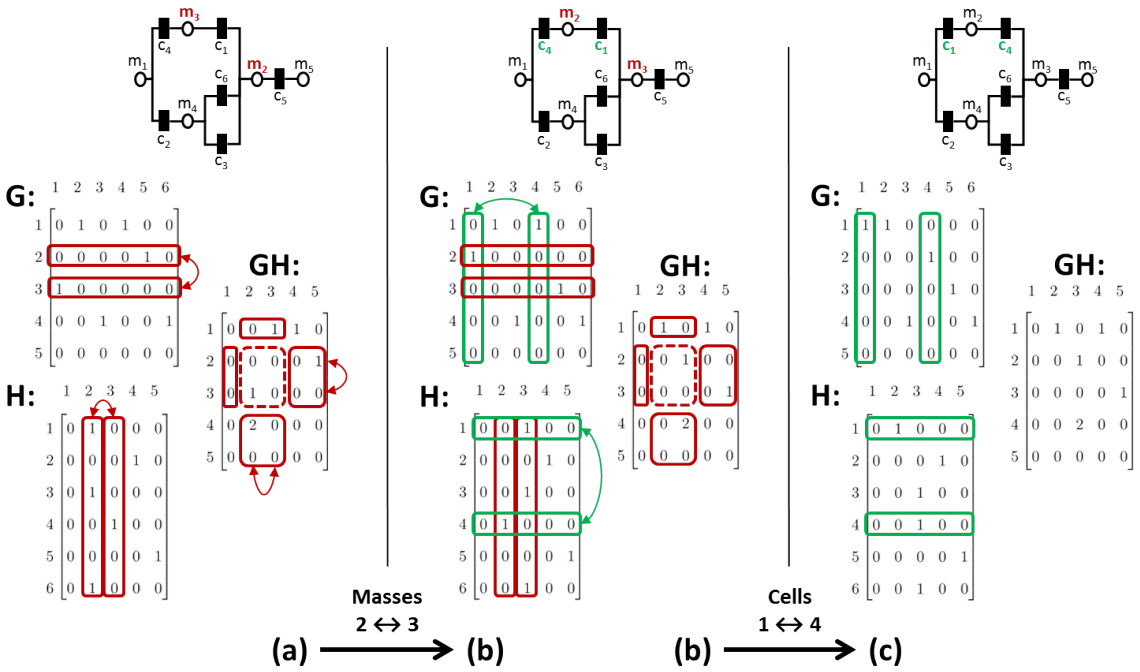


Figure 16: Example transitions between identical topologies.

columns swapped. This is shown in Fig. 16 as the transformation between frame (a) and frame (b).

$$\begin{aligned} \mathbf{G}_{(a,:)} &\Rightarrow \mathbf{G}^{new}_{(b,:)} & \mathbf{H}_{(:,a)} &\Rightarrow \mathbf{H}^{new}_{(:,b)} \\ \mathbf{G}_{(b,:)} &\Rightarrow \mathbf{G}^{new}_{(a,:)} & \mathbf{H}_{(:,b)} &\Rightarrow \mathbf{H}^{new}_{(:,a)} \\ \mathbf{G}_{(i,:)} &\Rightarrow \mathbf{G}^{new}_{(i,:)},\ \forall i \neq \{a,b\} & \mathbf{H}_{(:,i)} &\Rightarrow \mathbf{H}^{new}_{(:,i)},\ \forall i \neq \{a,b\} \end{aligned} \tag{6}$$

Transformation 2 (T_c) switches cell a and cell b by switching columns a and b in $\mathbf{G}$ and rows a and b in $\mathbf{H}$ as shown in (7). This works by the same reasoning as criteria 1. The multiplication of $\mathbf{G}$ and $\mathbf{H}$, $\mathbf{GH}$, is unchanged as it maps from masses to mass. This is shown in Fig. 16 as the transformation between frame (b) and frame (c).

$$\begin{aligned} \mathbf{G}_{(:,a)} &\Rightarrow \mathbf{G}^{new}_{(:,b)} & \mathbf{H}_{(a,:)} &\Rightarrow \mathbf{H}^{new}_{(b,:)} \\ \mathbf{G}_{(:,b)} &\Rightarrow \mathbf{G}^{new}_{(:,a)} & \mathbf{H}_{(b,:)} &\Rightarrow \mathbf{H}^{new}_{(a,:)} \\ \mathbf{G}_{(:,i)} &\Rightarrow \mathbf{G}^{new}_{(:,i)},\ \forall i \neq \{a,b\} & \mathbf{H}_{(i,:)} &\Rightarrow \mathbf{H}^{new}_{(i,:)},\ \forall i \neq \{a,b\} \end{aligned} \tag{7}$$

Two array topologies are identical if a sequence of T_m and T_c can be found to move from one topology to another. The formal proof of this is left to future work, though no counter example has

been found.

2.5.2 Incidence Matrix Similarity Transforms

2.5.2.1 Mirror Transform

The first similarity transform works on the idea of mirroring an array. Since $\mathbf{G}$ is a mapping from masses to cells and $\mathbf{H}$ is a mapping from cells to masses, the two can be reversed to provide the mirrored mapping. The transform first mirrors $\mathbf{G}$ and $\mathbf{H}$ about the vertical axis and then takes the transpose of the result.

The same process can be done for sub-arrays so long as the number of masses connecting directly to the sub-array on either side of the sub-array are consistent. This is helpful to show relationships between array topologies but does not imply static property similarity unless only a single mass connects to either side of the sub-array. Mirror transforms do not imply dynamic property similarity regardless of number of masses on either side. The exception to this is when the array, or sub-array, is symmetric though this is not covered directly in the current work.

The sub-array mirror transform begins by using transforms 1 and 2 to group all of the elements in the sub-array together in both $\mathbf{G}$ and $\mathbf{H}$. Once grouped, the subsection $\mathbf{G}_{([a:b],[c:d])}$ will contain rows a through b and columns c through d of $\mathbf{G}$. Likewise, the subsection $\mathbf{H}_{([c:d],[a:b])}$ will contain rows c through d and columns a through b of $\mathbf{H}$. The two subsections will contain the sub-array and can be transformed by mirroring across the vertical axis and then taking the transpose. This is shown in (8). The formal proof of the mirror transforms is again left to future work.

$$\begin{gathered}\left[\mathbf{G}_{([a:b],[d:c])}\right]^{\top} \Rightarrow \mathbf{H}^{new}_{([c:d],[a:b])} \\ \mathbf{G}_{(i,j)} \Rightarrow \mathbf{G}^{new}_{(i,j)}, \forall[i<a],[i>b],[j<c],[j>d] \\ \\ \left[\mathbf{H}_{([c:d],[b:a])}\right]^{\top} \Rightarrow \mathbf{G}^{new}_{([a:b],[c:d])} \\ \mathbf{H}_{(i,j)} \Rightarrow \mathbf{H}^{new}_{(i,j)}, \forall[i<c],[i>d],[j<a],[j>b]\end{gathered} \tag{8}$$

2.5.2.2 Cell/Mass Addition Transform

Adding cells and masses to the $\mathbf{G}$ and $\mathbf{H}$ incidence matrices allows for judging the difference between a sub-array and a more complex full array. On its own this provides limited utility, however

similar array topologies generally share common sub-array roots. When a common root is found, the minimum number of steps (cell/mass addition and mirror transforms) required to move from the root to each full array provides a numerical similarity comparison between the two arrays.

Addition of elements from a common root provides a better measure than subtracting elements since subtracting an element can make entire sections of an array useless and create a premature root between two topologies. Future work could develop subtraction methods to determine robustness similarity between arrays, though that was not handled as part of this thesis.

Since cells connect to masses directly and any number of cells can connect to a single mass, adding a cell simply requires selecting the masses to connect to either side of the new cell. Once selected, a new column vector is appended to $\mathbf{G}$ with a 1 in the row corresponding to the mass connected on the left and 0 for all other elements. Similarly a new row vector is added to $\mathbf{H}$ with a 1 in the column corresponding to the mass connected on the right and 0 for all other elements. This is shown in (9).

$$\mathbf{g}_m = \begin{cases} 0 & \text{if mass } m \text{ is not connected to the new cell on the left} \\ 1 & \text{if mass } m \text{ is connected to the new cell on the left} \end{cases}$$
$$\begin{bmatrix} \mathbf{G} & \mathbf{g} \end{bmatrix} \Rightarrow \mathbf{G}^{new}$$
$$\mathbf{h}_m = \begin{cases} 0 & \text{if mass } m \text{ is not connected to the new cell on the right} \\ 1 & \text{if mass } m \text{ is connected to the new cell on the right} \end{cases} \tag{9}$$
$$\begin{bmatrix} \mathbf{H} \\ \mathbf{h} \end{bmatrix} \Rightarrow \mathbf{H}^{new}$$

Since masses connect to cells on either side and only a single mass can connect to any given side of a cell, the process for adding masses also requires adding additional cells. This is treated as a two-step process. First an empty row vector is appended to the top of $\mathbf{G}$ and an empty row vector is appended to the left side of $\mathbf{H}$. This creates the additional mass in the system which takes the place as the first mass in the array. The new mass is not yet connected to the rest of the array. The second step utilizes the cell addition transformation to add two cells to the array, one which has a

right side connection to the new mass and the other that has a left side connection to the new mass.

2.6 Conclusion

This chapter addressed the challenges of representing complex cellular actuator array topologies, exploring possible topologies given certain criteria, and determining the similarity between related topologies. The chapter began by first addressing the main components of an array, the cell and the mass, and restrictions on what types of actuators can be used in each cell of an array. The discussion then moved into representing arrays using a modified incidence matrix from graph theoretic modeling and introduced a reduced representation of well-ordered topologies called a fingerprint. The fingerprint was then used to deterministically identify possible array topologies so that they can be evaluated. The chapter closed by presenting a method for determining similarity between topologies which future work can use to search for topologies with desirable properties.

This chapter specifically contributed two incidence matrix representation for topologies which enables the dynamic equations of motion to be more easily generated than with traditional incidence matrices from graph theoretic modeling. A new compact mathematical representation of topologies called the 'fingerprint' was also created along with a method to automatically generate array topologies based on the 'fingerprint.' Finally, a mathematical process was then introduced which allows direct generation of the new incidence matrices directly from the 'fingerprint' and vice versa.

CHAPTER III

DYNAMIC MODELING

This chapter addresses the challenge of generating the dynamic equations of motion for varied cellular actuator array topologies with a method that accommodates any valid linear internal cell dynamics. This is needed in order to control and simulate the response of actuator arrays when applied to useful systems. The chapter begins by describing a Hill-Type model cell and detailing the equations of motion for that cell. This provides an example as a guide for the following sections which develop the dynamic equations of motion for any generalized actuator array with valid linear cell dynamics. The process works much like a plug-and-chug method starting from the incidence matrices from the previous chapter and ending with the dynamic equations in a standardized linear form. An example is then presented to clarify the process for a small Hill-Type model actuator array. Numerical results are presented following the example to show the validity of the presented method compared to MatLab's SimMechanics toolbox, a dynamic modeling suite. Finally, the proof and derivation of the presented method is given to show its validity theoretically.

3.1 Hill-Type Cell Modeling

A real and useful version of a Hill-Type cell model Fig. 7 is the damped Hill-Type cell model shown in Fig. 17. This is used to represent a physical damped shape memory alloy based actuator array in the experimental chapter. The justification behind the design is presented there. While other Hill-Type cell models exist, the explanations and examples throughout this chapter refer to this damped Hill-Type model (called Hill-Type model).

The cells in this Hill-Type model can be represented in state-space form by choosing the states to be the positions (p_m) and velocities ($\dot{p}_m$) of each mass in the array (m) plus and the internal states (ω_q^n) of each cell (n) in the array. The subscript q for ω_q^n refers to the internal state number for the cell - for which there is only one in the Hill-Type cell model - while the superscript refers to a particular cell. Equation (10) represents the tension force carried across cell n, the cumulative force carried by the passive elastic element and the contractile element. This is the force exerted

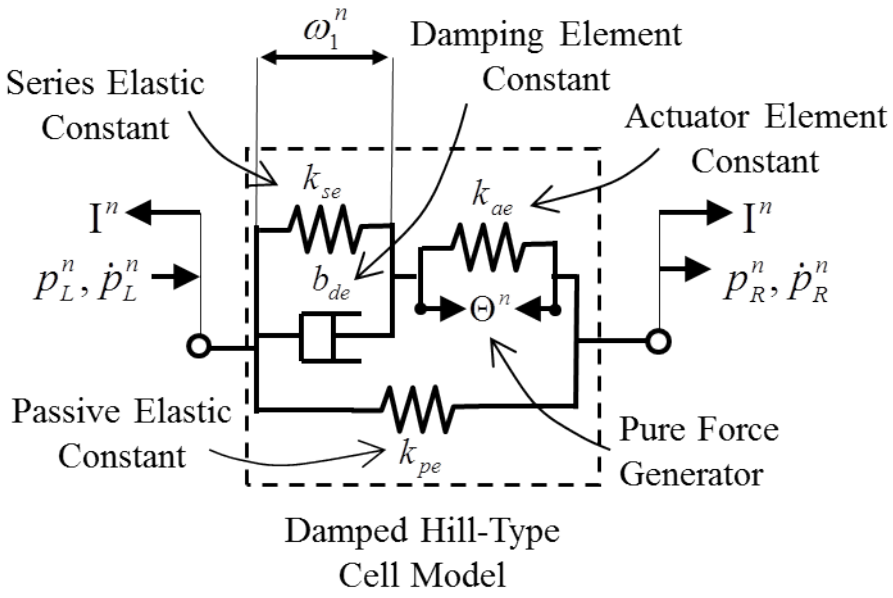


Figure 17: Damped Hill-Type Cell Model.

on the mass elements that cell n connects to. The mass of the connection between the series elastic and contractile elements are considered negligible, therefore the force carried by each end of the cell is equal. The force carried across each elastic element is the spring constant (k) multiplied by the displacement of the element, and the force carried across the damping element is the damping coefficient (b) multiplied by the derivative of the displacement of the element.

$$I^n = k_{pe} \cdot (p_R^n - p_L^n) + k_{ae} \cdot (\omega_1^n) + \theta^n \tag{10}$$

Equation (11) shows the force balance of the upper portion of the Hill-Type cell model for cell n. Here the cumulative force carried by the series elastic element and the damper element is the same as force carried by the contractile element.

$$k_{se} \cdot (p_R^n - p_L^n - \omega_1^n) + b_{de} \cdot (\dot{p}_R^n - \dot{p}_L^n - \dot{\omega}_1^n) = k_{ae} \cdot (\omega_1^n) + \theta^n \tag{11}$$

In both equations, p_L^n is the position of the mass connected to the left of cell n and p_R^n is the position of the mass connected to the right of cell n. ω_1^n is the length of cell n's contractile element,

and $\dot{\omega}_1^n$ is the time derivative of ω_1^n. k_{se}, k_{pe}, and k_{ae} are the spring constants of the series elastic, parallel elastic, and actuator elements respectively. b_{de} is the damping coefficient of the damper and θ^n is the pure force due to activating the actuator. θ^n is zero when the cell is inactive. Electrical dynamics are assumed to act significantly faster than the physical dynamics of the cell and are thus not modeled. Fig. 17 shows the elements and associated variables and can be used as a reference to verify equations (10) and (11).

Equation (11) can be rearranged to solve for $\dot{\omega}_1^n$ as a function of the state variables, as shown in (12). $\dot{\omega}_1^n$ shows the change in the contractile element length, an essential part of the final dynamics.

$$\dot{\omega}_1^n = \tfrac{k_{se}}{b_{de}} \cdot (p_R^n - p_L^n) - \tfrac{k_{se}+k_{ae}}{b_{de}} \cdot (\omega_1^n) - \tfrac{1}{b_{de}} \cdot (\theta^n) + (\dot{p}_R^n - \dot{p}_L^n) \tag{12}$$

Since masses are only connected to cells, a force balance for each mass contains only forces from cells on either side of the mass. Cells on the left side pull the mass in the negative direction while those on the right pull in the positive direction. Equation (13) shows the resultant acceleration for mass m with list of cells ℓ_L^m connected to the left and list of cells ℓ_R^m connected to the right. The mass of mass m is ϕ_m.

$$\begin{aligned}
\ddot{p}_m = &-\frac{k_{pe}}{\phi_m} \cdot \left[\sum_{n=\ell_L^m} 1 + \sum_{n=\ell_R^m} 1 \right] \cdot p_m \\
&+\frac{k_{pe}}{\phi_m} \cdot \left[\sum_{n=\ell_L^m} p_L^n + \sum_{n=\ell_R^m} p_R^n \right] \\
&-\frac{k_{ae}}{\phi_m} \cdot \left[\sum_{n=\ell_L^m} \omega_1^n - \sum_{n=\ell_R^m} \omega_1^n \right] \\
&-\frac{1}{\phi_m} \cdot \left[\sum_{n=\ell_L^m} \theta^n - \sum_{n=\ell_R^m} \theta^n \right]
\end{aligned} \tag{13}$$

Using mass 6 in Fig. 10 as an example, the summations in (13) would be (14).

$$
\begin{aligned}
\ell_L^m &= [7,8] & \ell_R^m &= [12,13] \\
\sum_{n=\ell_L^m} 1 &= 2 & \sum_{n=\ell_R^m} 1 &= 2 \\
\sum_{n=\ell_L^m} p_L^n &= p_L^7 + p_L^8 = p_3 + p_4 & \sum_{n=\ell_R^m} p_R^n &= p_R^{12} + p_R^{13} = 2 \cdot p_8 \\
\sum_{n=\ell_L^m} \omega_1^n &= \omega_1^7 + \omega_1^8 & \sum_{n=\ell_R^m} \omega_1^n &= \omega_1^{12} + \omega_1^{13}
\end{aligned} \tag{14}
$$

The trivial equation (15) completes the needed state space equations.

$$
\dot{p}_m = \frac{\mathrm{d}}{\mathrm{dt}} p_m \tag{15}
$$

3.2 General Linear Dynamic Cell Equations

The standard linear state-space form given by (16) allows for using standard controllability, stability, etc. analysis techniques. This work provides a direct method to generate the standard form from any array topology and linear internal cell dynamics - not just the Hill-Type cell model dynamics - given that cells connect solely to masses (thus the only external states needed for internal cell dynamics are the mass positions and velocities). The cells treated in this work are intended to contain mechanical elements, though future work could extend the theory to include elements from electrical or thermal domains. The linear restriction will be relaxed to include well-structured non-linear cell dynamics in chapter 5.

$$
\begin{aligned}
\dot{\mathbf{X}} &= \mathbf{A} \cdot \mathbf{X} + \mathbf{B} \cdot \mathbf{U} \\
\mathbf{F} &= \mathbf{C} \cdot \mathbf{X} + \mathbf{D} \cdot \mathbf{U}
\end{aligned} \tag{16}
$$

The linear internal cell dynamics for any cell n which follows the criteria in the cell and topology sections of chapter 2 can be represented in generalized form by (17), (18), and (19). (17) represents the force to the left of the cell, (18) represents the force to the right of the cell, and (19) represents the change in cell internal states. These functions are dependent only on the position of the masses to the left and right of the cell (p_L^n and p_R^n), the velocities of the masses to the left and right of

the cell ($\dot{p}_L^n$ and $\dot{p}_R^n$), and the cell's internal states (ω_q^n). The superscript n defines which cell the variable or equation refers to, subscripts L and R denote left and right, and subscript q refers to the cell internal state q (with Q total). Tables 1 and 2 define the variables more succinctly for reference.

$$\begin{aligned} E_L^n =& \text{coe}(p_L)_L^n \cdot \{p_L^n\} + \text{coe}(p_R)_L^n \cdot \{p_R^n\} + \text{coe}(\dot{p}_L)_L^n \cdot \{\dot{p}_L^n\} + \text{coe}(\dot{p}_R)_L^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\text{coe}(\theta_v)_L^n \cdot \{\theta_v\}\right) + \sum_{q=1}^{Q} \left(\text{coe}(w_q)_L^n \cdot \left\{w_q^n\right\}\right) \end{aligned} \tag{17}$$

$$\begin{aligned} E_R^n =& \text{coe}(p_L)_R^n \cdot \{p_L^n\} + \text{coe}(p_R)_R^n \cdot \{p_R^n\} + \text{coe}(\dot{p}_L)_R^n \cdot \{\dot{p}_L^n\} + \text{coe}(\dot{p}_R)_R^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\text{coe}(\theta_v)_R^n \cdot \{\theta_v\}\right) + \sum_{q=1}^{Q} \left(\text{coe}(w_q)_R^n \cdot \left\{w_q^n\right\}\right) \end{aligned} \tag{18}$$

$$\begin{aligned} \dot{w}_q^n =& \text{cow}(p_L)_q^n \cdot \{p_L^n\} + \text{cow}(p_R)_q^n \cdot \{p_R^n\} + \text{cow}(\dot{p}_L)_q^n \cdot \{\dot{p}_L^n\} + \text{cow}(\dot{p}_R)_q^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\text{cow}(\theta_v)_q^n \cdot \{\theta_v\}\right) + \sum_{i=1}^{Q} \left(\text{cow}(w_i)_q^n \cdot \{w_i^n\}\right) \end{aligned} \tag{19}$$

$\text{coe}(y)_L$, $\text{coe}(y)_R$, and $\text{cow}(y)_q$ are coefficient matrix functions for variable y and contain the system dynamics information for the cells in the array. Substitute the proper variable (p_L, p_R, θ_v, ω_q, ...) for y. All coefficient matrices except $\text{coe}(\theta_v)_L$, $\text{coe}(\theta_v)_R$, $\text{cow}(\theta_v)_L$, and $\text{cow}(\theta_v)_R$ are square nxn matrices and adding superscript n refers to the term on the nth row/column. When all cells in an array are identical, all terms in the matrix are also identical. (20) shows $\text{coe}(p_L)_L$ as an illustrative example.

$$\text{coe}(p_L)_L = \begin{bmatrix} \text{coe}(p_L)_L^1 & 0 & 0 \\ 0 & \text{coe}(p_L)_L^n & 0 \\ 0 & 0 & \text{coe}(p_L)_L^N \end{bmatrix} \tag{20}$$

$\text{coe}(\theta_v)_L$, $\text{coe}(\theta_v)_R$, $\text{cow}(\theta_v)_L$, and $\text{cow}(\theta_v)_R$ - the coefficient matrices for the inputs - work differently as they are dependent on the connections of the cells to the inputs of the physical system. (101) shows $\text{coe}(\theta_v)_L$ as an illustrative example. In these matrices, row n refers to cell n while

Table 1: Cell Reference Variables

Cell n Reference Variables	**Symbol**
Left Endpoint Position	$\{p_L^n\}$
Right Endpoint Position	$\{p_R^n\}$
Left Endpoint Velocity	$\{\dot{p}_L^n\}$
Right Endpoint Velocity	$\{\dot{p}_R^n\}$
Cell Input(s)	$\{\theta_v\}$
State q Internal Value	$\{w_q^n\}$
Left Side Output Force	$\{E_L^n\}$
Right Side Output Force	$\{E_R^n\}$

Table 2: Cell Effect Variables

Effects on Cell n Internal Variables	**Symbol**
Effect of Left Endpoint Position	$\text{cow}(p_L)_q^n \cdot \{p_L^n\}$
Effect of Right Endpoint Position	$\text{cow}(p_R)_q^n \cdot \{p_R^n\}$
Effect of Left Endpoint Velocity	$\text{cow}(\dot{p}_L)_q^n \cdot \{\dot{p}_L^n\}$
Effect of Right Endpoint Velocity	$\text{cow}(\dot{p}_R)_q^n \cdot \{\dot{p}_R^n\}$
Effect of Cell Input(s)	$\sum_{v=1}^{V} \left(\text{cow}(\theta_v)_q \cdot \{\theta_v\}\right)$
Effect of Internal Variables	$\sum_{i=1}^{Q} \left(\text{cow}(w_i)_q^n \cdot \{w_i^n\}\right)$

column v refers to input v. If cell n is not connected to input v, coefficient cell (n,v) will be 0. Otherwise it will contain the coefficient needed to properly represent the dynamic effect of the input on the cell's out force or internal state. (21) shows $\text{coe}(\theta_v)_L$ as an illustrative example.

$$\text{coe}\left(\boldsymbol{\Theta}\right)_L = \begin{bmatrix} \text{coe}(\theta_1)_L^1 & \dots & \text{coe}(\theta_v)_L^1 & \dots & \text{coe}(\theta_V)_L^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_L^n & 0 & \text{coe}(\theta_v)_L^n & 0 & \text{coe}(\theta_V)_L^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_L^N & 0 & \text{coe}(\theta_v)_L^N & 0 & \text{coe}(\theta_V)_L^N \end{bmatrix} \tag{21}$$

Tables 1 and 2 and the List of Symbols at the beginning of the thesis provide references for the variables used in array modeling.

For the sake of consistency, the sign convention in this work holds that right is positive for force, position, and velocity for all array elements. As such elements of $\text{coe}(p_L)_L$, $\text{coe}(p_R)_R$, $\text{coe}(\dot{p}_L)_L$, and $\text{coe}(\dot{p}_R)_R$ are generally negative while elements of $\text{coe}(p_R)_L$, $\text{coe}(p_L)_R$, $\text{coe}(\dot{p}_R)_L$, and $\text{coe}(\dot{p}_L)_R$ are generally positive.

3.3 State-Space State Vector and Input Vector

In order to represent the cellular actuator array equations of motion in the state-space form given by (16) there first needs to exist a convention for the state vector and input vector. In order to simplify calculations, the state vector was chosen according to (22). Here $\mathbf{X}$ is the state vector, $\mathbf{P}$ contains the positions of the $M-2$ masses [p_2 ... $p_{(M-1)}$] internal to the array, $\dot{\mathbf{P}}$ contains the velocities of the masses [$\dot{p}_2$ ... $\dot{p}_{(M-1)}$], and $\mathbf{J}$ contains the internal state variables for all of the cells. $\mathbf{P}$ and $\dot{\mathbf{P}}$ are combined in $\mathbf{P_v}$ to simplify later explanation. The endpoints of the array are treated as inputs instead of states; thus p_1 is z_L, p_M is z_R, $\dot{p}_1$ is $\dot{z}_L$, and $\dot{p}_M$ is $\dot{z}_R$. $\mathbf{J}$ can be further broken down into Q sub vectors ($\mathbf{J}_q$), one for each internal state of the cells. Each sub vector can finally be broken down into N elements, one for each cell in the array. For arrays with non-uniform cells some elements of higher number sub vectors (higher q) may contain 0's since not all cells require the same Q total internal states as the cell with the highest number of states. Removal of these unused states is left to post-processing as the added computational complexity in generating the system system of equations with the unused states is negligible. The internal cell state for internal variable q and cell n would is ω_q^n.

$$\mathbf{X} = \begin{bmatrix} \mathbf{P}_v \\ \mathbf{J} \end{bmatrix}, \quad \mathbf{P_v} = \begin{bmatrix} \mathbf{P} \\ \dot{\mathbf{P}} \end{bmatrix} = \begin{bmatrix} p_2 \\ \vdots \\ p_{(M-1)} \\ \dot{p}_2 \\ \vdots \\ \dot{p}_{(M-1)} \end{bmatrix}, \quad \mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_q \\ \vdots \\ \mathbf{J}_Q \end{bmatrix}, \quad \mathbf{J}_q = \begin{bmatrix} w_q^1 \\ \vdots \\ w_q^n \\ \vdots \\ w_q^N \end{bmatrix} \tag{22}$$

Input vector $\mathbf{U}$ was chosen according to (23) where $\mathbf{U}$ is the input vector, $\mathbf{Z}$ is the position input, $\dot{\mathbf{Z}}$ is the velocity input, and $\mathbf{\Theta}$ contains the control inputs to the cells. $\mathbf{Z}$ contains the positions of the left and right endpoints of the array (z_L and z_R), and $\dot{\mathbf{Z}}$ likewise contains the velocities of the left and right endpoints of the array ($\dot{z}_L$ and $\dot{z}_R$). $\mathbf{Z}$ and $\dot{\mathbf{Z}}$ are treated as inputs. Additional outside dynamics are needed to fully represent arrays with any physical load attached, and these outside dynamics are treated as input/output functions taking in array endpoint force and returning an updated array endpoint position and velocity. Outside dynamics are treated later in this chapter to further addresses

the combination of outside dynamics and array dynamics into full system dynamics. It should also be noted that isometric contraction cases are fully represented by the developed system of equations when $\mathbf{Z}$ and $\dot{\mathbf{Z}}$ are set to 0.

$$\mathbf{U} = \begin{bmatrix} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \boldsymbol{\Theta} \end{bmatrix}, \begin{bmatrix} \mathbf{Z} \\ \dot{\mathbf{Z}} \end{bmatrix} = \begin{bmatrix} z_L \\ z_R \\ \dot{z}_L \\ \dot{z}_R \end{bmatrix}, \boldsymbol{\Theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_v \\ \vdots \\ \theta_V \end{bmatrix} \tag{23}$$

Tab. 3 and Tab. 4 define the state variables and input variables more succinctly for reference.

3.4 General Dynamic System Modeling Method

The state-space equations given by (16) can be broken down into (24), (25), (26), (27) based on type of state in the state vector and input vector.

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \\ \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s & \boldsymbol{\iota} \end{bmatrix} \tag{24}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{\mathbf{Z}} & \mathbf{B}_{\boldsymbol{\Theta}} \end{bmatrix} \tag{25}$$

$$\mathbf{C} = \begin{bmatrix} \boldsymbol{\alpha}_b & \boldsymbol{\beta}_b & \boldsymbol{\gamma}_b \\ \boldsymbol{\alpha}_c & \boldsymbol{\beta}_c & \boldsymbol{\gamma}_c \end{bmatrix} \tag{26}$$

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{\mathbf{Z}} & \mathbf{D}_{\boldsymbol{\Theta}} \end{bmatrix} \tag{27}$$

(25) can be further broken down into (28) and (29) to separate array endpoint inputs from cell control inputs. Likewise (27) can be further broken down into (30) and (31).

Table 3: State Vector Variables

State Vector Variables	Symbol
State	$\mathbf{X} = \begin{bmatrix} \mathbf{P} \\ \dot{\mathbf{P}} \\ \mathbf{J} \end{bmatrix}$
Array Mass Positions	$\mathbf{P} = \begin{bmatrix} p_2 \\ \vdots \\ p_m \\ \vdots \\ p_{(M-1)} \end{bmatrix}$
Array Mass Velocities	$\dot{\mathbf{P}} = \begin{bmatrix} \dot{p}_2 \\ \vdots \\ \dot{p}_m \\ \vdots \\ \dot{p}_{(M-1)} \end{bmatrix}$
# Masses	M
Internal States	$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_q \\ \vdots \\ \mathbf{J}_Q \end{bmatrix}$
# Internal States	Q
State q Cell Values	$\mathbf{J}_q = \begin{bmatrix} w_q^1 \\ \vdots \\ w_q^n \\ \vdots \\ w_q^N \end{bmatrix}$
# Cells	N

$$\mathbf{B_Z} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d \\ \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d \end{bmatrix} \tag{28}$$

$$\mathbf{B_\Theta} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\delta}_s \\ \boldsymbol{\kappa} \end{bmatrix} \tag{29}$$

Table 4: Input Vector Variables

Input Vector Variables	Symbol
Array Input Vector	$\mathbf{U} = \begin{bmatrix} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \boldsymbol{\Theta} \end{bmatrix}$
Array Endpoint Positions	$\mathbf{Z} = \begin{bmatrix} z_L \\ z_R \end{bmatrix}$
Array Endpoint Velocities	$\dot{\mathbf{Z}} = \begin{bmatrix} \dot{z}_L \\ \dot{z}_R \end{bmatrix}$
Cell Inputs	$\boldsymbol{\Theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_v \\ \vdots \\ \theta_V \end{bmatrix}$
# Cell Inputs	V

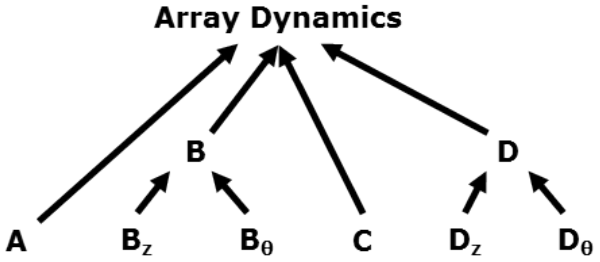


Figure 18: Graphical representation of the process to generate the dynamic equations of motion for an actuator array.

$$\mathbf{D_Z} = \begin{bmatrix} \alpha_{ab} & \alpha_{bd} & \beta_{ab} & \beta_{bd} \\ \alpha_{ac} & \alpha_{cd} & \beta_{ac} & \beta_{cd} \end{bmatrix} \tag{30}$$

$$\mathbf{D_\Theta} = \begin{bmatrix} \boldsymbol{\delta}_b \\ \boldsymbol{\delta}_c \end{bmatrix} \tag{31}$$

The final system dynamics can then be written as (32) and the output equations can be written as (33). Fig. 18 shows this process graphically and Tab. 5 shows the equation used to define each variable.

Table 5: Modeling Equation Guide.

Variable	Defining Equation	Variable	Defining Equation
Complete Dynamic System	(128) , (129)	$\boldsymbol{\alpha}$	(34) , (37)
Array Dynamics	(16) , (32) , (33)	$\boldsymbol{\beta}$	(34) , (38)
Outside Dynamics	(126), (127)	$\boldsymbol{\gamma}$	(35) , (41) , (42)
$\mathbf{A}$	(24)	$\boldsymbol{\delta}$	(35) , (43)
$\mathbf{B}$	(25)	$\boldsymbol{\zeta}$	(36) , (44) , (46)
$\mathbf{B}_Z$	(28)	$\boldsymbol{\eta}$	(36) , (45) , (46)
$\mathbf{C}$	(26)	$\boldsymbol{\iota}$	(47) , (49)
$\mathbf{D}$	(27)	$\boldsymbol{\kappa}$	(48) , (49)
$\mathbf{D}_Z$	(30)	$\mathbf{G}$	(4)
$\mathbf{D}_\Theta$	(31)	$\mathbf{H}$	(4)

$$\left[\begin{array}{c} \dot{\mathbf{P}}_v \\ \hline \dot{\mathbf{J}} \end{array}\right] = \left[\begin{array}{cc|c} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s & \boldsymbol{\iota} \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \\ \hline \mathbf{J} \end{array}\right] + \left[\begin{array}{cccc|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d & \boldsymbol{\delta}_s \\ \hline \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d & \boldsymbol{\kappa} \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta} \end{array}\right] \tag{32}$$

$$\left[\begin{array}{c} \mathbf{F} \end{array}\right] = \left[\begin{array}{ccc} \boldsymbol{\alpha}_b & \boldsymbol{\beta}_b & \boldsymbol{\gamma}_b \\ \boldsymbol{\alpha}_c & \boldsymbol{\beta}_c & \boldsymbol{\gamma}_c \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \\ \hline \mathbf{J} \end{array}\right] + \left[\begin{array}{cccc|c} \alpha_{ab} & \alpha_{bd} & \beta_{ab} & \beta_{bd} & \boldsymbol{\delta}_b \\ \alpha_{ac} & \alpha_{cd} & \beta_{ac} & \beta_{cd} & \boldsymbol{\delta}_c \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta} \end{array}\right] \tag{33}$$

Based on these definitions $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\delta}$ contain the dynamics information needed to determine the accelerations of the array masses and forces at the array endpoints. $\boldsymbol{\alpha}$ deals with the effects of mass locations, $\boldsymbol{\beta}$ deals with the effects of mass velocities, $\boldsymbol{\gamma}$ deals with the effects of cell internal states, and $\boldsymbol{\delta}$ deals with the effects of array control inputs. Similarly $\boldsymbol{\zeta}$, $\boldsymbol{\eta}$, $\boldsymbol{\iota}$, and $\boldsymbol{\kappa}$ contain the dynamics information needed to determine changes to the internal cell states of the array. $\boldsymbol{\zeta}$ deals with the effects of mass locations, $\boldsymbol{\eta}$ deals with the effects of mass velocities, $\boldsymbol{\iota}$ deals with the effects of cell internal states, and $\boldsymbol{\kappa}$ deals with the effects of array control inputs.

p_1, p_M, $\dot{p}_1$, and $\dot{p}_M$ are the endpoints of the array, not masses in the array. In order to separate endpoint dynamics from state dynamics, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ must be split according to (34) before being used in the dynamic equations.

$$\boldsymbol{\alpha} = \left[\begin{array}{c|c|c} \alpha_{ab} & \boldsymbol{\alpha}_b & \alpha_{bd} \\ \hline \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_s & \boldsymbol{\alpha}_d \\ \hline \alpha_{ac} & \boldsymbol{\alpha}_c & \alpha_{cd} \end{array}\right] = \left[\begin{array}{c|c|c} 1\text{x}1 & 1\text{x}(M-2) & 1\text{x}1 \\ \hline (M-2)\text{x}1 & (M-2)\text{x}(M-2) & (M-2)\text{x}1 \\ \hline 1\text{x}1 & 1\text{x}(M-2) & 1\text{x}1 \end{array}\right] \tag{34}$$

$$\boldsymbol{\beta} = \left[\begin{array}{c|c|c} \beta_{ab} & \boldsymbol{\beta}_b & \beta_{bd} \\ \hline \boldsymbol{\beta}_a & \boldsymbol{\beta}_s & \boldsymbol{\beta}_d \\ \hline \beta_{ac} & \boldsymbol{\beta}_c & \beta_{cd} \end{array}\right] = \left[\begin{array}{c|c|c} 1\text{x}1 & 1\text{x}(M-2) & 1\text{x}1 \\ \hline (M-2)\text{x}1 & (M-2)\text{x}(M-2) & (M-2)\text{x}1 \\ \hline 1\text{x}1 & 1\text{x}(M-2) & 1\text{x}1 \end{array}\right]$$

α_{ab}, α_{bd}, α_{ac}, α_{cd}, β_{ab}, β_{bd}, β_{ac}, and β_{cd} are all scaler values. α_{ab} holds the effect of the left endpoint position on left endpoint force, α_{bd} holds the effect of the right endpoint position on left endpoint force, α_{ac} holds the effect of the left endpoint position on right endpoint force, and α_{cd} holds the effect of the right endpoint position on right endpoint force. Similarly β_{ab} holds the effect of the left endpoint velocity on left endpoint force, β_{bd} holds the effect of the right endpoint velocity on left endpoint force, β_{ac} holds the effect of the left endpoint velocity on right endpoint force, and β_{cd} holds the effect of the right endpoint velocity on right endpoint force.

$\boldsymbol{\alpha}_a$, $\boldsymbol{\alpha}_b$, $\boldsymbol{\alpha}_c$, $\boldsymbol{\alpha}_d$, $\boldsymbol{\beta}_a$, $\boldsymbol{\beta}_b$, $\boldsymbol{\beta}_c$, and $\boldsymbol{\beta}_d$ are one dimensional column/row vectors. $\boldsymbol{\alpha}_a$ holds the effect of the left endpoint position on the forces exerted on the array masses, $\boldsymbol{\alpha}_b$ holds the effect of the array mass positions on the left endpoint force, $\boldsymbol{\alpha}_c$ holds the effect of the right endpoint position on the forces exerted on the array masses, and $\boldsymbol{\alpha}_d$ holds the effect of the array mass positions on the right endpoint force. Similarly $\boldsymbol{\beta}_a$ holds the effect of the left endpoint velocity on the forces exerted on the array masses, $\boldsymbol{\beta}_b$ holds the effect of the array mass velocities on the left endpoint force, $\boldsymbol{\beta}_c$ holds the effect of the right endpoint velocity on the forces exerted on the array masses, and $\boldsymbol{\beta}_d$ holds the effect of the array mass velocities on the right endpoint force. $\boldsymbol{\alpha}_s$ and $\boldsymbol{\beta}_s$ contain the remaining values of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. These contain the effects of the array mass position and velocities, respectively, on the forces exerted on the array masses.

Continuing to separate endpoint dynamics from state dynamics, $\boldsymbol{\gamma}_q$ and $\boldsymbol{\delta}$ need to be split according to (35).

$$\boldsymbol{\gamma} = \left[\begin{array}{c} \boldsymbol{\gamma}_b \\ \hline \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\gamma}_c \end{array}\right] = \left[\begin{array}{c} 1\text{x}Q \\ \hline (M-2)\text{x}Q \\ \hline 1\text{x}Q \end{array}\right]$$

$$\boldsymbol{\delta} = \left[\begin{array}{c} \boldsymbol{\delta}_b \\ \hline \boldsymbol{\delta}_s \\ \hline \boldsymbol{\delta}_c \end{array}\right] = \left[\begin{array}{c} 1\text{x}V \\ \hline (M-2)\text{x}V \\ \hline 1\text{x}V \end{array}\right] \tag{35}$$

$\boldsymbol{\gamma}_b$, $\boldsymbol{\gamma}_d$, $\boldsymbol{\delta}_b$, and $\boldsymbol{\delta}_d$ are one dimensional row vectors; $\boldsymbol{\gamma}_b$ holds the effects of cell internal states on left endpoint force, $\boldsymbol{\gamma}_d$ holds the effects of cell internal states on right endpoint force, $\boldsymbol{\delta}_b$ holds the effects of cell control inputs on left endpoint force, and $\boldsymbol{\delta}_d$ holds the effect of cell control inputs on right endpoint force. $\boldsymbol{\gamma}_s$ and $\boldsymbol{\delta}_s$ contain the remaining values of $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$ which are the effects of cell internal states and cell control inputs, respectively, on the forces exerted on the array masses.

For the last separation, $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$ need to be split according to (36).

$$\boldsymbol{\zeta} = \left[\begin{array}{c|c|c} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_s & \boldsymbol{\zeta}_d \end{array}\right] = \left[\begin{array}{c|c|c} (Q\cdot N)\text{x}1 & (Q\cdot N)\text{x}(M-2) & (Q\cdot N)\text{x}1 \end{array}\right]$$

$$\boldsymbol{\eta} = \left[\begin{array}{c|c|c} \boldsymbol{\eta}_a & \boldsymbol{\eta}_s & \boldsymbol{\eta}_d \end{array}\right] = \left[\begin{array}{c|c|c} (Q\cdot N)\text{x}1 & (Q\cdot N)\text{x}(M-2) & (Q\cdot N)\text{x}1 \end{array}\right] \tag{36}$$

$\boldsymbol{\zeta}_a$, $\boldsymbol{\zeta}_c$, $\boldsymbol{\eta}_a$, and $\boldsymbol{\eta}_c$ are one dimensional column vectors. $\boldsymbol{\zeta}_a$ holds the effects of the left endpoint position on the cell internal states, $\boldsymbol{\zeta}_c$ holds the effects of the right endpoint position on the cell internal states, $\boldsymbol{\eta}_a$ holds the effects of the left endpoint velocities on the cell internal states, and $\boldsymbol{\eta}_c$ holds the effects of the right endpoint velocities on the cell internal states. $\boldsymbol{\zeta}_S$ and $\boldsymbol{\eta}_S$ contain the remaining values of $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$ which are the effects of array mass positions and velocities, respectively, on the cell internal states.

$\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$, $\boldsymbol{\zeta}$, $\boldsymbol{\eta}$, $\boldsymbol{\iota}$, and $\boldsymbol{\kappa}$ can all be populated directly from the incidence matrices (G and H) and the coefficient functions. $\boldsymbol{\alpha}$ is populated according to (37) and $\boldsymbol{\beta}$ is populated according to (38).

$\boldsymbol{\Phi}$ is a diagonal matrix where each term is the inverse of the mass of the corresponding mass in the array. Defining each mass m in the array to have a mass of ϕ_m, $\boldsymbol{\Phi}$ is built according to (39).

This is used by $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\delta}$ to correctly scale the effects of forces on the mass elements of the array. ϕ_1 and ϕ_M are 1 since they affect the output of the system, a force instead of the effect of forces on a state.

$$\boldsymbol{\alpha} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \text{diag}\left(\left[\text{coe}(p_R)_R \cdot \mathbf{H} \right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \text{coe}(p_L)_L \cdot \boldsymbol{\mu} \right) \\ + \left[\mathbf{G} \cdot \text{coe}(p_L)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \text{coe}(p_R)_L \cdot \mathbf{H} \end{bmatrix} = \left[\begin{array}{c|c|c} \alpha_{ab} & \boldsymbol{\alpha}_b & \alpha_{bd} \\ \hline \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_s & \boldsymbol{\alpha}_d \\ \hline \alpha_{ac} & \boldsymbol{\alpha}_c & \alpha_{cd} \end{array}\right] \tag{37}$$

$$\boldsymbol{\beta} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \text{diag}\left(\left[\text{coe}(\dot{p}_R)_R \cdot \mathbf{H} \right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \text{coe}(\dot{p}_L)_L \cdot \boldsymbol{\mu} \right) \\ + \left[\mathbf{G} \cdot \text{coe}(\dot{p}_L)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \text{coe}(\dot{p}_R)_L \cdot \mathbf{H} \end{bmatrix} = \left[\begin{array}{c|c|c} \beta_{ab} & \boldsymbol{\beta}_b & \beta_{bd} \\ \hline \boldsymbol{\beta}_a & \boldsymbol{\beta}_s & \boldsymbol{\beta}_d \\ \hline \beta_{ac} & \boldsymbol{\beta}_c & \beta_{cd} \end{array}\right] \tag{38}$$

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{\phi_2} & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \frac{1}{\phi_m} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & \dots & \frac{1}{\phi_{(M-1)}} & 0 \\ 0 & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix} \tag{39}$$

Vector $\boldsymbol{\mu}$ can be defined as a column vector of length N with 1's in every element as shown in (40) and is used to aid in summations as shown in the proof below. Furthermore, function $\text{diag}(\mathbf{y})$ creates a diagonal matrix from the elements of column vector $\mathbf{y}$.

$$\mu_n = 1 \text{ for } n = 1...N \tag{40}$$

$\boldsymbol{\gamma}_q$ is populated according to (41) for each internal state q and then combined according to (42) to yield $\boldsymbol{\gamma}$.

$$\boldsymbol{\gamma}_q = \left[\left[\text{coe}(w_q)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \text{coe}(w_q)_L \right] \ , \ \forall q \tag{41}$$

$$\boldsymbol{\gamma} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \boldsymbol{\gamma}_1 & \dots & \boldsymbol{\gamma}_q & \dots & \boldsymbol{\gamma}_Q \end{bmatrix} = \begin{bmatrix} \boldsymbol{\gamma}_b \\ \hline \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\gamma}_c \end{bmatrix} \tag{42}$$

$\boldsymbol{\delta}$ is populated according to (43).

$$\boldsymbol{\delta} = \boldsymbol{\Phi} \cdot \left[\mathbf{H}^\top \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_R + \mathbf{G} \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_L \right] = \begin{bmatrix} \boldsymbol{\delta}_b \\ \hline \boldsymbol{\delta}_s \\ \hline \boldsymbol{\delta}_c \end{bmatrix} \tag{43}$$

$\boldsymbol{\zeta}_q$ and $\boldsymbol{\eta}_q$ are populated according to (44) and (45) respectively for each cell internal state q. They are then combined using (46) to yield $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$.

$$\boldsymbol{\zeta}_q = \left[\left[\mathbf{G} \cdot \text{cow}(p_L)_q \right]^\top + \text{cow}(p_R)_q \cdot \mathbf{H} \right] \ , \ \forall q \tag{44}$$

$$\boldsymbol{\eta}_q = \left[\left[\mathbf{G} \cdot \text{cow}(\dot{p}_L)_q \right]^\top + \text{cow}(\dot{p}_R)_q \cdot \mathbf{H} \right] \ , \ \forall q \tag{45}$$

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\zeta}_1 \\ \vdots \\ \boldsymbol{\zeta}_q \\ \vdots \\ \boldsymbol{\zeta}_Q \end{bmatrix} = \left[\begin{array}{c|c|c} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_s & \boldsymbol{\zeta}_d \end{array} \right] \qquad \boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \vdots \\ \boldsymbol{\eta}_q \\ \vdots \\ \boldsymbol{\eta}_Q \end{bmatrix} = \left[\begin{array}{c|c|c} \boldsymbol{\eta}_a & \boldsymbol{\eta}_s & \boldsymbol{\eta}_d \end{array} \right] \tag{46}$$

$\boldsymbol{\iota}_{i,q}$ is populated according to (47) for each combination of cell internal states (i,q). $\boldsymbol{\kappa}_q$ is populated according to (48) for each cell internal state q. They are then combined using (49) to yield $\boldsymbol{\iota}$ and $\boldsymbol{\kappa}$.

$$\boldsymbol{\iota}_{i,q} = \text{cow}(w_i)_q \ , \ \forall i, q \tag{47}$$

Table 6: General Array Variables

General Array Variables	Symbol
Array Force Output	$\mathbf{F} = \begin{bmatrix} F_L \\ F_R \end{bmatrix}$
Mass of Array Masses	$\mathbf{\Phi} = \begin{bmatrix} \phi_2 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & \phi_m & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & \phi_{(M-1)} \end{bmatrix}$

$$\boldsymbol{\kappa}_q = \text{cow}\left(\mathbf{\Theta}\right)_q \ , \ \forall q \tag{48}$$

$$\boldsymbol{\iota} = \begin{bmatrix} \boldsymbol{\iota}_{1,1} & \dots & \boldsymbol{\iota}_{1,q} & \dots & \boldsymbol{\iota}_{1,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{i,1} & \dots & \boldsymbol{\iota}_{i,q} & \dots & \boldsymbol{\iota}_{i,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{I,1} & \dots & \boldsymbol{\iota}_{I,q} & \dots & \boldsymbol{\iota}_{I,Q} \end{bmatrix} \qquad \boldsymbol{\kappa} = \begin{bmatrix} \boldsymbol{\kappa}_1 \\ \vdots \\ \boldsymbol{\kappa}_q \\ \vdots \\ \boldsymbol{\kappa}_Q \end{bmatrix} \tag{49}$$

Making the appropriate substitutions yields the final system of dynamic equations according to (16) which can be used with standard controllability, stability, etc. analysis techniques. Output $\mathbf{F}$ of the system of equations is structured according to (50) where F_L and F_R are the forces provided by the left and right endpoints of the array respectively. Tab. 6 defines the general array variables for reference.

$$\mathbf{F} = \begin{bmatrix} F_L \\ F_R \end{bmatrix} \tag{50}$$

3.5 Hill-Type Array Example

As an example, an array built from the damped Hill-Type cell model shown in Fig. 17 and with fingerprint (51) is modeled here. This is shown graphically in Fig. 19. Each cell in the array carries

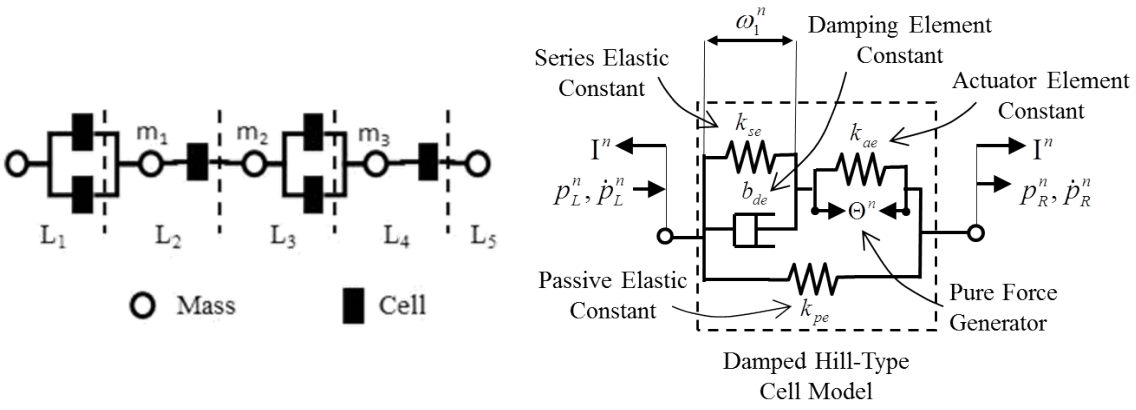


Figure 19: Graphical representation of the example actuator array with the fingerprint given in (51).

an output force given by (10), repeated here as (52), and an internal state which changes according to (12), repeated here as (53).

$$\left[\begin{array}{c|c|c|c|c} \# & 3 & 1 & 3 & 1 \\ 2 & 1 & 2 & 1 & \# \end{array}\right] \tag{51}$$

$$I^n = k_{pe} \cdot (p_R^n - p_L^n) + k_{ae} \cdot (\omega_1^n) + \theta^n \tag{52}$$

$$\dot{\omega}_1^n = \frac{k_{se}}{b_{de}} \cdot (p_R^n - p_L^n) - \frac{k_{se} + k_{ae}}{b_{de}} \cdot (\omega_1^n) - \frac{1}{b_{de}} \cdot (\theta^n) + (\dot{p}_R^n - \dot{p}_L^n) \tag{53}$$

We will further assume that all masses in the array have a mass of ϕ, thus giving $\mathbf{\Phi}$ represented by (54).

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{\phi} & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \frac{1}{\phi} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & \dots & \frac{1}{\phi} & 0 \\ 0 & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix} \tag{54}$$

(52) represents a tension force which has equal magnitude but opposite direction on both sides of the cell. Accordingly, the simplification in (55) can be made.

$$I^n = E_L^n = -E_R^n \tag{55}$$

This allows for setting coe according to (56) and cow according to (57).

$$\begin{aligned} k_{pe} &= -\text{coe}(p_L)_L^n = \text{coe}(p_R)_L^n = \text{coe}(p_L)_R^n = -\text{coe}(p_R)_R^n \ , \ \forall n \\ 0 &= \text{coe}(\dot{p}_L)_L^n = \text{coe}(\dot{p}_R)_L^n = \text{coe}(\dot{p}_L)_R^n = \text{coe}(\dot{p}_R)_R^n \ , \ \forall n \\ k_{ae} &= \text{coe}(w_1)_L^n = -\text{coe}(w_1)_R^n \ , \ \forall n \end{aligned} \tag{56}$$

$$\begin{aligned} \frac{k_{se}}{b_{de}} &= -\text{cow}(p_L)_1^n = \text{cow}(p_R)_1^n \ , \ \forall n \\ 1 &= -\text{cow}(\dot{p}_L)_1^n = \text{cow}(\dot{p}_R)_1^n \ , \ \forall n \\ -\frac{k_{se}+k_{ae}}{b_{de}} &= \text{cow}(w_i)_1^n \ , \ \forall n \end{aligned} \tag{57}$$

$\text{coe}(\theta_v)_L^n$, $\text{coe}(\theta_v)_R^n$, and $\text{cow}(\theta_v)_1^n$ depend on the connection of cells to control inputs. Treating these control inputs as having a magnitude equal to the force generated by the actuator, we can set $\text{coe}(\theta_v)_L^n$, $\text{coe}(\theta_v)_R^n$, and $\text{cow}(\theta_v)_1^n$ equal to 1, -1, and $-\frac{1}{b_{de}}$ respectively when control v is connected to cell n. They are set to 0 otherwise.

Given fingerprint (51), $\mathbf{G}$ and $\mathbf{H}$ are given by (58).

$$\mathbf{G} = \left[\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \qquad \mathbf{H} = \left[\begin{array}{c|cccc} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array}\right] \tag{58}$$

Using (37), $\boldsymbol{\alpha}$ is given by (59).

$$\begin{aligned} \boldsymbol{\alpha} &= \boldsymbol{\Phi} \cdot \left[\begin{array}{c} \operatorname{diag}\left(\left[\operatorname{coe}(p_R)_R \cdot \mathbf{H}\right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \operatorname{coe}(p_L)_L \cdot \boldsymbol{\mu}\right) \\ + \left[\mathbf{G} \cdot \operatorname{coe}(p_L)_R \cdot \mathbf{H}\right]^\top + \mathbf{G} \cdot \operatorname{coe}(p_R)_L \cdot \mathbf{H} \end{array}\right] = \left[\begin{array}{c|c|c} \alpha_{ab} & \boldsymbol{\alpha}_b & \alpha_{bd} \\ \hline \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_s & \boldsymbol{\alpha}_d \\ \hline \alpha_{ac} & \boldsymbol{\alpha}_c & \alpha_{cd} \end{array}\right] \\ \boldsymbol{\alpha} &= \left[\begin{array}{c|cccc} -2 \cdot k_{pe} & 2 \cdot k_{pe} & 0 & 0 & 0 \\ \hline \frac{2 \cdot k_{pe}}{\phi_2} & \frac{-3 \cdot k_{pe}}{\phi_2} & \frac{k_{pe}}{\phi_2} & 0 & 0 \\ 0 & \frac{k_{pe}}{\phi_3} & \frac{-3 \cdot k_{pe}}{\phi_3} & \frac{2 \cdot k_{pe}}{\phi_3} & 0 \\ 0 & 0 & \frac{2 \cdot k_{pe}}{\phi_4} & \frac{-3 \cdot k_{pe}}{\phi_4} & \frac{k_{pe}}{\phi_4} \\ \hline 0 & 0 & 0 & k_{pe} & -k_{pe} \end{array}\right] = \left[\begin{array}{c|c|c} \alpha_{ab} & \boldsymbol{\alpha}_b & \alpha_{bd} \\ \hline \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_s & \boldsymbol{\alpha}_d \\ \hline \alpha_{ac} & \boldsymbol{\alpha}_c & \alpha_{cd} \end{array}\right] \end{aligned} \tag{59}$$

Using (38), $\boldsymbol{\beta}$ is given by (60).

$$\begin{aligned} \boldsymbol{\beta} &= \boldsymbol{\Phi} \cdot \left[\begin{array}{c} \operatorname{diag}\left(\left[\operatorname{coe}(\dot{p}_R)_R \cdot \mathbf{H}\right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \operatorname{coe}(\dot{p}_L)_L \cdot \boldsymbol{\mu}\right) \\ + \left[\mathbf{G} \cdot \operatorname{coe}(\dot{p}_L)_R \cdot \mathbf{H}\right]^\top + \mathbf{G} \cdot \operatorname{coe}(\dot{p}_R)_L \cdot \mathbf{H} \end{array}\right] = \left[\begin{array}{c|c|c} \beta_{ab} & \boldsymbol{\beta}_b & \beta_{bd} \\ \hline \boldsymbol{\beta}_a & \boldsymbol{\beta}_s & \boldsymbol{\beta}_d \\ \hline \beta_{ac} & \boldsymbol{\beta}_c & \beta_{cd} \end{array}\right] \\ \boldsymbol{\beta} &= \left[\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \end{array}\right] = \left[\begin{array}{c|c|c} \beta_{ab} & \boldsymbol{\beta}_b & \beta_{bd} \\ \hline \boldsymbol{\beta}_a & \boldsymbol{\beta}_s & \boldsymbol{\beta}_d \\ \hline \beta_{ac} & \boldsymbol{\beta}_c & \beta_{cd} \end{array}\right] \end{aligned} \tag{60}$$

Using (41) and (42), $\boldsymbol{\gamma}$ is given by (61).

$$\boldsymbol{\gamma}_q = \boldsymbol{\Phi} \cdot \left[\left[\text{coe}(w_q)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \text{coe}(w_q)_L \right] \ , \ \forall q$$

$$\boldsymbol{\gamma} = \begin{bmatrix} \boldsymbol{\gamma}_1 & \dots & \boldsymbol{\gamma}_q & \dots & \boldsymbol{\gamma}_Q \end{bmatrix} = \begin{bmatrix} \boldsymbol{\gamma}_b \\ \hline \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\gamma}_c \end{bmatrix}$$

$$\boldsymbol{\gamma} = \left[\begin{array}{cccccc} k_{ae} & k_{ae} & 0 & 0 & 0 & 0 \\ \hline \frac{-k_{ae}}{\phi_2} & \frac{-k_{ae}}{\phi_2} & \frac{k_{ae}}{\phi_2} & 0 & 0 & 0 \\ 0 & 0 & \frac{-k_{ae}}{\phi_3} & \frac{k_{ae}}{\phi_3} & \frac{k_{ae}}{\phi_3} & 0 \\ 0 & 0 & 0 & \frac{-k_{ae}}{\phi_4} & \frac{-k_{ae}}{\phi_4} & \frac{k_{ae}}{\phi_4} \\ \hline 0 & 0 & 0 & 0 & 0 & -k_{ae} \end{array}\right] = \left[\begin{array}{c} \boldsymbol{\gamma}_b \\ \hline \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\gamma}_c \end{array}\right] \tag{61}$$

Using (43), $\boldsymbol{\delta}$ is given by (62).

$$\boldsymbol{\delta} = \boldsymbol{\Phi} \cdot \left[\mathbf{H}^\top \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_R + \mathbf{G} \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_L \right] = \begin{bmatrix} \boldsymbol{\delta}_b \\ \hline \boldsymbol{\delta}_s \\ \hline \boldsymbol{\delta}_c \end{bmatrix}$$

$$\boldsymbol{\delta} = \left[\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ \hline \frac{-1}{\phi_2} & \frac{-1}{\phi_2} & \frac{1}{\phi_2} & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{\phi_3} & \frac{1}{\phi_3} & \frac{1}{\phi_3} & 0 \\ 0 & 0 & 0 & \frac{-1}{\phi_4} & \frac{-1}{\phi_4} & \frac{1}{\phi_4} \\ \hline 0 & 0 & 0 & 0 & 0 & -1 \end{array}\right] = \left[\begin{array}{c} \boldsymbol{\delta}_b \\ \hline \boldsymbol{\delta}_s \\ \hline \boldsymbol{\delta}_c \end{array}\right] \tag{62}$$

Using (44) and (46), $\boldsymbol{\zeta}$ is given by (63).

$$\boldsymbol{\zeta}_q = \left[\left[\mathbf{G} \cdot \text{cow}(p_L)_q \right]^\top + \text{cow}(p_R)_q \cdot \mathbf{H} \right] \ , \ \forall q$$

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\zeta}_1 \\ \vdots \\ \boldsymbol{\zeta}_q \\ \vdots \\ \boldsymbol{\zeta}_Q \end{bmatrix} = \left[\begin{array}{c|c|c} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_s & \boldsymbol{\zeta}_d \end{array} \right] \tag{63}$$

$$\boldsymbol{\zeta} = \left[\begin{array}{c|ccc|c} -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & 0 & 0 \\ -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & 0 & 0 \\ 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & 0 \\ 0 & 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 \\ 0 & 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 \\ 0 & 0 & 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} \end{array} \right] = \left[\begin{array}{c|c|c} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_s & \boldsymbol{\zeta}_d \end{array} \right]$$

Using (45) and (46), $\boldsymbol{\eta}$ is given by (64).

$$\boldsymbol{\eta}_q = \left[\left[\mathbf{G} \cdot \mathrm{cow}(\dot{p}_L)_q \right]^\top + \mathrm{cow}(\dot{p}_R)_q \cdot \mathbf{H} \right] \ , \ \forall q$$

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \vdots \\ \boldsymbol{\eta}_q \\ \vdots \\ \boldsymbol{\eta}_Q \end{bmatrix} = \left[\begin{array}{c|c|c} \boldsymbol{\eta}_a & \boldsymbol{\eta}_s & \boldsymbol{\eta}_d \end{array} \right] \tag{64}$$

$$\boldsymbol{\eta} = \left[\begin{array}{c|ccc|c} -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{array} \right] = \left[\begin{array}{c|c|c} \boldsymbol{\eta}_a & \boldsymbol{\eta}_s & \boldsymbol{\eta}_d \end{array} \right]$$

Using (47), $\boldsymbol{\iota}$ is given by (65).

$$\boldsymbol{\iota}_{i,q} = \text{cow}(w_i)_q \ , \ \forall i, q$$

$$\boldsymbol{\iota} = \begin{bmatrix} \boldsymbol{\iota}_{1,1} & \dots & \boldsymbol{\iota}_{1,q} & \dots & \boldsymbol{\iota}_{1,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{i,1} & \dots & \boldsymbol{\iota}_{i,q} & \dots & \boldsymbol{\iota}_{i,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{I,1} & \dots & \boldsymbol{\iota}_{I,q} & \dots & \boldsymbol{\iota}_{I,Q} \end{bmatrix} \tag{65}$$

$$\boldsymbol{\iota} = \begin{bmatrix} -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} \end{bmatrix}$$

Using (48), $\boldsymbol{\kappa}$ is given by (66).

$$\boldsymbol{\kappa}_q = \text{cow}\left(\boldsymbol{\Theta}\right)_q \ \ , \ \forall q$$

$$\boldsymbol{\kappa} = \begin{bmatrix} \boldsymbol{\kappa}_1 \\ \vdots \\ \boldsymbol{\kappa}_q \\ \vdots \\ \boldsymbol{\kappa}_Q \end{bmatrix} \tag{66}$$

$$\boldsymbol{\kappa} = \begin{bmatrix} -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} \end{bmatrix}$$

Arranging according to (24), (25), (26), (27), (28), (29), (30), and (31) yields the final dynamic equations of motion for the system. (67) shows $\mathbf{A}$, (68) shows $\mathbf{B}$, (69) shows $\mathbf{C}$, and (70) shows $\mathbf{D}$.

$$
\mathbf{A} = \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \hline \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \\ \hline \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s & \boldsymbol{\iota} \end{array}\right] = \left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -\frac{3 \cdot k_{pe}}{\phi} & \frac{k_{pe}}{\phi} & 0 & 0 & 0 & 0 \\ \frac{k_{pe}}{\phi} & -\frac{3 \cdot k_{pe}}{\phi} & \frac{2 \cdot k_{pe}}{\phi} & 0 & 0 & 0 \\ 0 & \frac{2 \cdot k_{pe}}{\phi} & -\frac{3 \cdot k_{pe}}{\phi} & 0 & 0 & 0 \\ \hline \frac{k_{se}}{b_{de}} & 0 & 0 & 1 & 0 & 0 \\ \frac{k_{se}}{b_{de}} & 0 & 0 & 1 & 0 & 0 \\ -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 & 0 \\ 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 \\ 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 \\ 0 & 0 & -\frac{k_{se}}{b_{de}} & 0 & 0 & -1 \end{array}\right] \cdots \tag{67}
$$

$$
\cdots \left[\begin{array}{cccccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \hline -\frac{k_{ae}}{\phi} & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & 0 \\ 0 & 0 & 0 & -\frac{k_{ae}}{\phi} & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} \\ \hline -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} \end{array}\right]
$$

$$
\mathbf{B} = \left[\begin{array}{cc|cc|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d & \boldsymbol{\delta}_s \\ \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d & \boldsymbol{\kappa} \end{array}\right]
$$

$$
= \left[\begin{array}{cc|cc|cccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{2 \cdot k_{pe}}{\phi} & 0 & 0 & 0 & \frac{-1}{\phi} & \frac{-1}{\phi} & \frac{1}{\phi} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\phi} & \frac{1}{\phi} & \frac{1}{\phi} & 0 \\ 0 & \frac{k_{pe}}{\phi} & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\phi} & \frac{-1}{\phi} & \frac{1}{\phi} \\ -\frac{k_{se}}{b_{de}} & 0 & -1 & 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ -\frac{k_{se}}{b_{de}} & 0 & -1 & 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 \\ 0 & \frac{k_{se}}{b_{de}} & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} \end{array}\right] \tag{68}
$$

$$
\mathbf{C} = \left[\begin{array}{c|c|c} \boldsymbol{\alpha}_b & \boldsymbol{\beta}_b & \boldsymbol{\gamma}_b \\ \boldsymbol{\alpha}_c & \boldsymbol{\beta}_c & \boldsymbol{\gamma}_c \end{array}\right]
$$

$$
= \left[\begin{array}{ccc|ccc|cccccc} 2 \cdot k_{pe} & 0 & 0 & 0 & 0 & 0 & k_{ae} & k_{ae} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_{ae} \end{array}\right] \tag{69}
$$

$$
\mathbf{D} = \left[\begin{array}{cc|cc|c} \alpha_{ab} & \alpha_{bd} & \beta_{ab} & \beta_{bd} & \boldsymbol{\delta}_b \\ \alpha_{ac} & \alpha_{cd} & \beta_{ac} & \beta_{cd} & \boldsymbol{\delta}_c \end{array}\right]
$$

$$
= \left[\begin{array}{cc|cc|cccccc} -2 \cdot k_{pe} & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{array}\right] \tag{70}
$$

3.6 Dynamic Numerical Results

Numerical results were used to judge the validity and efficiency of the presented dynamic modeling methods. For each trial two cases were run, one testing the isometric forces experienced by

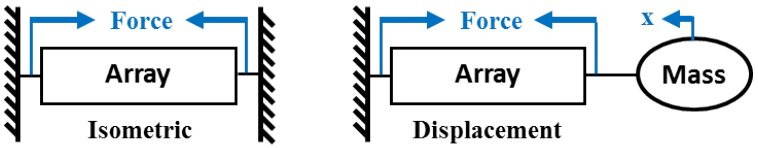


Figure 20: Isometric and Displacement trials conducted for numeric actuator array tests.

Table 7: Comparison of presented dynamics method and Matlab's SimMechanics environment.

# Cells	Model	Thesis Method Config (min)	Thesis Method Computation (sec)	SimMechanics Method Programing (min)
6	Damped Hill	3	0.00162	8
14	Damped Hill	3	0.00049	10
100	Damped Hill	4	0.00370	51
200	Forced Spring	4	0.02527	N/A
600	Forced Spring	4	0.59380	N/A
1000	Forced Spring	4	3.03786	N/A

the array and one testing the displacement of the array when connected to mass. Fig. 20 shows these two cases. The isometric force graph for each result shows the force at the right endpoint of the array and the displacement graph shows the displacement of each cell in the actuator array.

For each trial the system dynamics were computed using the method presented in the thesis and then simulated using Matlab's ODE 45 dynamic equation solver. The same system was then programed into Matlab's Simulink package SimMechanics (Second Generation). The physical elements were represented directly in this graphical environment and the system was allowed to determine the appropriate dynamics. Tab. 7 provides a comparison between the two methods by showing configuration and computation time for the method presented in this thesis and the programing time to input the model into SimMechanics.

The 6, 14, and 100 cell trials used the damped Hill-Type model cell and a shaped input, shown in Fig. 21, to match the example in this chapter and the physical damped SMA experimental device presented in chapter 43. While the model was similar to the physical system, the spring stiffnesses and damping coefficient were different to show a dynamic response.

The 200, 600, and 1000 cell trials used a simpler internal cell model consisting a spring acted upon by a pure force and were acted upon by a step input, as shown in Fig. 21. This simplification was made due to the time required to simulate the model, not due to the calculation method itself. The computation time difference between the damped spring and damped Hill-Type model was

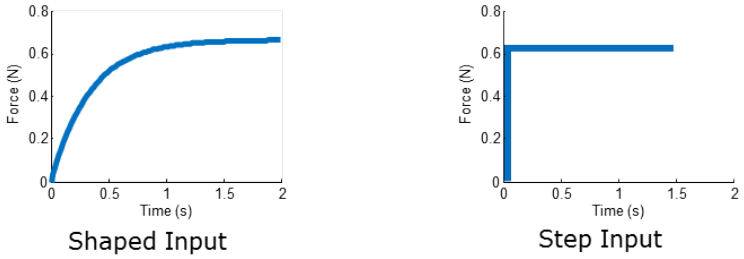


Figure 21: Shaped Input and Step Input used for numerical simulation.

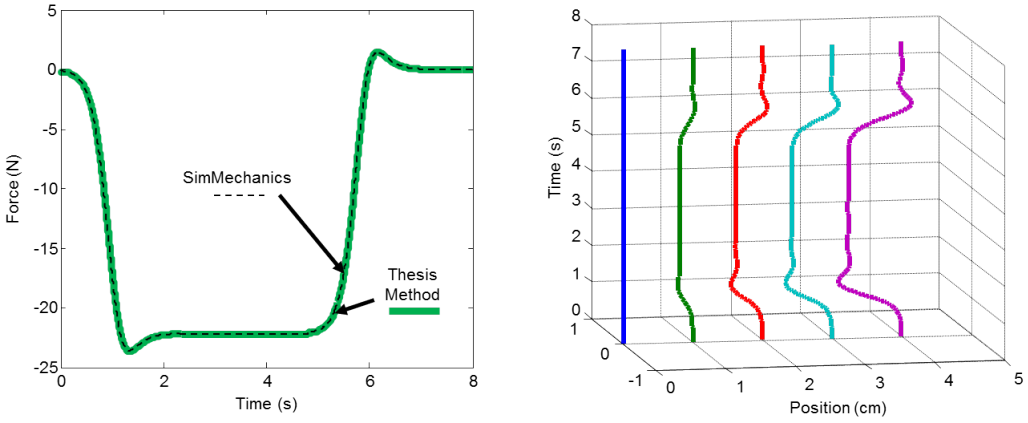


Figure 22: Isometric and displacement numeric trials for a 6 cell damped Hill-Type model actuator array. The left graph shows the isometric force at the right endpoint of the array. The right graph shows the displacements of all cells in the array when the array is connected to a load. The topology for this cell is given in Fig. 19

negligible even for the 1000 cell trial. Trials past 100 cells were not programed into SimMechanics due to the diminishing returns of programing such a large model and the memory requirements of running it. As such, the results for the 200, 600, and 1000 cell trials only contain simulated results from the method provided in this thesis. The oscillation in these results is expected due to the undamped cell dynamics and actuation by a step function.

The 14 cell system in Fig. 10 was also simulated using a dynamic Hill-Type cell model shown in Fig. 28. The cell model is based on a Miga NanoMuscle 704 SMA actuator and lightly damped pen springs. Further details of the model are given in chapter 6. For this trial, the cells were grouped to yield 5 control levels with the 14 cells. Each time an additional control group is activated, the force level of the array doubles giving a force profile resembling signal dependent noise observed in biological muscle. This is explored later in chapter 5.

Further validation, both numerically and experimentally, is provided in chapter 6.

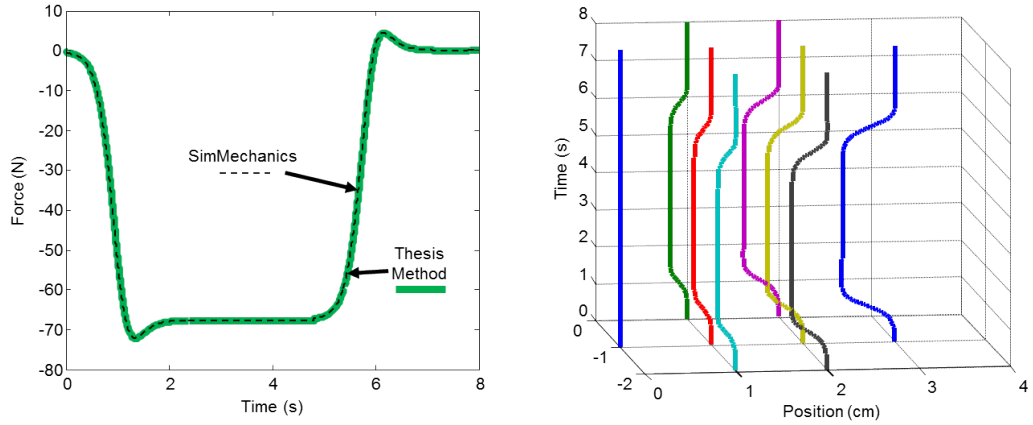


Figure 23: Isometric and displacement numeric trials for a 14 cell damped Hill-Type model actuator array. The left graph shows the isometric force at the right endpoint of the array. The right graph shows the displacements of all cells in the array when the array is connected to a load. The topology for this cell is given in Fig. 10

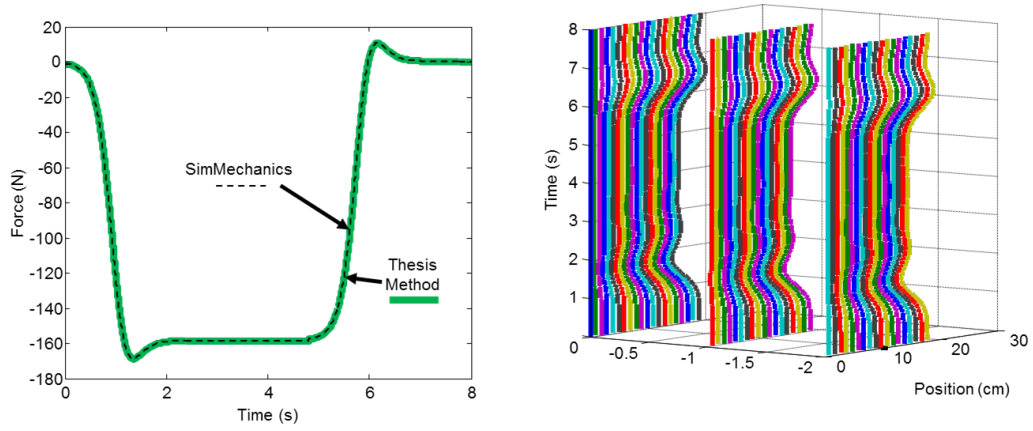


Figure 24: Isometric and displacement numeric trials for a 100 cell damped Hill-Type model actuator array. The left graph shows the isometric force at the right endpoint of the array. The right graph shows the displacements of all cells in the array when the array is connected to a load.

3.6.1 Algorithm Comparison

The SimMechanics results matched the theory, however programming the system into SimMechanics took much longer. For low numbers of cells the difference is manageable, however as cell count increased the SimMechanics method became unusable. More complex topologies also took significantly longer to program in SimMechanics while topological complexity did not affect the provided theory. Finally, any changes to the model, either topological or to cell equations of motion, required a complete reprogram of the SimMechanics model versus changing only a few

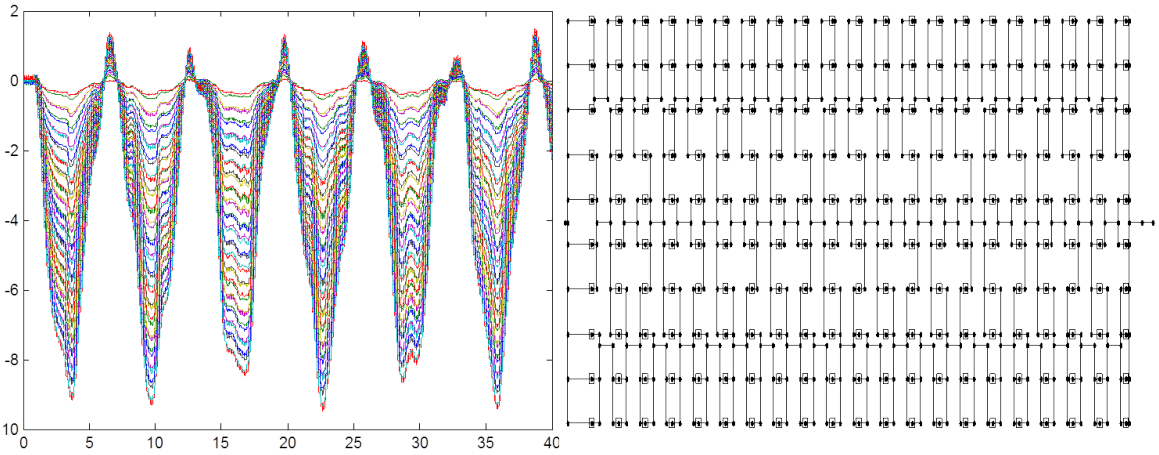


Figure 25: Displacement numeric trial for a 200 cell damped Hill-Type model actuator array. The left graph shows the displacements of all cells in the array when the array is connected to a load. The right graph shows the structure of the array.

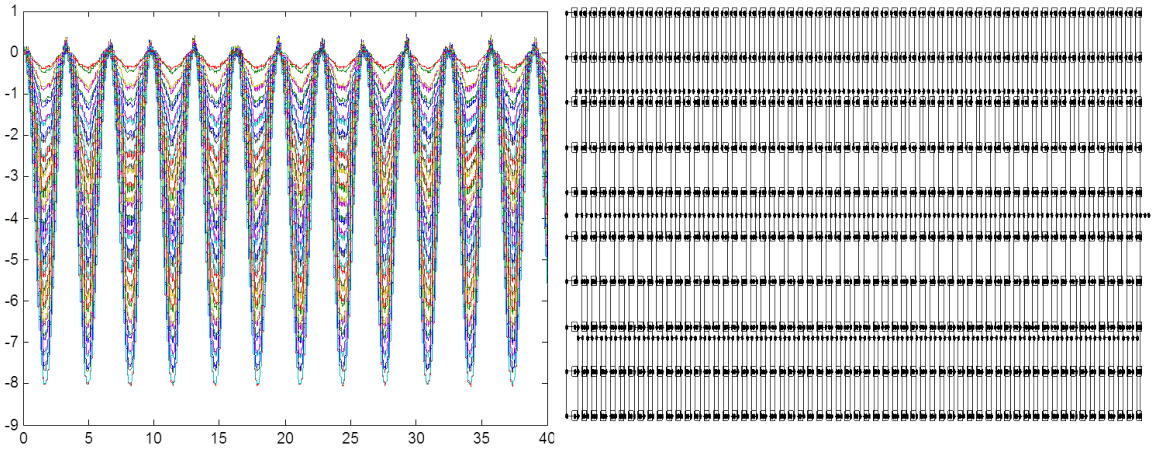


Figure 26: Displacement numeric trial for a 600 cell damped Hill-Type model actuator array. The left graph shows the displacements of all cells in the array when the array is connected to a load. The right graph shows the structure of the array.

constants using the above theory.

3.7 General Linear Dynamic Equations Proof

The process for developing the linear dynamic equations for actuator arrays begins with Newtons second law, a force balance about each mass in the array and the resulting acceleration. Since each mass is only connected to cells, the force acting on the mass is the sum of the forces from cells pulling from the left and those pulling from the right. Equation (71) shows the resultant acceleration $\ddot{p}_m$ for mass m with list of cells ℓ_L^m connected to the left and list of cells ℓ_R^m connected to the

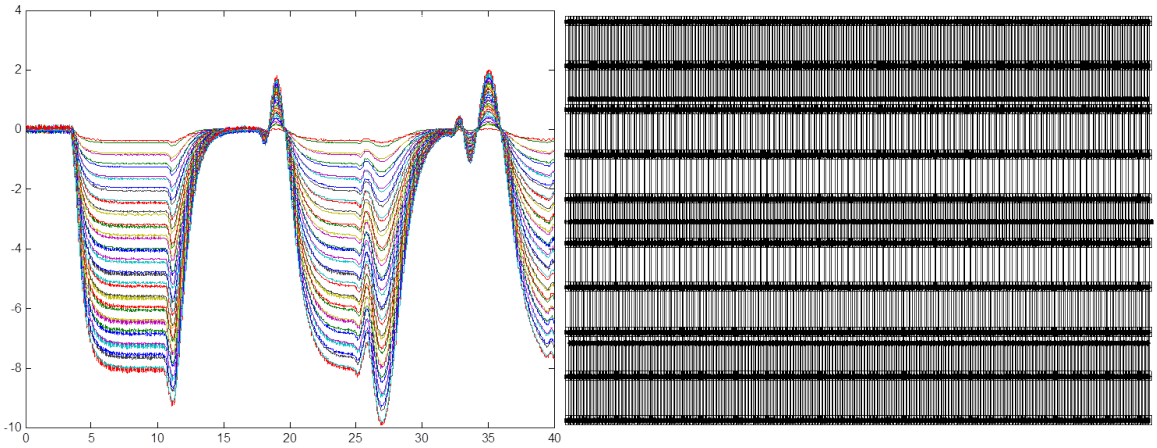


Figure 27: Displacement numeric trial for a 1000 cell damped Hill-Type model actuator array. The left graph shows the displacements of all cells in the array when the array is connected to a load. The right graph shows the structure of the array.

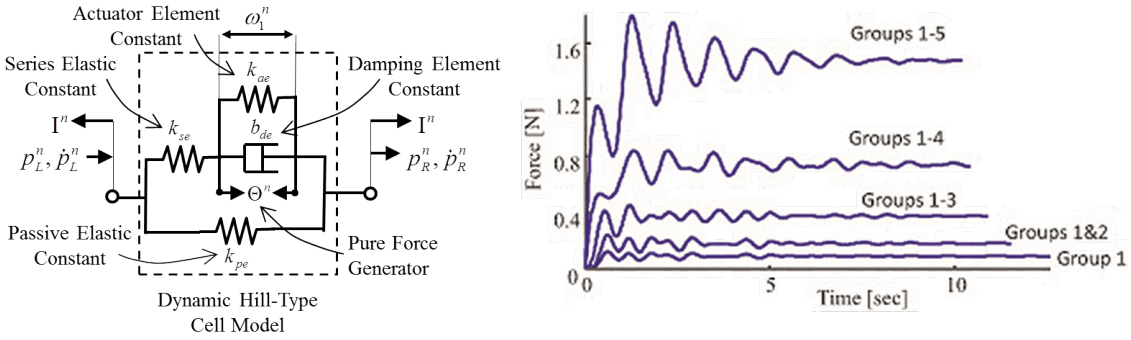


Figure 28: Left: Dynamic Hill-Type cell model based on Miga NanoMuscle 714 SMA actuators and lightly damped pen springs. Further details of the model are given in chapter 6. Right: Floating-point quantized actuation of an non-uniform actuator array.

right. The mass of mass m is ϕ_m. Sign convention holds that right is positive for force, position, and velocity for all array elements, so all forces are treated as positive in the theory and coefficient matrices will be used to define negative forces on a per cell basis.

$$\phi_m \cdot \ddot{p}_m = \left[\sum_{i=\ell_L^m} E_R^i\right] + \left[\sum_{i=\ell_R^m} E_L^i\right] \tag{71}$$

Substituting according to (17) and (18), the force on the left and right of each cell respectively, we get (72).

$$\begin{aligned}\phi_m \cdot \ddot{p}_m = &\sum_{i=\ell_L^m} \left(\text{coe}(p_L)_R^i \cdot \{p_L^i\}\right) + \sum_{i=\ell_L^m} \left(\text{coe}(p_R)_R^i \cdot \{p_R^i\}\right) \\ &+ \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_L)_R^i \cdot \{\dot{p}_L^i\}\right) + \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_R)_R^i \cdot \{\dot{p}_R^i\}\right) \\ &+ \sum_{i=\ell_R^m} \left(\text{coe}(p_L)_L^i \cdot \{p_L^i\}\right) + \sum_{i=\ell_R^m} \left(\text{coe}(p_R)_L^i \cdot \{p_R^i\}\right) \\ &+ \sum_{i=\ell_R^m} \left(\text{coe}(\dot{p}_L)_L^i \cdot \{\dot{p}_L^i\}\right) + \sum_{i=\ell_R^m} \left(\text{coe}(\dot{p}_R)_L^i \cdot \{\dot{p}_R^i\}\right) \\ &+ \sum_{i=\ell_L^m} \left[\sum_{v=1}^{V} \left(\text{coe}(\theta_v)_R^i \cdot \{\theta_v\}\right)\right] + \sum_{i=\ell_R^m} \left[\sum_{v=1}^{V} \left(\text{coe}(\theta_v)_L^i \cdot \{\theta_v\}\right)\right] \\ &+ \sum_{i=\ell_L^m} \left[\sum_{q=1}^{Q} \left(\text{coe}(w_q)_R^i \cdot \{w_q^i\}\right)\right] + \sum_{i=\ell_R^m} \left[\sum_{q=1}^{Q} \left(\text{coe}(w_q)_L^i \cdot \{w_q^i\}\right)\right] \end{aligned} \tag{72}$$

(17) and (18) are repeated here as (73) and (74) for clarity.

$$\begin{aligned}E_L^n = &\text{coe}(p_L)_L^n \cdot \{p_L^n\} + \text{coe}(p_R)_L^n \cdot \{p_R^n\} + \text{coe}(\dot{p}_L)_L^n \cdot \{\dot{p}_L^n\} + \text{coe}(\dot{p}_R)_L^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\text{coe}(\theta_v)_L^n \cdot \{\theta_v\}\right) + \sum_{q=1}^{Q} \left(\text{coe}(w_q)_L^n \cdot \{w_q^n\}\right) \end{aligned} \tag{73}$$

$$\begin{aligned}E_R^n = &\text{coe}(p_L)_R^n \cdot \{p_L^n\} + \text{coe}(p_R)_R^n \cdot \{p_R^n\} + \text{coe}(\dot{p}_L)_R^n \cdot \{\dot{p}_L^n\} + \text{coe}(\dot{p}_R)_R^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\text{coe}(\theta_v)_R^n \cdot \{\theta_v\}\right) + \sum_{q=1}^{Q} \left(\text{coe}(w_q)_R^n \cdot \{w_q^n\}\right) \end{aligned} \tag{74}$$

Since the mass to the right of the cell connected left of the current mass is the current mass, the simplification in (75) can be made.

$$\begin{aligned}\sum_{i=\ell_L^m} \left(\text{coe}(p_R)_R^i \cdot \{p_R^i\}\right) &= \sum_{i=\ell_L^m} \left(\text{coe}(p_R)_R^i \cdot \{p_m\}\right) \\ \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_R)_R^i \cdot \{\dot{p}_R^i\}\right) &= \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_R)_R^i \cdot \{\dot{p}_m\}\right) \end{aligned} \tag{75}$$

Similarly since the mass to the left of the cell connected right of the current mass is the current mass, the simplification in (76) can be made.

$$\sum_{i=\ell_R^m}\Big(\text{coe}(p_L)_L^i\cdot\{p_L^i\}\Big)=\sum_{i=\ell_R^m}\Big(\text{coe}(p_L)_L^i\cdot\{p_m\}\Big)$$
$$\sum_{i=\ell_R^m}\Big(\text{coe}(\dot{p}_L)_L^i\cdot\{\dot{p}_L^i\}\Big)=\sum_{i=\ell_R^m}\Big(\text{coe}(\dot{p}_L)_L^i\cdot\{\dot{p}_m\}\Big) \tag{76}$$

This results in the more simplified form (77).

$$\begin{aligned}\phi_m\cdot\ddot{p}_m=&\sum_{i=\ell_L^m}\Big(\text{coe}(p_R)_R^i\cdot\{p_m\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(p_L)_L^i\cdot\{p_m\}\Big)\\&+\sum_{i=\ell_L^m}\Big(\text{coe}(p_L)_R^i\cdot\{p_L^i\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(p_R)_L^i\cdot\{p_R^i\}\Big)\\&+\sum_{i=\ell_L^m}\Big(\text{coe}(\dot{p}_R)_R^i\cdot\{\dot{p}_m\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(\dot{p}_L)_L^i\cdot\{\dot{p}_m\}\Big)\\&+\sum_{i=\ell_L^m}\Big(\text{coe}(\dot{p}_L)_R^i\cdot\{\dot{p}_L^i\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(\dot{p}_R)_L^i\cdot\{\dot{p}_R^i\}\Big)\\&+\sum_{v=1}^{V}\left[\sum_{i=\ell_L^m}\Big(\text{coe}(\theta_v)_R^i\cdot\{\theta_v\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(\theta_v)_L^i\cdot\{\theta_v\}\Big)\right]\\&+\sum_{q=1}^{Q}\left[\sum_{i=\ell_L^m}\Big(\text{coe}(w_q)_R^i\cdot\{w_q^i\}\Big)+\sum_{i=\ell_R^m}\Big(\text{coe}(w_q)_L^i\cdot\{w_q^i\}\Big)\right]\end{aligned} \tag{77}$$

Given $\mathbf{G}$ and $\mathbf{H}$ - as defined by (4) in chapter 2 - and 1's vector $\boldsymbol{\mu}$ defined by (40), $(\mathbf{H})^\top\cdot\boldsymbol{\mu}$ gives a vector where the m'th row is the sum of the incoming cells connected to mass m, or $\Big(\sum_{i=\ell_L^m}1\Big)$. Likewise, $\mathbf{G}\cdot\boldsymbol{\mu}$ gives a vector where the m'th row is the sum of outgoing cells connected to mass m, or $\Big(\sum_{i=\ell_R^m}1\Big)$.

Combining with $\text{coe}(p_R)_R$ and $\text{coe}(p_L)_L$, as defined by (78) and (79), we can simplify the first line of (77) using (80) and (81).

$$\text{coe}(p_R)_R=\begin{bmatrix}\text{coe}(p_R)_R^1 & 0 & 0\\ 0 & \text{coe}(p_R)_R^n & 0\\ 0 & 0 & \text{coe}(p_R)_R^N\end{bmatrix} \tag{78}$$

$$\text{coe}(p_L)_L = \begin{bmatrix} \text{coe}(p_L)_L^1 & 0 & 0 \\ 0 & \text{coe}(p_L)_L^n & 0 \\ 0 & 0 & \text{coe}(p_L)_L^N \end{bmatrix} \tag{79}$$

$$\begin{aligned} \sum_{i=\ell_L^m} \left(\text{coe}(p_R)_R^i \cdot \{p_m\} \right) &= \left[[\text{coe}(p_R)_R \cdot \mathbf{H}]^\top \cdot \boldsymbol{\mu} \right]_{(m)} \cdot p_m \\ &= \left[\text{diag} \left([\text{coe}(p_R)_R \cdot \mathbf{H}]^\top \cdot \boldsymbol{\mu} \right) \cdot \mathbf{P} \right]_{(m,m)} \end{aligned} \tag{80}$$

$$\begin{aligned} \sum_{i=\ell_R^m} \left(\text{coe}(p_L)_L^i \cdot \{p_m\} \right) &= [\mathbf{G} \cdot \text{coe}(p_L)_L \cdot \boldsymbol{\mu}]_{(m)} \cdot p_m \\ &= [\text{diag} \left(\mathbf{G} \cdot \text{coe}(p_L)_L \cdot \boldsymbol{\mu} \right) \cdot \mathbf{P}]_{(m,m)} \end{aligned} \tag{81}$$

Using the same logic but combining with $\text{coe}(\dot{p}_R)_R$ and $\text{coe}(\dot{p}_L)_L$, as defined by (82) and (83), we can simplify the 3rd line of (77) using (84) and (85).

$$\text{coe}(\dot{p}_R)_R = \begin{bmatrix} \text{coe}(\dot{p}_R)_R^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_R)_R^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_R)_R^N \end{bmatrix} \tag{82}$$

$$\text{coe}(\dot{p}_L)_L = \begin{bmatrix} \text{coe}(\dot{p}_L)_L^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_L)_L^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_L)_L^N \end{bmatrix} \tag{83}$$

$$\begin{aligned} \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_R)_R^i \cdot \{\dot{p}_m\} \right) &= \left[[\text{coe}(\dot{p}_R)_R \cdot \mathbf{H}]^\top \cdot \boldsymbol{\mu} \right]_{(m)} \cdot \dot{p}_m \\ &= \left[\text{diag} \left([\text{coe}(\dot{p}_R)_R \cdot \mathbf{H}]^\top \cdot \boldsymbol{\mu} \right) \cdot \dot{\mathbf{P}} \right]_{(m,m)} \end{aligned} \tag{84}$$

$$\sum_{i=\ell_R^m}\left(\text{coe}(\dot{p}_L)_L^i\cdot\{\dot{p}_m\}\right)=\left[\mathbf{G}\cdot\text{coe}(\dot{p}_L)_L\cdot\boldsymbol{\mu}\right]_{(m)}\cdot\dot{p}_m$$

$$=\left[\text{diag}\left(\mathbf{G}\cdot\text{coe}(\dot{p}_L)_L\cdot\boldsymbol{\mu}\right)\cdot\dot{\mathbf{P}}\right]_{(m,m)} \tag{85}$$

Row m of $\mathbf{G}\cdot\mathbf{H}$ contains the connections between masses in the forward (outgoing) direction while the transpose contains the same information in the reverse (incoming) direction relative to mass m.

Combining with $\text{coe}(p_L)_R$ and $\text{coe}(p_R)_L$, as defined by (86) and (87), we can simplify the second line of (77) using (88) and (89).

$$\text{coe}(p_L)_R=\begin{bmatrix}\text{coe}(p_L)_R^1 & 0 & 0\\ 0 & \text{coe}(p_L)_R^n & 0\\ 0 & 0 & \text{coe}(p_L)_R^N\end{bmatrix} \tag{86}$$

$$\text{coe}(p_R)_L=\begin{bmatrix}\text{coe}(p_R)_L^1 & 0 & 0\\ 0 & \text{coe}(p_R)_L^n & 0\\ 0 & 0 & \text{coe}(p_R)_L^N\end{bmatrix} \tag{87}$$

$$\sum_{i=\ell_L^m}\left(\text{coe}(p_L)_R^i\cdot\left\{p_L^i\right\}\right)=\left[\left[\mathbf{G}\cdot\text{coe}(p_L)_R\cdot\mathbf{H}\right]^\top\right]_{(m,m)}\cdot p_m$$

$$=\left[\left[\mathbf{G}\cdot\text{coe}(p_L)_R\cdot\mathbf{H}\right]^\top\cdot\mathbf{P}\right]_{(m,m)} \tag{88}$$

$$\sum_{i=\ell_R^m}\left(\text{coe}(p_R)_L^i\cdot\left\{p_R^i\right\}\right)=\left[\mathbf{G}\cdot\text{coe}(p_R)_L\cdot\mathbf{H}\right]_{(m,m)}\cdot p_m$$

$$=\left[\mathbf{G}\cdot\text{coe}(p_R)_L\cdot\mathbf{H}\cdot\mathbf{P}\right]_{(m,m)} \tag{89}$$

Using the same logic but combining with $\text{coe}(\dot{p}_L)_R$ and $\text{coe}(\dot{p}_R)_L$, as defined by (90) and (91), we can simplify the 4th line of (77) using (92) and (93).

$$\text{coe}(\dot{p}_L)_R = \begin{bmatrix} \text{coe}(\dot{p}_L)_R^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_L)_R^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_L)_R^N \end{bmatrix} \tag{90}$$

$$\text{coe}(\dot{p}_R)_L = \begin{bmatrix} \text{coe}(\dot{p}_R)_L^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_R)_L^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_R)_L^N \end{bmatrix} \tag{91}$$

$$\begin{aligned} \sum_{i=\ell_L^m} \left(\text{coe}(\dot{p}_L)_R^i \cdot \left\{ \dot{p}_L^i \right\} \right) &= \left[\left[\mathbf{G} \cdot \text{coe}(\dot{p}_L)_R \cdot \mathbf{H} \right]^\top \right]_{(m,m)} \cdot \dot{p}_m \\ &= \left[\left[\mathbf{G} \cdot \text{coe}(\dot{p}_L)_R \cdot \mathbf{H} \right]^\top \cdot \dot{\mathbf{P}} \right]_{(m,m)} \end{aligned} \tag{92}$$

$$\begin{aligned} \sum_{i=\ell_R^m} \left(\text{coe}(\dot{p}_R)_L^i \cdot \left\{ \dot{p}_R^i \right\} \right) &= \left[\mathbf{G} \cdot \text{coe}(\dot{p}_R)_L \cdot \mathbf{H} \right]_{(m,m)} \cdot \dot{p}_m \\ &= \left[\mathbf{G} \cdot \text{coe}(\dot{p}_R)_L \cdot \mathbf{H} \cdot \dot{\mathbf{P}} \right]_{(m,m)} \end{aligned} \tag{93}$$

In all simplifications subscript $_{(m)}$ implies element m, subscript $_{(m,m)}$ implies the element in row m and column m, and function $\text{diag}(\mathbf{y})$ creates a diagonal matrix from the elements of vector $\mathbf{y}$. Note that, due to the sign convention, $\text{coe}(p_L)_L^n$, $\text{coe}(p_R)_R^n$, $\text{coe}(\dot{p}_L)_L^n$, and $\text{coe}(\dot{p}_R)_R^n$ are generally negative while $\text{coe}(p_R)_L^n$, $\text{coe}(p_L)_R^n$, $\text{coe}(\dot{p}_R)_L^n$, and $\text{coe}(\dot{p}_L)_R^n$ are generally positive.

With simplifications (80) and (81) the first line of (77) becomes the diagonal terms in $\boldsymbol{\alpha}$, the effects of a mass's position on the force applied to it. With simplifications (88) and (89) the third line of (77) becomes the off-diagonal terms in $\boldsymbol{\alpha}$, the effects of each mass's position on the force applied to other masses. These are shown in (94).

$$\boldsymbol{\alpha} \cdot \mathbf{P} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \text{diag}\left(\left[\text{coe}(p_R)_R \cdot \mathbf{H} \right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \text{coe}(p_L)_L \cdot \boldsymbol{\mu} \right) \\ + \left[\mathbf{G} \cdot \text{coe}(p_L)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \text{coe}(p_R)_L \cdot \mathbf{H} \end{bmatrix} \cdot \mathbf{P} \tag{94}$$

Likewise with simplifications (84) and (85) the second line of (77) becomes the diagonal terms in $\boldsymbol{\beta}$, the effects of a mass's velocity on the force applied to it. With simplifications (92) and (93) the forth line of (77) becomes the off-diagonal terms in $\boldsymbol{\beta}$, the effects of of each mass's velocity on the force applied to other masses. These are shown in (95).

$$\boldsymbol{\beta} \cdot \dot{\mathbf{P}} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \operatorname{diag}\left(\left[\operatorname{coe}(\dot{p}_R)_R \cdot \mathbf{H} \right]^\top \cdot \boldsymbol{\mu} + \mathbf{G} \cdot \operatorname{coe}(\dot{p}_L)_L \cdot \boldsymbol{\mu} \right) \\ + \left[\mathbf{G} \cdot \operatorname{coe}(\dot{p}_L)_R \cdot \mathbf{H} \right]^\top + \mathbf{G} \cdot \operatorname{coe}(\dot{p}_R)_L \cdot \mathbf{H} \end{bmatrix} \cdot \dot{\mathbf{P}} \tag{95}$$

$\mathbf{G}$ and $\mathbf{H}$ contain the direct connections between cells and masses for outgoing and incoming structures respectively. $\mathbf{G}$ can be used directly to index the cells connected to the right of each mass, however $\mathbf{H}$ must be reversed by taking the transpose in order to index the cells connected to the left of each mass.

Combining with $\operatorname{coe}(w_q)_R$ and $\operatorname{coe}(w_q)_L$, as defined by (96) and (97), we can simplify the sixth line of (77) using (98) and (99).

$$\operatorname{coe}(w_q)_R = \begin{bmatrix} \operatorname{coe}(w_q)_R^1 & 0 & 0 \\ 0 & \operatorname{coe}(w_q)_R^n & 0 \\ 0 & 0 & \operatorname{coe}(w_q)_R^N \end{bmatrix} \tag{96}$$

$$\operatorname{coe}(w_q)_L = \begin{bmatrix} \operatorname{coe}(w_q)_L^1 & 0 & 0 \\ 0 & \operatorname{coe}(w_q)_L^n & 0 \\ 0 & 0 & \operatorname{coe}(w_q)_L^N \end{bmatrix} \tag{97}$$

$$\begin{aligned} \sum_{i=\ell_L^m} \left(\operatorname{coe}(w_q)_R^i \cdot \left\{ w_q^i \right\} \right) &= \left[\left[\operatorname{coe}(w_q)_R \cdot \mathbf{H} \right]^\top \right]_{(m,q)} \cdot w_q \\ &= \left[\left[\operatorname{coe}(w_q)_R \cdot \mathbf{H} \right]^\top \cdot \mathbf{J}_q \right]_{(m,q)} \end{aligned} \tag{98}$$

$$\begin{aligned} \sum_{i=\ell_R^m} \left(\operatorname{coe}(w_q)_L^i \cdot \left\{ w_q^i \right\} \right) &= \left[\mathbf{G} \cdot \operatorname{coe}(w_q)_L \right]_{(m,q)} \cdot w_q \\ &= \left[\mathbf{G} \cdot \operatorname{coe}(w_q)_L \cdot \mathbf{J}_q \right]_{(m,q)} \end{aligned} \tag{99}$$

Combining with $\text{coe}\left(\boldsymbol{\Theta}\right)_R$ and $\text{coe}\left(\boldsymbol{\Theta}\right)_L$, as defined by (100) and (101), we can simplify the fifth line of (77) using (102) and (103).

$$\text{coe}\left(\boldsymbol{\Theta}\right)_R = \begin{bmatrix} \text{coe}(\theta_1)_R^1 & \dots & \text{coe}(\theta_v)_R^1 & \dots & \text{coe}(\theta_V)_R^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_R^n & 0 & \text{coe}(\theta_v)_R^n & 0 & \text{coe}(\theta_V)_R^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_R^N & 0 & \text{coe}(\theta_v)_R^N & 0 & \text{coe}(\theta_V)_R^N \end{bmatrix} \tag{100}$$

$$\text{coe}\left(\boldsymbol{\Theta}\right)_L = \begin{bmatrix} \text{coe}(\theta_1)_L^1 & \dots & \text{coe}(\theta_v)_L^1 & \dots & \text{coe}(\theta_V)_L^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_L^n & 0 & \text{coe}(\theta_v)_L^n & 0 & \text{coe}(\theta_V)_L^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{coe}(\theta_1)_L^N & 0 & \text{coe}(\theta_v)_L^N & 0 & \text{coe}(\theta_V)_L^N \end{bmatrix} \tag{101}$$

$$\begin{aligned} \sum_{i=\ell_L^m} \left(\text{coe}(\theta_v)_R^i \cdot \{\theta_v\}\right) &= \left[\mathbf{H}^\top \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_R\right]_{(m,v)} \cdot \theta_v \\ &= \left[\mathbf{H}^\top \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_R \cdot \boldsymbol{\Theta}\right]_{(m,v)} \end{aligned} \tag{102}$$

$$\begin{aligned} \sum_{i=\ell_R^m} \left(\text{coe}(\theta_v)_L^i \cdot \{\theta_v\}\right) &= \left[\mathbf{G} \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_L\right]_{(m,v)} \cdot \theta_v \\ &= \left[\mathbf{G} \cdot \text{coe}\left(\boldsymbol{\Theta}\right)_L \cdot \boldsymbol{\Theta}\right]_{(m,v)} \end{aligned} \tag{103}$$

In these simplifications subscript $_{(m,q)}$ implies the element in row m and column q and subscript $_{(m,v)}$ implies the element in row m and column v.

With simplifications (98) and (99) the sixth line of (77) becomes $\boldsymbol{\gamma}$, the effects of cell internal states on the force applied to each mass. This is shown in (104) and (105).

$$\boldsymbol{\gamma}_q \cdot \mathbf{J}_q = \left[\left[\text{coe}(w_q)_R \cdot \mathbf{H}\right]^\top + \mathbf{G} \cdot \text{coe}(w_q)_L\right] \cdot \mathbf{J}_q \ , \ \forall q \tag{104}$$

$$\boldsymbol{\gamma} \cdot \mathbf{J} = \boldsymbol{\Phi} \cdot \begin{bmatrix} \boldsymbol{\gamma}_1 & \dots & \boldsymbol{\gamma}_q & \dots & \boldsymbol{\gamma}_Q \end{bmatrix} \cdot \mathbf{J} \tag{105}$$

With simplifications (102) and (103) the fifth line of (77) becomes $\boldsymbol{\delta}$, the effects of the cell control input values on the force applied to each mass. These are shown in (106).

$$\boldsymbol{\delta} \cdot \boldsymbol{\Theta} = \boldsymbol{\Phi} \cdot \left[\mathbf{H}^\top \cdot \operatorname{coe}(\boldsymbol{\Theta})_R + \mathbf{G} \cdot \operatorname{coe}(\boldsymbol{\Theta})_L \right] \cdot \boldsymbol{\Theta} \tag{106}$$

This completes the conversion of (77) and the derivation of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\delta}$; the dynamics affecting the forces on the array masses.

The change in each cell internal state is governed by (19), repeated here as (107) for clarity.

$$\begin{aligned} \dot{w}_q^n =& \operatorname{cow}(p_L)_q^n \cdot \{p_L^n\} + \operatorname{cow}(p_R)_q^n \cdot \{p_R^n\} + \operatorname{cow}(\dot{p}_L)_q^n \cdot \{\dot{p}_L^n\} + \operatorname{cow}(\dot{p}_R)_q^n \cdot \{\dot{p}_R^n\} \\ &+ \sum_{v=1}^{V} \left(\operatorname{cow}(\theta_v)_q^n \cdot \{\theta_v\} \right) + \sum_{i=1}^{Q} \left(\operatorname{cow}(w_i)_q^n \cdot \{w_i^n\} \right) \end{aligned} \tag{107}$$

$\mathbf{G}$ and $\mathbf{H}$ contain the direct connections between cells and masses for outgoing and incoming structures respectively. Since (107) deals with the mass position/velocity effects on cell internal states, the structures treated as incoming and outgoing need to be reversed. This means $\mathbf{H}$ can be used directly to index the masses connected to the right of each cell, however $\mathbf{G}$ must be reversed by taking the transpose in order to index the cells connected to the left of each cell.

Combining with $\operatorname{cow}(p_L)_q$ and $\operatorname{coe}(p_R)_q$, as defined by (108) and (109), we can simplify the first and second terms of (107) using (110) and (111).

$$\operatorname{cow}(p_L)_q = \begin{bmatrix} \operatorname{coe}(p_L)_q^1 & 0 & 0 \\ 0 & \operatorname{coe}(p_L)_q^n & 0 \\ 0 & 0 & \operatorname{coe}(p_L)_q^N \end{bmatrix} \tag{108}$$

$$\operatorname{coe}(p_R)_q = \begin{bmatrix} \operatorname{coe}(p_R)_q^1 & 0 & 0 \\ 0 & \operatorname{coe}(p_R)_q^n & 0 \\ 0 & 0 & \operatorname{coe}(p_R)_q^N \end{bmatrix} \tag{109}$$

$$\text{cow}(p_L)_q^n \cdot \{p_L^n\} = \left[\left[\mathbf{G} \cdot \text{cow}(p_L)_q\right]^\top \cdot \mathbf{P}\right]_{(n,m)}, p_m = p_L^n \tag{110}$$

$$\text{cow}(p_R)_q^n \cdot \{p_R^n\} = \left[\text{cow}(p_R)_q \cdot \mathbf{H} \cdot \mathbf{P}\right]_{(n,m)}, p_m = p_R^n \tag{111}$$

Combining with $\text{cow}(\dot{p}_L)_q$ and $\text{coe}(\dot{p}_R)_q$, as defined by (112) and (113), we can simplify the third and forth terms of (107) using (114) and (115).

$$\text{cow}(\dot{p}_L)_q = \begin{bmatrix} \text{coe}(\dot{p}_L)_q^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_L)_q^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_L)_q^N \end{bmatrix} \tag{112}$$

$$\text{coe}(\dot{p}_R)_q = \begin{bmatrix} \text{coe}(\dot{p}_R)_q^1 & 0 & 0 \\ 0 & \text{coe}(\dot{p}_R)_q^n & 0 \\ 0 & 0 & \text{coe}(\dot{p}_R)_q^N \end{bmatrix} \tag{113}$$

$$\text{cow}(\theta_v)_q^n \cdot \{\theta_v\} = \left[\left[\mathbf{G} \cdot \text{cow}(\dot{p}_L)_q\right]^\top \cdot \dot{\mathbf{P}}\right]_{(n,m)}, \dot{p}_m = \dot{p}_L^n \tag{114}$$

$$\text{cow}(\dot{p}_R)_q^n \cdot \{\dot{p}_R^n\} = \left[\text{cow}(\dot{p}_R)_q \cdot \mathbf{H} \cdot \dot{\mathbf{P}}\right]_{(m,m)}, \dot{p}_m = \dot{p}_R^n \tag{115}$$

In these simplifications subscript $_{(n,m)}$ implies the element in row n and column m and subscript $_{(m,m)}$ implies the element in row m and column m.

With simplifications (110) and (111) the first and second terms of (107) become $\boldsymbol{\zeta}$, the effects of array mass positions on cell internal states. With simplifications (114) and (115) the third and forth terms of (107) become $\boldsymbol{\eta}$, the effects of array mass velocities on cell internal states. These are shown in (116), (117), and (118).

$$\boldsymbol{\zeta}_q \cdot \mathbf{P} = \left[\left[\mathbf{G} \cdot \text{cow}(p_L)_q\right]^\top + \text{cow}(p_R)_q \cdot \mathbf{H}\right] \cdot \mathbf{P} \ , \ \forall q \tag{116}$$

$$\boldsymbol{\eta}_q \cdot \dot{\mathbf{P}} = \left[\left[\mathbf{G} \cdot \text{cow}(\dot{p}_L)_q \right]^\top + \text{cow}(\dot{p}_R)_q \cdot \mathbf{H} \right] \cdot \dot{\mathbf{P}} \ , \ \forall q \tag{117}$$

$$\boldsymbol{\zeta} \cdot \mathbf{P} = \begin{bmatrix} \boldsymbol{\zeta}_1 \\ \vdots \\ \boldsymbol{\zeta}_q \\ \vdots \\ \boldsymbol{\zeta}_Q \end{bmatrix} \cdot \mathbf{P} \qquad\qquad \boldsymbol{\eta} \cdot \dot{\mathbf{P}} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \vdots \\ \boldsymbol{\eta}_q \\ \vdots \\ \boldsymbol{\eta}_Q \end{bmatrix} \cdot \dot{\mathbf{P}} \tag{118}$$

No mapping is required between masses and cells for the effects of both cell input forces and cell internal states on cell internal states. As such $\text{cow}\left(\boldsymbol{\Theta}\right)_q$ and $\text{cow}(w_i)_q$, as defined by (119) and (121), we can directly simplify the fifth and sixth terms of (107) using (120) and (122).

$$\text{cow}\left(\boldsymbol{\Theta}\right)_q = \begin{bmatrix} \text{cow}(\theta_1)_q^1 & \dots & \text{cow}(\theta_v)_q^1 & \dots & \text{cow}(\theta_V)_q^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{cow}(\theta_1)_q^n & \dots & \text{cow}(\theta_v)_q^n & \dots & \text{cow}(\theta_V)_q^n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \text{cow}(\theta_1)_q^N & \dots & \text{cow}(\theta_v)_q^N & \dots & \text{cow}(\theta_V)_q^N \end{bmatrix} \tag{119}$$

$$\text{cow}(\theta_v)_q^n \cdot \{\theta_v\} = \left[\text{cow}\left(\boldsymbol{\Theta}\right)_q \cdot \boldsymbol{\Theta} \right]_{(n,v)} \tag{120}$$

$$\text{cow}(w_i)_q = \begin{bmatrix} \text{cow}(w_i)_q^1 & 0 & 0 \\ 0 & \text{cow}(w_i)_q^n & 0 \\ 0 & 0 & \text{cow}(w_i)_q^N \end{bmatrix} \tag{121}$$

$$\text{cow}(w_i)_q^n \cdot \{w_i^n\} = \left[\text{cow}(w_i)_q \cdot \mathbf{J}_q \right]_{(n,n)} \tag{122}$$

In these simplifications subscript $_{(n,v)}$ implies the element in row n and column v and subscript $_{(n,n)}$ implies the element in row n and column n.

With simplification (122) the sixth term of (107) become $\boldsymbol{\iota}$, the effects of cell internal states on cell internal states. With simplification (120) the fifth term of (107) become $\boldsymbol{\kappa}$, the effects of cell control input on cell internal states. These are shown in (123), (124), and (125).

$$\boldsymbol{\iota}_{i,q} \cdot \mathbf{J}_q = \text{cow}(w_i)_q \cdot \mathbf{J}_q \ , \ \forall i, q \tag{123}$$

$$\boldsymbol{\kappa}_q = \text{cow}\left(\boldsymbol{\Theta}\right)_q \ , \ \forall q \tag{124}$$

$$\boldsymbol{\iota} \cdot \mathbf{J} = \begin{bmatrix} \boldsymbol{\iota}_{1,1} & \dots & \boldsymbol{\iota}_{1,q} & \dots & \boldsymbol{\iota}_{1,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{i,1} & \dots & \boldsymbol{\iota}_{i,q} & \dots & \boldsymbol{\iota}_{i,Q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\iota}_{I,1} & \dots & \boldsymbol{\iota}_{I,q} & \dots & \boldsymbol{\iota}_{I,Q} \end{bmatrix} \cdot \mathbf{J} \qquad \boldsymbol{\kappa} \cdot \boldsymbol{\Theta} = \begin{bmatrix} \boldsymbol{\kappa}_1 \\ \vdots \\ \boldsymbol{\kappa}_q \\ \vdots \\ \boldsymbol{\kappa}_Q \end{bmatrix} \cdot \boldsymbol{\Theta} \tag{125}$$

Finally, arranging according to (24) through (31) yields systems dynamics in the form of (16) according to state vector (22) and input vector (23).

3.8 Conclusion

This chapter addressed the challenge of generating the dynamic equations of motion for varied cellular actuator array topologies with a method that accommodates any valid linear internal cell dynamics. This is needed in order to control and simulate the response of actuator arrays when applied to useful systems. The chapter began by describing a Hill-Type model cell and detailing the equations of motion for that cell. This provided an example as a guide for the following sections which developed the dynamic equations of motion for any generalized actuator array with valid linear cell dynamics. The process worked much like a plug-and-chug method starting with the incidence matrices from chapter 2 and ending with the dynamic equations in a standardized linear form. An example was then presented to clarify the process for a small Hill-Type model actuator array. Numerical results were presented following the example to show the validity of the presented

method compared to MatLab's SimMechanics toolbox, a dynamic modeling suite. Finally, the proof and derivation of the presented method was given to show its validity theoretically.

This chapter specifically contributed a mathematical formulation which allows the new incidence matrices from chapter 2 to be combined with the idea of 'stamping' to directly generate the dynamic equations of motion for an actuator array. The process was based on a new analysis of the mathematical structures of actuator arrays which identified the key to simplifying force sums using the linear incidence matrices. The results presented were then verified by numerical simulations which had an error of less than 1%.

CHAPTER IV

OUTSIDE DYNAMICS

This chapter addresses the challenge of adding dynamic elements to actuator arrays which do not fit the normal actuator array model. These could be simple mass loads, multi-array robotic arms, or even hierarchical arrangements of actuator arrays. The chapter demonstrates how outside dynamics can be represented relative to an actuator array and then walks through the mathematics behind adapting the actuator array dynamic equations of motion into a complete system including the outside dynamics. An example is provided where the example array from the previous chapter is attached to a simple mass load, and then numerical results are presented which show a complex three-layer hierarchical camera positioner actuator array's force response under isometric loading. These numerical results are compared against a SimMechanics model to validate the presented theory.

4.1 Outside Dynamics Process

According to state vector $\mathbf{X}$ in (22) and input vector $\mathbf{\Theta}$ in (23), the positions and velocities of the endpoints of a cellular actuator array are treated as inputs in the modeling. This allows the method to be used regardless of dynamics outside of the actuator array. The actuator array could be connected simply to a mass representing a robotic arm segment, or could be connected to more complex structures including springs, dampers, or other dynamic elements.

In cases beyond connecting array endpoints to fixed points (isometric contraction) the endpoint positions and velocities (inputs) are dependent on array force. For this reason standard controls methods dealing with system roots and controllability can produce erroneous results. These cases are referred to as incomplete. Adding outside dynamics can correct this issue by closing the loop between dependent inputs to their associated outputs. Outside dynamics are defined as the system with which an actuator array (or multiple actuator arrays) interacts. The combination of the actuator array dynamics and the outside dynamics forms the full system dynamics. Not every set of outside dynamics will complete a system and in some cases multiple hierarchical layers of outside

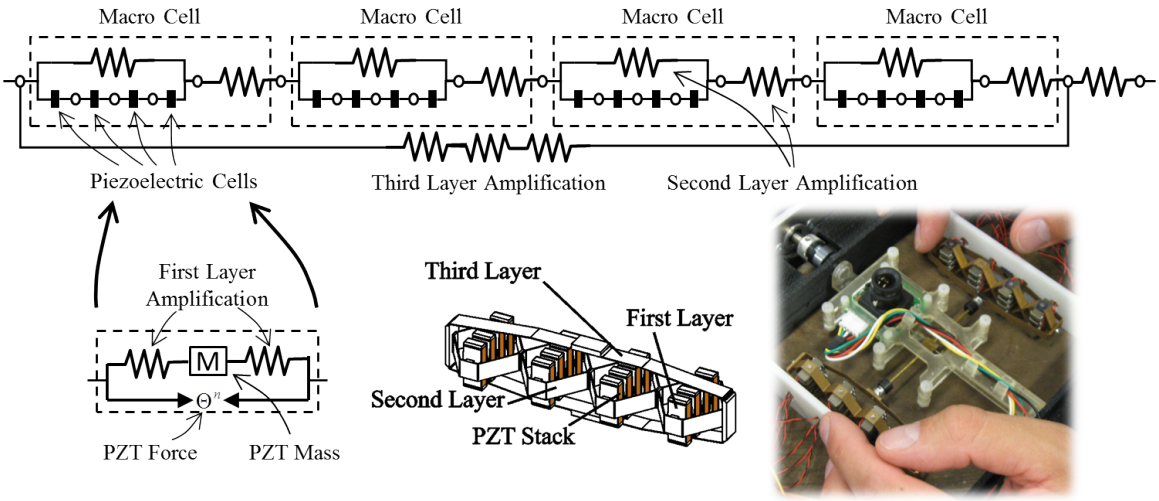


Figure 29: Camera positioner based on piezoelectric (PZT) actuators shown as an example of a multi-layer hierarchical actuator array.

dynamics must be added before the system is complete. Fig. 29 shows a camera positioner based on piezoelectric (PZT) actuators as an example of a multi-layer hierarchical actuator array with outside dynamics.

The outside dynamics acting on a cellular actuator array are assumed to be modeled by (126) and (127) where $\mathbf{Z}^i$ and $\dot{\mathbf{Z}}^i$ are the positions and velocities of the endpoints of array i, $\mathbf{Y}^i$ is the force output from array i, $\boldsymbol{\sigma}$ contains needed external dynamics states, and each F_k is an outside dynamics input.

$$\begin{bmatrix} \vdots \\ \dot{\mathbf{Z}}^i \\ \ddot{\mathbf{Z}}^i \\ \vdots \\ \hline \dot{\boldsymbol{\sigma}} \end{bmatrix} = \left[\begin{array}{ccc|c} & \vdots & \vdots & \vdots \\ \cdots & \boldsymbol{\tau}_{(\mathbf{Z}^i,\mathbf{Z}^j)} & \boldsymbol{\tau}_{\left(\mathbf{Z}^i,\dot{\mathbf{Z}}^j\right)} \quad \cdots & \boldsymbol{\tau}_{(\mathbf{Z}^i,\boldsymbol{\sigma})} \\ \cdots & \boldsymbol{\tau}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Z}^j\right)} & \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^i,\dot{\mathbf{Z}}^j)} \quad \cdots & \boldsymbol{\tau}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma}\right)} \\ & \vdots & \vdots & \vdots \\ \hline \cdots & \boldsymbol{\tau}_{(\boldsymbol{\sigma},\mathbf{Z}^j)} & \boldsymbol{\tau}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j\right)} \quad \cdots & \boldsymbol{\tau}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})} \end{array}\right] \cdot \begin{bmatrix} \vdots \\ \mathbf{Z}^j \\ \dot{\mathbf{Z}}^j \\ \vdots \\ \hline \boldsymbol{\sigma} \end{bmatrix}$$
$$+ \left[\begin{array}{c|ccc} & \vdots & & \\ \mathbf{B}_F & \cdots & \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} & \cdots \\ & \cdots & \boldsymbol{\lambda}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Y}^j\right)} & \cdots \\ & & \vdots & \\ \hline & \cdots & \boldsymbol{\lambda}_{(\boldsymbol{\sigma},\mathbf{Y}^j)} & \cdots \end{array}\right] \cdot \begin{bmatrix} \vdots \\ \mathrm{F}_k \\ \vdots \\ \hline \vdots \\ \mathbf{Y}^j \\ \vdots \end{bmatrix} \tag{126}$$

$$\begin{bmatrix} \vdots \\ \boldsymbol{\Upsilon}^i \\ \vdots \end{bmatrix} = \left[\begin{array}{ccc|c} & \vdots & \vdots & \vdots \\ \cdots & \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Z}^j)} & \boldsymbol{\psi}_{\left(\boldsymbol{\Upsilon}^i,\dot{\mathbf{Z}}^j\right)} \quad \cdots & \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})} \\ & \vdots & \vdots & \vdots \end{array}\right] \cdot \begin{bmatrix} \vdots \\ \mathbf{Z}^i \\ \dot{\mathbf{Z}}^i \\ \vdots \\ \hline \boldsymbol{\sigma} \end{bmatrix}$$
$$+ \left[\begin{array}{c|ccc} & \vdots & & \\ \mathbf{D}_F & \cdots & \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Y}^j)} & \cdots \\ & & \vdots & \end{array}\right] \cdot \begin{bmatrix} \vdots \\ \mathrm{F}_k \\ \vdots \\ \hline \vdots \\ \mathbf{Y}^j \\ \vdots \end{bmatrix} \tag{127}$$

Plugging each array's output forces in to (126) and (127) yields (128) and (129), the combined dynamic equations of motion including the outside dynamics.

$$
\left[\begin{array}{c}
\vdots \\
\dot{\mathbf{Z}}^i \\
\ddot{\mathbf{Z}}^i \\
\vdots \\
\hline
\dot{\boldsymbol{\sigma}} \\
\hline
\vdots \\
\dot{\mathbf{X}}^i \\
\vdots
\end{array}\right]
=
\left[\begin{array}{ccccc|c|ccc}
 & \vdots & \vdots & & & \vdots & & \vdots & \\
\cdots & \boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{Z}^j)} & \boldsymbol{\rho}_{\left(\mathbf{Z}^i,\dot{\mathbf{Z}}^j\right)} & \cdots & & \boldsymbol{\rho}_{(\mathbf{Z}^i,\boldsymbol{\sigma})} & \cdots & \boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{X}^j)} & \cdots \\
\cdots & \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Z}^j\right)} & \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\dot{\mathbf{Z}}^j\right)} & \cdots & & \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma}\right)} & \cdots & \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\mathbf{X}^j\right)} & \cdots \\
 & \vdots & \vdots & & & \vdots & & \vdots & \\
\hline
\cdots & \boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^j)} & \boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j\right)} & \cdots & & \boldsymbol{\rho}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})} & \cdots & \mathbf{0} & \cdots \\
\hline
 & \vdots & \vdots & & & \vdots & & \vdots & \\
\cdots & \boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{Z}^j)} & \boldsymbol{\rho}_{\left(\mathbf{X}^i,\dot{\mathbf{Z}}^j\right)} & \cdots & & \mathbf{0} & \cdots & \boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{X}^j)} & \cdots \\
 & \vdots & \vdots & & & \vdots & & \vdots &
\end{array}\right]
\cdot
\left[\begin{array}{c}
\vdots \\
\mathbf{Z}^j \\
\dot{\mathbf{Z}}^j \\
\vdots \\
\hline
\boldsymbol{\sigma} \\
\hline
\vdots \\
\mathbf{X}^j \\
\vdots
\end{array}\right]
$$

$$
+
\left[\begin{array}{c|ccc|c}
 & & \vdots & & \\
 & \cdots & \boldsymbol{\Lambda}_{(\mathbf{Z}^i,\boldsymbol{\Theta}^j)} & \cdots & \\
 & \cdots & \boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\Theta}^j\right)} & \cdots & \\
 & & \vdots & & \\
\hline
\mathbf{B}_F & \cdots & \boldsymbol{\Lambda}_{(\boldsymbol{\sigma},\boldsymbol{\Theta}^j)} & \cdots & \\
\hline
 & & \vdots & & \\
 & \cdots & \boldsymbol{\Lambda}_{(\mathbf{X}^i,\boldsymbol{\Theta}^j)} & \cdots & \\
 & & \vdots & &
\end{array}\right]
\cdot
\left[\begin{array}{c}
\vdots \\
\mathrm{F}_k \\
\vdots \\
\hline
\vdots \\
\boldsymbol{\Theta}^j \\
\vdots
\end{array}\right]
\tag{128}
$$

$$\begin{bmatrix} \vdots \\ \boldsymbol{\Upsilon}^i \\ \vdots \end{bmatrix} = \left[\begin{array}{ccccc|c|ccc} & \vdots & \vdots & & & \vdots & & \vdots & \\ \cdots & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Z}^j)} & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\dot{\mathbf{Z}}^j)} & \cdots & & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})} & \cdots & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{X}^j)} & \cdots \\ & \vdots & \vdots & & & \vdots & & \vdots & \end{array}\right] \cdot \left[\begin{array}{c} \vdots \\ \mathbf{Z}^i \\ \dot{\mathbf{Z}}^i \\ \vdots \\ \hline \boldsymbol{\sigma} \\ \hline \vdots \\ \mathbf{X}^i \\ \vdots \end{array}\right]$$

$$+ \left[\begin{array}{c|ccc} \mathbf{D}_F & & \vdots & \\ & \cdots & \boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\Theta}^j)} & \cdots \\ & & \vdots & \end{array}\right] \cdot \left[\begin{array}{c} \vdots \\ \mathrm{F}_k \\ \vdots \\ \hline \vdots \\ \boldsymbol{\Theta}^j \\ \vdots \end{array}\right] \tag{129}$$

The elements in (128) and (129) are further defined below to aid in the computation.

4.1.1 ρ Definition

First, $\boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{Z}^j)}$ given by (130), $\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\mathbf{Z}^j)}$ given by (131), $\boldsymbol{\rho}_{(\mathbf{Z}^i,\dot{\mathbf{Z}}^j)}$ given by (132), and $\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\dot{\mathbf{Z}}^j)}$ given by (133) contain the effects of the array j endpoint positions/velocities on array i endpoint positions/velocities. In each case, j can equal i which shows the effects of an array's endpoints on itself.

$$\boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{Z}^j)} = \boldsymbol{\tau}_{(\mathbf{Z}^i,\mathbf{Z}^j)} + \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^j_{ab} & \boldsymbol{\alpha}^j_{bd} \\ \boldsymbol{\alpha}^j_{ac} & \boldsymbol{\alpha}^j_{cd} \end{bmatrix} \tag{130}$$

$$\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\mathbf{Z}^j)} = \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^i,\mathbf{Z}^j)} + \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^j_{ab} & \boldsymbol{\alpha}^j_{bd} \\ \boldsymbol{\alpha}^j_{ac} & \boldsymbol{\alpha}^j_{cd} \end{bmatrix} \tag{131}$$

$$\boldsymbol{\rho}_{(\mathbf{Z}^i,\dot{\mathbf{Z}}^j)} = \boldsymbol{\tau}_{(\mathbf{Z}^i,\dot{\mathbf{Z}}^j)} + \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\beta}^j_{ab} & \boldsymbol{\beta}^j_{bd} \\ \boldsymbol{\beta}^j_{ac} & \boldsymbol{\beta}^j_{cd} \end{bmatrix} \tag{132}$$

$$\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\dot{\mathbf{Z}}^j)} = \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^i,\dot{\mathbf{Z}}^j)} + \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\beta}^j_{ab} & \boldsymbol{\beta}^j_{bd} \\ \boldsymbol{\beta}^j_{ac} & \boldsymbol{\beta}^j_{cd} \end{bmatrix} \tag{133}$$

Since external states and array endpoints are outside dynamics states, not array internal states, their effects on one another remain unchanged. Thus, $\boldsymbol{\rho}_{(\mathbf{Z}^i,\boldsymbol{\sigma})}$ and $\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma})}$ are given by (134). By similar logic, $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^j)}$ and $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j)}$ are given by (135) and $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})}$ is given by (136).

$$\begin{bmatrix} \boldsymbol{\rho}_{(\mathbf{Z}^i,\boldsymbol{\sigma})} \\ \boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma})} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_{(\mathbf{Z}^i,\boldsymbol{\sigma})} \\ \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma})} \end{bmatrix} \tag{134}$$

$$\begin{bmatrix} \boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^j)} & \boldsymbol{\rho}_{(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_{(\boldsymbol{\sigma},\mathbf{Z}^j)} + \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^j_{ab} & \boldsymbol{\alpha}^j_{bd} \\ \boldsymbol{\alpha}^j_{ac} & \boldsymbol{\alpha}^j_{cd} \end{bmatrix} & \boldsymbol{\tau}_{(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j)} + \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\beta}^j_{ab} & \boldsymbol{\beta}^j_{bd} \\ \boldsymbol{\beta}^j_{ac} & \boldsymbol{\beta}^j_{cd} \end{bmatrix} \end{bmatrix} \tag{135}$$

$$\boldsymbol{\rho}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})} = \boldsymbol{\tau}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})} \tag{136}$$

$\boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{X}^j)}$ and $\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^i,\mathbf{X}^j)}$ given by (137) contain the effects of array internal states on array endpoints. Similarly, $\boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{Z}^j)}$ and $\boldsymbol{\rho}_{(\mathbf{X}^i,\dot{\mathbf{Z}}^j)}$ given by (138) contain the effects of array endpoints on array internal states. Since arrays only affect their own endpoints and vice-versa, only cases where i and j are equal need to be considered. Cases where multiple arrays connect to the same point (external states) are considered below, but here each array is considered to have a unique set of endpoints.

$$\begin{bmatrix} \boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{X}^j)} \\ \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\mathbf{X}^j\right)} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \\ \boldsymbol{\lambda}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Y}^j\right)} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\alpha}_b^i & \boldsymbol{\beta}_b^i & \boldsymbol{\gamma}_b^i \\ \boldsymbol{\alpha}_c^i & \boldsymbol{\beta}_c^i & \boldsymbol{\gamma}_c^i \end{bmatrix} & \text{if } i = j \\ \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} & \text{if } i \neq j \end{cases} \tag{137}$$

$$\begin{bmatrix} \boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{Z}^j)} & \boldsymbol{\rho}_{\left(\mathbf{X}^i,\dot{\mathbf{Z}}^j\right)} \end{bmatrix} = \begin{cases} \mathbf{B}_Z^i & \text{if } i = j \\ \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} & \text{if } i \neq j \end{cases} \tag{138}$$

Since array internal states cannot affect other array internal states directly the internal dynamics of the array can be propagated to the outside dynamics. Thus, $\boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{X}^j)}$ is given by (139). Again, only cases where i equals j need to be considered.

$$\boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{X}^j)} = \begin{cases} \mathbf{A}^i & \text{if } i = j \\ \mathbf{0} & \text{if } i \neq j \end{cases} \tag{139}$$

4.1.2 $\boldsymbol{\Lambda}$ Definition

$\boldsymbol{\Lambda}_{(\mathbf{Z}^i,\boldsymbol{\Theta}^j)}$ given by (140) and $\boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\Theta}^j\right)}$ given by (141) contain the effects of the array j inputs on array i endpoint positions/velocities. In general these affects are only present when i equals j.

$$\boldsymbol{\Lambda}_{(\mathbf{Z}^i,\boldsymbol{\Theta}^j)} = \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_c^j \end{bmatrix} \tag{140}$$

$$\boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\Theta}^j\right)} = \boldsymbol{\lambda}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Y}^j\right)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_c^j \end{bmatrix} \tag{141}$$

$\mathbf{\Lambda}_{(\boldsymbol{\sigma},\boldsymbol{\Theta}^j)}$ given by (142) contains the effects of array j inputs on external dynamics states.

$$\mathbf{\Lambda}_{(\boldsymbol{\sigma},\boldsymbol{\Theta}^j)} = \boldsymbol{\lambda}_{(\boldsymbol{\sigma},\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_d^j \end{bmatrix} \tag{142}$$

$\mathbf{\Lambda}_{(\boldsymbol{\Upsilon}^i,\mathbf{X}^j)}$ given by (143) contain the effects of array j inputs on array i internal states. This is only present when i equals j since one array's inputs do not directly affect another array. This does not disallow a single control value from affecting multiple arrays, but that effect is represented separately for each array.

$$\mathbf{\Lambda}_{(\mathbf{X}^i,\boldsymbol{\Theta}^j)} = \begin{cases} \mathbf{B}_{\boldsymbol{\Theta}}^i & \text{if } i = j \\ \mathbf{0} & \text{if } i \neq j \end{cases} \tag{143}$$

4.1.3 $\boldsymbol{\Psi}$ Definition

$\mathbf{\Psi}_{(\mathbf{Z}^i,\mathbf{Z}^j)}$ given by (144) and $\mathbf{\Psi}_{\left(\mathbf{Z}^i,\dot{\mathbf{Z}}^j\right)}$ given by (145) contain the effects of the array j endpoint positions/velocities on array i endpoint positions/velocities.

$$\mathbf{\Psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Z}^j)} = \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Z}^j)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\alpha}_{ab}^j & \boldsymbol{\alpha}_{bd}^j \\ \boldsymbol{\alpha}_{ac}^j & \boldsymbol{\alpha}_{cd}^j \end{bmatrix} \tag{144}$$

$$\mathbf{\Psi}_{\left(\boldsymbol{\Upsilon}^i,\dot{\mathbf{Z}}^j\right)} = \boldsymbol{\psi}_{\left(\boldsymbol{\Upsilon}^i,\dot{\mathbf{Z}}^j\right)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\beta}_{ab}^j & \boldsymbol{\beta}_{bd}^j \\ \boldsymbol{\beta}_{ac}^j & \boldsymbol{\beta}_{cd}^j \end{bmatrix} \tag{145}$$

Since external states and array endpoints are outside dynamics states, not array internal states, their effects on one another remain unchanged. Thus, $\mathbf{\Psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})}$ is given by (146).

$$\mathbf{\Psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})} = \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})} \tag{146}$$

$\mathbf{\Psi}_{(\mathbf{Z}^i,\mathbf{X}^j)}$ and $\mathbf{\Psi}_{\left(\dot{\mathbf{Z}}^i,\mathbf{X}^j\right)}$ given by (147) contain the effects of array internal states on array endpoints. Similarly, $\mathbf{\Psi}_{(\mathbf{X}^i,\mathbf{Z}^j)}$ and $\mathbf{\Psi}_{\left(\mathbf{X}^i,\dot{\mathbf{Z}}^j\right)}$ given by (145) contain the effects of array endpoints on

array internal states. Since arrays only affect their own endpoints and vice-versa, only cases where i and j are equal need to be considered. Cases where multiple arrays connect to the same point (external states) are considered below, but here each array is considered to have a unique set of endpoints.

$$\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\mathbf{X}^j)} = \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\alpha}_b^i & \boldsymbol{\beta}_b^i & \boldsymbol{\gamma}_b^i \\ \boldsymbol{\alpha}_c^i & \boldsymbol{\beta}_c^i & \boldsymbol{\gamma}_c^i \end{bmatrix} \tag{147}$$

4.1.4 Ξ Definition

$\boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\Theta}^j)}$ given by (148) contains the effects of array inputs on the outside dynamics output.

$$\boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\Theta}^j)} = \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^i,\mathbf{Y}^j)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_c^j \end{bmatrix} \tag{148}$$

4.2 Removal of Duplicate States

Cases where two arrays share an endpoint in the outside dynamics will contain a duplicate states using the above process. The duplicate states can be recombined by adding the appropriate rows and columns in (126) and (127). Assuming the duplicate states are $\mathbf{Z}^a$ and $\mathbf{Z}^b$ (also $\dot{\mathbf{Z}}^a$ and $\dot{\mathbf{Z}}^b$), then new state $\mathbf{Z}^{new}$ ($\dot{\mathbf{Z}}^{new}$) is created and (128) and (129) are modified as shown below. Since $\mathbf{Z}^a$ and $\mathbf{Z}^b$ (also $\dot{\mathbf{Z}}^a$ and $\dot{\mathbf{Z}}^b$) are the same state, $\mathbf{Z}^{new}$ ($\dot{\mathbf{Z}}^{new}$), and the system is linear, the change in the new state is the summation of changes of the old states.

The effects of array the endpoints on the new states are $\boldsymbol{\rho}_{(\mathbf{Z}^{new},\mathbf{Z}^j)}$ given by (149), $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\mathbf{Z}^j\right)}$ given by (150), $\boldsymbol{\rho}_{\left(\mathbf{Z}^{new},\dot{\mathbf{Z}}^j\right)}$ given by (151) and $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\dot{\mathbf{Z}}^j\right)}$ given by (152). The effects of $\mathbf{Z}^{new}$ ($\dot{\mathbf{Z}}^{new}$) on itself are handled by the second line in each case in order to account for all possible effects.

$$\begin{aligned} \boldsymbol{\rho}_{(\mathbf{Z}^{new},\mathbf{Z}^j)} &= \boldsymbol{\rho}_{(\mathbf{Z}^a,\mathbf{Z}^j)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^b,\mathbf{Z}^j\right)} \ , \ \forall j \neq \{a,b\} \\ \boldsymbol{\rho}_{(\mathbf{Z}^{new},\mathbf{Z}^{new})} &= \boldsymbol{\rho}_{(\mathbf{Z}^a,\mathbf{Z}^a)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^b,\mathbf{Z}^a\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^b,\mathbf{Z}^a\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^b,\mathbf{Z}^b\right)} \end{aligned} \tag{149}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\mathbf{Z}^{j}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\mathbf{Z}^{j}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\mathbf{Z}^{j}\right)} \ , \ \forall j \neq \{a,b\}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\mathbf{Z}^{new}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\mathbf{Z}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\mathbf{Z}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\mathbf{Z}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\mathbf{Z}^{b}\right)} \tag{150}$$

$$\boldsymbol{\rho}_{\left(\mathbf{Z}^{new},\dot{\mathbf{Z}}^{j}\right)} = \boldsymbol{\rho}_{\left(\mathbf{Z}^{a},\dot{\mathbf{Z}}^{j}\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\dot{\mathbf{Z}}^{j}\right)} \ , \ \forall j \neq \{a,b\}$$

$$\boldsymbol{\rho}_{\left(\mathbf{Z}^{new},\dot{\mathbf{Z}}^{new}\right)} = \boldsymbol{\rho}_{\left(\mathbf{Z}^{a},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\dot{\mathbf{Z}}^{b}\right)} \tag{151}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\dot{\mathbf{Z}}^{j}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\dot{\mathbf{Z}}^{j}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\dot{\mathbf{Z}}^{j}\right)} \ , \ \forall j \neq \{a,b\}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\dot{\mathbf{Z}}^{new}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\dot{\mathbf{Z}}^{b}\right)} \tag{152}$$

The effects of the external states on the new states are $\boldsymbol{\rho}_{(\mathbf{Z}^{new},\boldsymbol{\sigma})}$ given by (153) and $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\boldsymbol{\sigma}\right)}$ given by (154).

$$\boldsymbol{\rho}_{(\mathbf{Z}^{new},\boldsymbol{\sigma})} = \boldsymbol{\rho}_{(\mathbf{Z}^{a},\boldsymbol{\sigma})} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\boldsymbol{\sigma}\right)} \tag{153}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\boldsymbol{\sigma}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\boldsymbol{\sigma}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\boldsymbol{\sigma}\right)} \tag{154}$$

The effects of the array internal states on the new states are $\boldsymbol{\rho}_{(\mathbf{Z}^{new},\mathbf{X}^{j})}$ given by (155) and $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\mathbf{X}^{j}\right)}$ given by (156).

$$\boldsymbol{\rho}_{(\mathbf{Z}^{new},\mathbf{X}^{j})} = \boldsymbol{\rho}_{(\mathbf{Z}^{a},\mathbf{X}^{j})} + \boldsymbol{\rho}_{\left(\mathbf{Z}^{b},\mathbf{X}^{j}\right)} \tag{155}$$

$$\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{new},\mathbf{X}^{j}\right)} = \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{a},\mathbf{X}^{j}\right)} + \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^{b},\mathbf{X}^{j}\right)} \tag{156}$$

The effects of the new states on the external states are $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^{new})}$ given by (157) and $\boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^{new}\right)}$ given by (158).

$$\boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^{new})} = \boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^{a})} + \boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\mathbf{Z}^{b}\right)} \tag{157}$$

$$\boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^{new}\right)} = \boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^{b}\right)} \tag{158}$$

The effects of the new states on the array internal states are $\boldsymbol{\rho}_{(\mathbf{X}^{i},\mathbf{Z}^{new})}$ given by (159) and $\boldsymbol{\rho}_{\left(\mathbf{X}^{i},\dot{\mathbf{Z}}^{new}\right)}$ given by (160).

$$\boldsymbol{\rho}_{(\mathbf{X}^{i},\mathbf{Z}^{new})} = \boldsymbol{\rho}_{(\mathbf{X}^{i},\mathbf{Z}^{a})} + \boldsymbol{\rho}_{\left(\mathbf{X}^{i},\mathbf{Z}^{b}\right)} \tag{159}$$

$$\boldsymbol{\rho}_{\left(\mathbf{X}^{i},\dot{\mathbf{Z}}^{new}\right)} = \boldsymbol{\rho}_{\left(\mathbf{X}^{i},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\rho}_{\left(\mathbf{X}^{i},\dot{\mathbf{Z}}^{b}\right)} \tag{160}$$

The effects of inputs on the new states are accounted for by combining row a and row b in the input matrix as shown in (161) and (162).

$$\boldsymbol{\Lambda}_{(\mathbf{Z}^{new},\boldsymbol{\Theta}^{j})} = \boldsymbol{\Lambda}_{(\mathbf{Z}^{a},\boldsymbol{\Theta}^{j})} + \boldsymbol{\Lambda}_{\left(\mathbf{Z}^{b},\boldsymbol{\Theta}^{j}\right)} \tag{161}$$

$$\boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^{new},\boldsymbol{\Theta}^{j}\right)} = \boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^{a},\boldsymbol{\Theta}^{j}\right)} + \boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^{b},\boldsymbol{\Theta}^{j}\right)} \tag{162}$$

The effects of the new states on the output are accounted for by combining column a and column b in the output matrix as shown in (163) and (164).

$$\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^{i},\mathbf{Z}^{new})} = \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^{i},\mathbf{Z}^{a})} + \boldsymbol{\Psi}_{\left(\boldsymbol{\Upsilon}^{i},\mathbf{Z}^{b}\right)} \tag{163}$$

$$\boldsymbol{\Psi}_{\left(\boldsymbol{\Upsilon}^{i},\dot{\mathbf{Z}}^{new}\right)} = \boldsymbol{\Psi}_{\left(\boldsymbol{\Upsilon}^{i},\dot{\mathbf{Z}}^{a}\right)} + \boldsymbol{\Psi}_{\left(\boldsymbol{\Upsilon}^{i},\dot{\mathbf{Z}}^{b}\right)} \tag{164}$$

4.3 Outside Dynamics Example

Starting from the Hill-Type cell model example with fingerprint (51) given above, a mass is attached to the right end of the array while the left is fixed. The outside dynamics are given by (165) and (166) where M is the mass of the mass attached to the right endpoint of the array. The outputs of the system are the position of the right endpoint of the array and the force acting on the mass at the right endpoint.

$$\begin{aligned}\left[\begin{array}{c}\dot{\mathbf{Z}}^1\\ \hline \ddot{\mathbf{Z}}^1\end{array}\right] &= \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ \hline 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \hline \dot{\mathbf{Z}}^1\end{array}\right] + \left[\begin{array}{cc} 0 & 0\\ 0 & 0\\ \hline 0 & 0\\ 0 & \frac{1}{M}\end{array}\right]\cdot\left[\mathbf{Y}^1\right]\\ &= \left[\begin{array}{c|c}\boldsymbol{\tau}_{(\mathbf{Z}^1,\mathbf{Z}^1)} & \boldsymbol{\tau}_{(\mathbf{Z}^1,\dot{\mathbf{Z}}^1)}\\ \hline \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^1,\mathbf{Z}^1)} & \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^1,\dot{\mathbf{Z}}^1)}\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \hline \dot{\mathbf{Z}}^1\end{array}\right] + \left[\begin{array}{c}\boldsymbol{\lambda}_{(\mathbf{Z}^1,\mathbf{Y}^1)}\\ \hline \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^1,\mathbf{Y}^1)}\end{array}\right]\cdot\left[\mathbf{Y}^1\right]\end{aligned} \tag{165}$$

$$\begin{aligned}\left[\begin{array}{c}\boldsymbol{\Upsilon}^1\\ \hline \boldsymbol{\Upsilon}^2\end{array}\right] &= \left[\begin{array}{cc|cc} 0 & 1 & 0 & 0\\ \hline 0 & 0 & 0 & 0\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \hline \dot{\mathbf{Z}}^1\end{array}\right] + \left[\begin{array}{cc} 0 & 0\\ \hline 0 & 1\end{array}\right]\cdot\left[\mathbf{Y}^1\right]\\ &= \left[\begin{array}{c|c}\boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Z}^1)} & \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^1,\dot{\mathbf{Z}}^1)}\\ \hline \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Z}^1)} & \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^2,\dot{\mathbf{Z}}^1)}\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \hline \dot{\mathbf{Z}}^1\end{array}\right] + \left[\begin{array}{c}\boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Y}^1)}\\ \hline \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Y}^1)}\end{array}\right]\cdot\left[\mathbf{Y}^1\right]\end{aligned} \tag{166}$$

The final dynamic equations of motion will be in the form of (167) and (168).

$$\left[\begin{array}{c}\dot{\mathbf{Z}}^1\\ \ddot{\mathbf{Z}}^1\\ \dot{\mathbf{X}}^1\end{array}\right] = \left[\begin{array}{ccc}\boldsymbol{\rho}_{(\mathbf{Z}^1,\mathbf{Z}^1)} & \boldsymbol{\rho}_{(\mathbf{Z}^1,\dot{\mathbf{Z}}^1)} & \boldsymbol{\rho}_{(\mathbf{Z}^1,\mathbf{X}^1)}\\ \boldsymbol{\rho}_{(\dot{\mathbf{Z}}^1,\mathbf{Z}^1)} & \boldsymbol{\rho}_{(\dot{\mathbf{Z}}^1,\dot{\mathbf{Z}}^1)} & \boldsymbol{\rho}_{(\dot{\mathbf{Z}}^1,\mathbf{X}^1)}\\ \boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{Z}^j)} & \boldsymbol{\rho}_{(\mathbf{X}^i,\dot{\mathbf{Z}}^j)} & \boldsymbol{\rho}_{(\mathbf{X}^i,\mathbf{X}^j)}\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \dot{\mathbf{Z}}^1\\ \mathbf{X}^1\end{array}\right] + \left[\begin{array}{c}\boldsymbol{\Lambda}_{(\mathbf{Z}^1,\boldsymbol{\Theta}^1)}\\ \boldsymbol{\Lambda}_{(\dot{\mathbf{Z}}^1,\boldsymbol{\Theta}^1)}\\ \boldsymbol{\Lambda}_{(\mathbf{X}^1,\boldsymbol{\Theta}^1)}\end{array}\right]\cdot\left[\begin{array}{c}\boldsymbol{\Theta}^1\end{array}\right] \tag{167}$$

$$\left[\begin{array}{c}\boldsymbol{\Upsilon}^1\\ \boldsymbol{\Upsilon}^2\end{array}\right] = \left[\begin{array}{ccc}\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Z}^1)} & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\dot{\mathbf{Z}}^1)} & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{X}^1)}\\ \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Z}^1)} & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\dot{\mathbf{Z}}^1)} & \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{X}^1)}\end{array}\right]\cdot\left[\begin{array}{c}\mathbf{Z}^1\\ \dot{\mathbf{Z}}^1\\ \mathbf{X}^1\end{array}\right] + \left[\begin{array}{c}\boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^1,\boldsymbol{\Theta}^1)}\\ \boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^2,\boldsymbol{\Theta}^1)}\end{array}\right]\cdot\left[\begin{array}{c}\boldsymbol{\Theta}^1\end{array}\right] \tag{168}$$

4.3.1 ρ Components

Using the substitution in (130), $\boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{Z}^j)}$ is given by (169).

$$\begin{aligned}\boldsymbol{\rho}_{(\mathbf{Z}^1,\mathbf{Z}^1)} &= \boldsymbol{\tau}_{(\mathbf{Z}^1,\mathbf{Z}^1)} + \boldsymbol{\lambda}_{(\mathbf{Z}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^1_{ab} & \boldsymbol{\alpha}^1_{bd} \\ \boldsymbol{\alpha}^1_{ac} & \boldsymbol{\alpha}^1_{cd} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} -2 \cdot k_{pe} & 0 \\ 0 & -k_{pe} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}\end{aligned} \tag{169}$$

Using the substitution in (131), $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Z}^j\right)}$ is given by (170).

$$\begin{aligned}\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^1,\mathbf{Z}^1)} &= \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^1,\mathbf{Z}^1)} + \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^1_{ab} & \boldsymbol{\alpha}^1_{bd} \\ \boldsymbol{\alpha}^1_{ac} & \boldsymbol{\alpha}^1_{cd} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{M} \end{bmatrix} \cdot \begin{bmatrix} -2 \cdot k_{pe} & 0 \\ 0 & -k_{pe} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & -\frac{k_{pe}}{M} \end{bmatrix}\end{aligned} \tag{170}$$

Using the substitution in (132), $\boldsymbol{\rho}_{\left(\mathbf{Z}^1,\dot{\mathbf{Z}}^1\right)}$ is given by (171).

$$\begin{aligned}\boldsymbol{\rho}_{(\mathbf{Z}^1,\dot{\mathbf{Z}}^1)} &= \boldsymbol{\tau}_{(\mathbf{Z}^1,\dot{\mathbf{Z}}^1)} + \boldsymbol{\lambda}_{(\mathbf{Z}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\beta}^1_{ab} & \boldsymbol{\beta}^1_{bd} \\ \boldsymbol{\beta}^1_{ac} & \boldsymbol{\beta}^1_{cd} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}\end{aligned} \tag{171}$$

Using the substitution in (133), $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^1,\dot{\mathbf{Z}}^1\right)}$ is given by (172).

$$\begin{aligned}\boldsymbol{\rho}_{(\dot{\mathbf{Z}}^1,\dot{\mathbf{Z}}^1)} &= \boldsymbol{\tau}_{(\dot{\mathbf{Z}}^1,\dot{\mathbf{Z}}^1)} + \boldsymbol{\lambda}_{(\dot{\mathbf{Z}}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\beta}^1_{ab} & \boldsymbol{\beta}^1_{bd} \\ \boldsymbol{\beta}^1_{ac} & \boldsymbol{\beta}^1_{cd} \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{M} \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}\end{aligned} \tag{172}$$

$\boldsymbol{\rho}_{(\mathbf{Z}^i,\boldsymbol{\sigma})}$, $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\boldsymbol{\sigma}\right)}$, $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\mathbf{Z}^j)}$, $\boldsymbol{\rho}_{\left(\boldsymbol{\sigma},\dot{\mathbf{Z}}^j\right)}$, and $\boldsymbol{\rho}_{(\boldsymbol{\sigma},\boldsymbol{\sigma})}$ given by (134), (135), and (136) are not needed in this example as there are no external states beyond those describing the endpoints of the array.

Using the substitution in (137), $\boldsymbol{\rho}_{(\mathbf{Z}^1,\mathbf{X}^1)}$ and $\boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^1,\mathbf{X}^1\right)}$ are given by (173).

$$
\begin{aligned}
\left[\begin{array}{c} \boldsymbol{\rho}_{(\mathbf{Z}^i,\mathbf{X}^j)} \\ \hline \boldsymbol{\rho}_{\left(\dot{\mathbf{Z}}^i,\mathbf{X}^j\right)} \end{array}\right] &= \left[\begin{array}{c} \boldsymbol{\lambda}_{(\mathbf{Z}^i,\mathbf{Y}^j)} \\ \hline \boldsymbol{\lambda}_{\left(\dot{\mathbf{Z}}^i,\mathbf{Y}^j\right)} \end{array}\right] \cdot \left[\begin{array}{c|c|c} \boldsymbol{\alpha}_b^1 & \boldsymbol{\beta}_b^1 & \boldsymbol{\gamma}_b^1 \\ \hline \boldsymbol{\alpha}_c^1 & \boldsymbol{\beta}_c^1 & \boldsymbol{\gamma}_c^1 \end{array}\right] \\
&= \left[\begin{array}{cc} 0 & 0 \\ 0 & 0 \\ \hline 0 & 0 \\ 0 & \frac{1}{M} \end{array}\right] \cdot \left[\begin{array}{ccc|ccc|cccccc} 2\cdot k_{pe} & 0 & 0 & 0 & 0 & 0 & k_{ae} & k_{ae} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_{ae} \end{array}\right] \\
&= \left[\begin{array}{ccc|ccc|cccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{k_{pe}}{M} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{k_{ae}}{M} \end{array}\right]
\end{aligned} \tag{173}
$$

Using the substitution in (138), $\boldsymbol{\rho}_{(\mathbf{X}^1,\mathbf{Z}^1)}$ and $\boldsymbol{\rho}_{\left(\mathbf{X}^1,\dot{\mathbf{Z}}^1\right)}$ are given by (174).

$$\left[\begin{array}{c|c} \boldsymbol{\rho}_{(\mathbf{X}^1,\mathbf{Z}^1)} & \boldsymbol{\rho}_{(\mathbf{X}^1,\dot{\mathbf{Z}}^1)} \end{array}\right] = \left[\begin{array}{cc|cc} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d \\ \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d \end{array}\right]$$

$$= \left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{2\cdot k_{pe}}{\phi} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{k_{pe}}{\phi} & 0 & 0 \\ \frac{-k_{se}}{b_{de}} & 0 & -1 & 0 \\ \frac{-k_{se}}{b_{de}} & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{k_{se}}{b_{de}} & 0 & 1 \end{array}\right] \tag{174}$$

Using the substitution in (139), $\boldsymbol{\rho}_{(\mathbf{X}^1,\mathbf{X}^1)}$ is given by (175).

$$\boldsymbol{\rho}_{(\mathbf{X}^1,\mathbf{X}^1)} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \\ \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s & \boldsymbol{\iota} \end{bmatrix} = \left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -\frac{3\cdot k_{pe}}{\phi} & \frac{k_{pe}}{\phi} & 0 & 0 & 0 & 0 \\ \frac{k_{pe}}{\phi} & -\frac{3\cdot k_{pe}}{\phi} & \frac{2\cdot k_{pe}}{\phi} & 0 & 0 & 0 \\ 0 & \frac{2\cdot k_{pe}}{\phi} & -\frac{3\cdot k_{pe}}{\phi} & 0 & 0 & 0 \\ \hline \frac{k_{se}}{b_{de}} & 0 & 0 & 1 & 0 & 0 \\ \frac{k_{se}}{b_{de}} & 0 & 0 & 1 & 0 & 0 \\ -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 & 0 \\ 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 \\ 0 & -\frac{k_{se}}{b_{de}} & \frac{k_{se}}{b_{de}} & 0 & -1 & 1 \\ 0 & 0 & -\frac{k_{se}}{b_{de}} & 0 & 0 & -1 \end{array}\right] \cdots$$

$$\cdots \left[\begin{array}{cccccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \hline -\frac{k_{ae}}{\phi} & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} & 0 \\ 0 & 0 & 0 & -\frac{k_{ae}}{\phi} & -\frac{k_{ae}}{\phi} & \frac{k_{ae}}{\phi} \\ \hline -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{k_{se}+k_{ae}}{b_{de}} \end{array}\right] \tag{175}$$

4.3.2 $\boldsymbol{\Lambda}$ Components

Using the substitution in (140), $\boldsymbol{\Lambda}_{(\mathbf{Z}^1,\boldsymbol{\Theta}^1)}$ is given by (176).

$$\begin{aligned}
\boldsymbol{\Lambda}_{(\mathbf{Z}^1,\boldsymbol{\Theta}^1)} &= \boldsymbol{\lambda}_{(\mathbf{Z}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_c^j \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{176}$$

Using the substitution in (141), $\boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^1,\boldsymbol{\Theta}^1\right)}$ is given by (177).

$$\begin{aligned}
\boldsymbol{\Lambda}_{\left(\dot{\mathbf{Z}}^1,\boldsymbol{\Theta}^1\right)} &= \boldsymbol{\lambda}_{\left(\dot{\mathbf{Z}}^1,\mathbf{Y}^1\right)} \cdot \begin{bmatrix} \boldsymbol{\delta}_b^j \\ \boldsymbol{\delta}_c^j \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{M} \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{M} \end{bmatrix}
\end{aligned} \tag{177}$$

The substitution in (142), $\boldsymbol{\Lambda}_{(\boldsymbol{\sigma},\boldsymbol{\Theta}^j)}$, is not needed in this example as there are no external states beyond those describing the endpoints of the array.

Using the substitution in (143), $\boldsymbol{\Lambda}_{(\mathbf{X}^1,\boldsymbol{\Theta}^1)}$ is given by (178).

$$\mathbf{\Lambda}_{(\mathbf{X}^1,\mathbf{\Theta}^1)} = \mathbf{B}^1_{\mathbf{\Theta}}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-1}{\phi} & \frac{-1}{\phi} & \frac{1}{\phi} & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{\phi} & \frac{1}{\phi} & \frac{1}{\phi} & 0 \\ 0 & 0 & 0 & \frac{-1}{\phi} & \frac{-1}{\phi} & \frac{1}{\phi} \\ -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{b_{de}} \end{bmatrix} \tag{178}$$

4.3.3 Ψ Components

Using the substitution in (144), $\mathbf{\Psi}_{(\mathbf{\Upsilon}^1,\mathbf{Z}^1)}$ and $\mathbf{\Psi}_{(\mathbf{\Upsilon}^2,\mathbf{Z}^1)}$ are given by (179).

$$\left[\begin{array}{c} \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Z}^1)} \\ \hline \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Z}^1)} \end{array}\right] = \left[\begin{array}{c} \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Z}^1)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^1_{ab} & \boldsymbol{\alpha}^1_{bd} \\ \boldsymbol{\alpha}^1_{ac} & \boldsymbol{\alpha}^1_{cd} \end{bmatrix} \\ \hline \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Z}^1)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\alpha}^1_{ab} & \boldsymbol{\alpha}^1_{bd} \\ \boldsymbol{\alpha}^1_{ac} & \boldsymbol{\alpha}^1_{cd} \end{bmatrix} \end{array}\right]$$

$$= \left[\begin{array}{c} \begin{bmatrix} 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} -2 \cdot k_{pe} & 0 \\ 0 & -k_{pe} \end{bmatrix} \\ \hline \begin{bmatrix} 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -2 \cdot k_{pe} & 0 \\ 0 & -k_{pe} \end{bmatrix} \end{array}\right] \tag{179}$$

$$= \left[\begin{array}{cc} 0 & 1 \\ \hline 0 & -k_{pe} \end{array}\right]$$

Using the substitution in (145), $\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\dot{\mathbf{Z}}^1)}$ and $\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\dot{\mathbf{Z}}^1)}$ are given by (180).

$$\left[\begin{array}{c} \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\dot{\mathbf{Z}}^1)} \\ \hline \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\dot{\mathbf{Z}}^1)} \end{array}\right] = \left[\begin{array}{c} \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^1,\dot{\mathbf{Z}}^1)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\beta}^1_{ab} & \boldsymbol{\beta}^1_{bd} \\ \boldsymbol{\beta}^1_{ac} & \boldsymbol{\beta}^1_{cd} \end{bmatrix} \\ \hline \boldsymbol{\psi}_{(\boldsymbol{\Upsilon}^2,\dot{\mathbf{Z}}^1)} + \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Y}^1)} \cdot \begin{bmatrix} \boldsymbol{\beta}^1_{ab} & \boldsymbol{\beta}^1_{bd} \\ \boldsymbol{\beta}^1_{ac} & \boldsymbol{\beta}^1_{cd} \end{bmatrix} \end{array}\right]$$

$$= \left[\begin{array}{c} \begin{bmatrix} 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \hline \begin{bmatrix} 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \end{array}\right] \tag{180}$$

$$= \left[\begin{array}{cc} 0 & 0 \\ \hline 0 & 0 \end{array}\right]$$

The substitution in (146), $\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^i,\boldsymbol{\sigma})}$, is not needed in this example as there are no external states beyond those describing the endpoints of the array.

Using the substitution in (147), $\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{X}^1)}$ and $\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{X}^1)}$ are given by (181).

$$\left[\begin{array}{c}\boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^1,\mathbf{X}^1)}\\ \hline \boldsymbol{\Psi}_{(\boldsymbol{\Upsilon}^2,\mathbf{X}^1)}\end{array}\right] = \left[\begin{array}{c}\boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^1,\mathbf{Y}^1)} \cdot \left[\begin{array}{c|c|c}\boldsymbol{\alpha}_b^1 & \boldsymbol{\beta}_b^1 & \boldsymbol{\gamma}_b^1\\ \boldsymbol{\alpha}_c^1 & \boldsymbol{\beta}_c^1 & \boldsymbol{\gamma}_c^1\end{array}\right]\\ \hline \boldsymbol{\xi}_{(\boldsymbol{\Upsilon}^2,\mathbf{Y}^1)} \cdot \left[\begin{array}{c|c|c}\boldsymbol{\alpha}_b^1 & \boldsymbol{\beta}_b^1 & \boldsymbol{\gamma}_b^1\\ \boldsymbol{\alpha}_c^1 & \boldsymbol{\beta}_c^1 & \boldsymbol{\gamma}_c^1\end{array}\right]\end{array}\right]$$

$$= \left[\begin{array}{c}\begin{bmatrix}0 & 0\end{bmatrix} \cdot \left[\begin{array}{ccc|ccc|ccccc}2\cdot k_{pe} & 0 & 0 & 0 & 0 & 0 & k_{ae} & k_{ae} & 0 & 0 & 0 & 0\\ 0 & 0 & k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_{ae}\end{array}\right]\\ \hline \begin{bmatrix}0 & 1\end{bmatrix} \cdot \left[\begin{array}{ccc|ccc|ccccc}2\cdot k_{pe} & 0 & 0 & 0 & 0 & 0 & k_{ae} & k_{ae} & 0 & 0 & 0 & 0\\ 0 & 0 & k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_{ae}\end{array}\right]\end{array}\right]$$

$$= \left[\begin{array}{ccc|ccc|cccccc}0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ \hline 0 & 0 & k_{pe} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k_{ae}\end{array}\right] \tag{181}$$

4.3.4 Ξ Definition

Using the substitution in (148), $\boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^1,\boldsymbol{\Theta}^1)}$ and $\boldsymbol{\Xi}_{(\boldsymbol{\Upsilon}^2,\boldsymbol{\Theta}^1)}$ are given by (182).

$$\left[\begin{array}{c} \Xi_{(\Upsilon^1,\Theta^1)} \\ \hline \Xi_{(\Upsilon^2,\Theta^1)} \end{array}\right] = \left[\begin{array}{c} \xi_{(\Upsilon^1,\Upsilon^1)} \cdot \begin{bmatrix} \delta_b^1 \\ \delta_c^1 \end{bmatrix} \\ \hline \xi_{(\Upsilon^2,\Upsilon^1)} \cdot \begin{bmatrix} \delta_b^1 \\ \delta_c^1 \end{bmatrix} \end{array}\right]$$

$$= \left[\begin{array}{c} \begin{bmatrix} 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \\ \hline \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \end{array}\right] \tag{182}$$

$$= \left[\begin{array}{c} 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \\ \hline 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad -1 \end{array}\right]$$

Plugging the substitutions into (167) and (168) yields the final dynamic equations of motion for the system.

4.4 Hierarchical Actuator Arrays

In cases where multiple hierarchical levels are needed to represent an actuator array, the dynamic modeling and outside dynamics methods can be applied recursively. Each inner level should contain one force output on the left and one to the right. Likewise there should be inputs for the positions and velocities of the endpoints of the inner levels. This allows each inner level to be treated as a macro-cell such that it can be combined into larger outer array structures. Fig. 29 shows a camera positioner based on piezoelectric (PZT) actuators as an example of a multi-level hierarchical actuator array. The first level - the base cell - models the PZT along with an amplification structure that attached by the manufacturer (Cedrat). This PZT has a displacement around 30 micrometers, not enough for typical robotics applications. The second level - the inner actuator array - combines four first level cells in series and places them inside a second amplification structure, modeled by two springs. The dynamic modeling method is used to combine the four base cells while the outside dynamics method is used to add the amplification structure. This new macro-cell, now

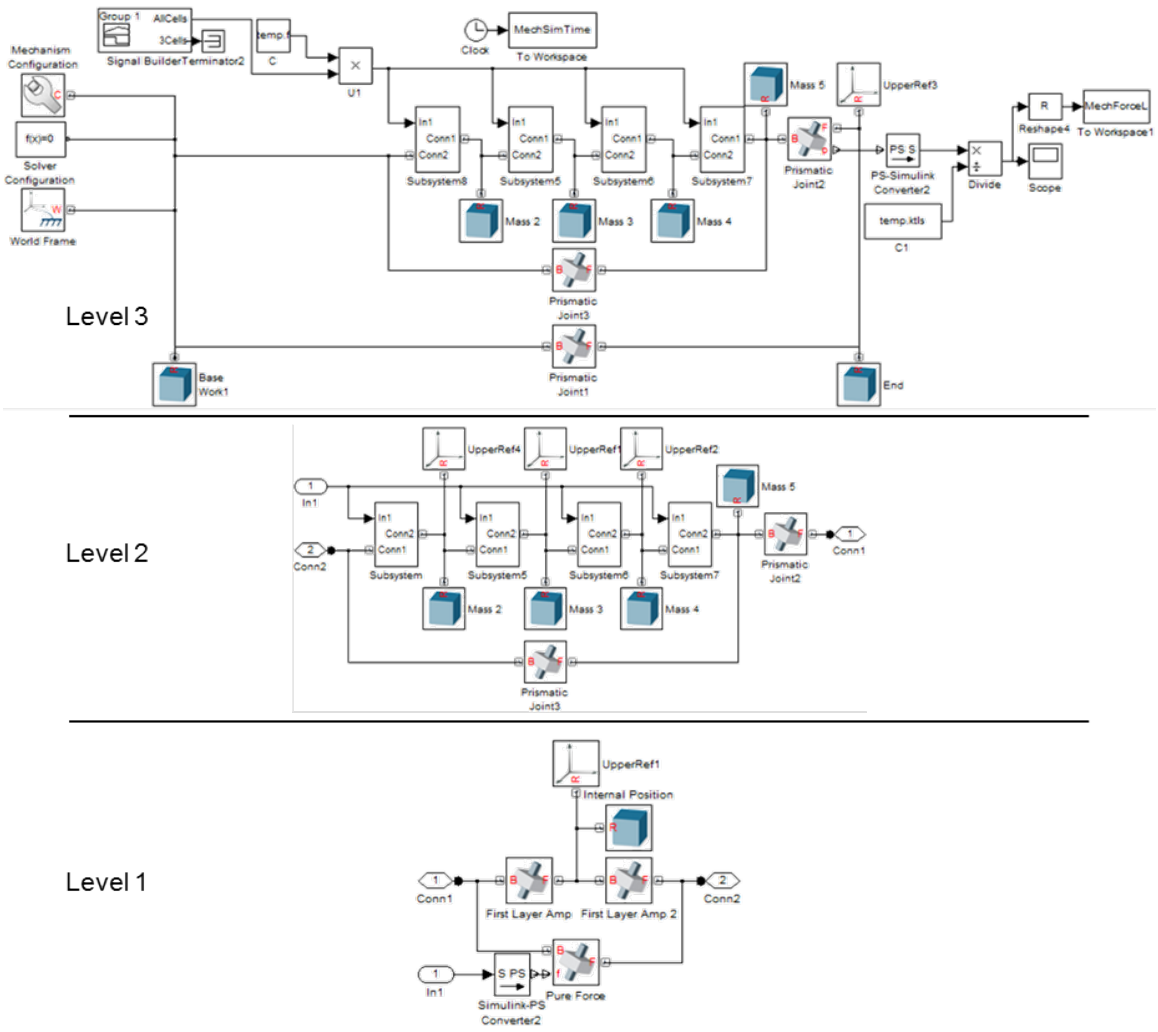


Figure 30: SimMechanics model used to evaluate the piezoelectric based camera positioner actuator array under isometric contraction.

containing four PZT cells, becomes the cell for third level. The third level combines four second level structures in series (using the dynamic modeling method) before adding the third and final amplification structure (using the outside dynamics method). The result is a hierarchical actuator array with a usable displacement for robotics applications.

4.5 Numerical Results

The displacement numerical results (Figures 22, 23, 24, 25, 26, and 27) from the previous chapter show the outside dynamics process used to add a light mass to the end of each trial. The 6 cell trial (Fig. 22) specifically shows the results from the outside dynamics example given in this chapter. The hierarchical example shown in Fig. 29 was also numerically evaluated against

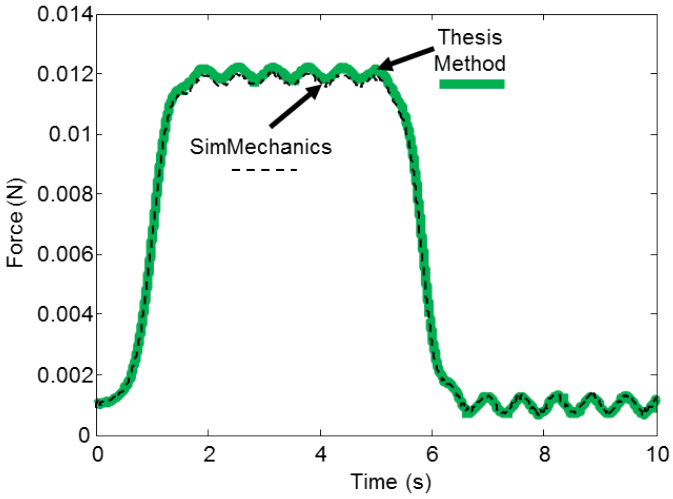


Figure 31: Force at the end of the piezoelectric based camera positioner actuator array under isometric contraction.

SimMechanics. Fig. 30 shows the SimMechanics model used to evaluate the camera positioner. Fig. 31 shows the force at the endpoint of the array as calculated by both methods.

4.6 Conclusion

This chapter addressed the challenge of adding dynamic elements to actuator arrays which do not fit the normal actuator array model. These could be simple mass loads, multi-array robotic arms, or even hierarchical arrangements of actuator arrays. The chapter demonstrated how outside dynamics can be represented relative to an actuator array and then walked through the mathematics behind adapting the actuator array dynamic equations of motion into a complete system including the outside dynamics. An example was provided where the example array from the chapter 3 was attached to a simple mass load, and then numerical results were presented which showed a complex three-layer hierarchical camera positioner actuator array's force response under isometric loading. These numerical results were compared against a SimMechanics model to validate the presented theory.

This chapter specifically contributed a mathematical treatment of the separation of internal array dynamics from outside environment dynamics and a way to mathematically describe the interaction between the two. It also defined a method to use the outside dynamics mathematical treatment to represent complex hierarchical systems. Both the outside dynamics and hierarchical systems

modeling were verified using a numerical model of a camera positioner which had less than a 3% error.

CHAPTER V

STATIC MODELING, SIGNAL DISTRIBUTION, AND NON-LINEAR CELLS

This chapter addresses the three challenges of developing static properties to judge engineering utility of actuator arrays, controlling the redundant actuators in an actuator array, and handling non-linear internal cell dynamics. The chapter begins with the static modeling portion and provides several useful engineering properties to keep in mind when designing actuator arrays. Since most applications of actuator arrays deal with force control, the force function is given particular attention. Actuator array expected force levels are determined by using a modified version of the dynamic modeling equations. The issue of controlling redundant actuators is then addressed. Two signal distribution methods are presented, one based on probability and the other on deterministic grouping and recruitment of cells. Both methods are biologically inspired, one from the stochastic process of calcium diffusion in muscle and the other on motor unit recruitment. Next a numeric comparison of the static and stochastic properties of several example actuator arrays are presented and discussed to show how the properties can guide design. Finally a non-linear extension to the dynamic modeling method presented previously is given which allows structured non-linearities to be handled in the internal cell dynamics.

5.1 Static Modeling

While dynamic equations of motion allow for simulation and more effective control of an actuator array, static properties can be more helpful for determining differences between arrays and to inform actuator array design. Several important properties to consider during the design and analysis process include:

Number of Cells: As the number of cells increases, the actuator array cost increases, the power requirement increases, and the actuator array has a larger volume and mass. Increasing cells also generally increases the actuator array displacement and/or force capacity and decreases the normalized variance. This property needs no calculation beyond counting.

Force Function: The force function is a function of the input and endpoint locations which yields the force an actuator array will provide. This can be created for a multi-dimensional input, but is most useful when used with a forward-loop signal distribution method which reduces the command input to a single value. For a linear array with a fixed length, a single command input of 0 gives the minimum possible force and a command of 1 gives the maximum possible force. The command to mean-force relationship can be chosen to be linear, exponential, or follow some other profile based on the chosen forward-loop signal distribution method. It is preferable to choose a monotonically increasing profile to simplify the controller, however this is not required. A controller uses the force function to achieve a desired mean force output.

Actuator Array Free Displacement: The actuator array free displacement is the difference between the relaxed unforced length and the active unforced length. The relaxed unforced length is the length of the actuator array when all cells are relaxed and no external force is applied. Similarly, the actuated unforced length is the length when all cells are active and no external force is applied.

Variance Function: For probabilistic Gaussian based forward-loop signal distribution methods (discussed later), this is a function of the input probability value and the endpoint locations which yields the expected variance in the output force for an actuator array. While the mean force output will generally remain constant for a constant input probability value, the actual force output will vary over time. The larger the force variance, the farther from the mean value the force is likely to be. Variance will lead to a greater potential for positioning error with open-loop control and reduced stability and accuracy with closed loop controllers. The variance function will be derived and explained further in the forward loop signal distribution chapter.

Robustness (MCLU, WFFF): The worst case failure of a cell is a break, making the broken cell and all cells connected in series to the broken cell seem to vanish from the array. This is considered worse than simply having a non-functional cell since a non-functional cell still has an intact structure. Two robustness measures were developed to characterize actuator array resilience. The "Minimum Cell Loss to Uncontrollability" (MCLU) defines, in the worst case scenario, how many cells would have to break to have zero controllable force capacity. The "Worst Failure Force Function" (WFFF) is the force function an actuator array can achieve after breaking a given number of its most critical cells (the cells which results in the lowest achievable total force once lost). These

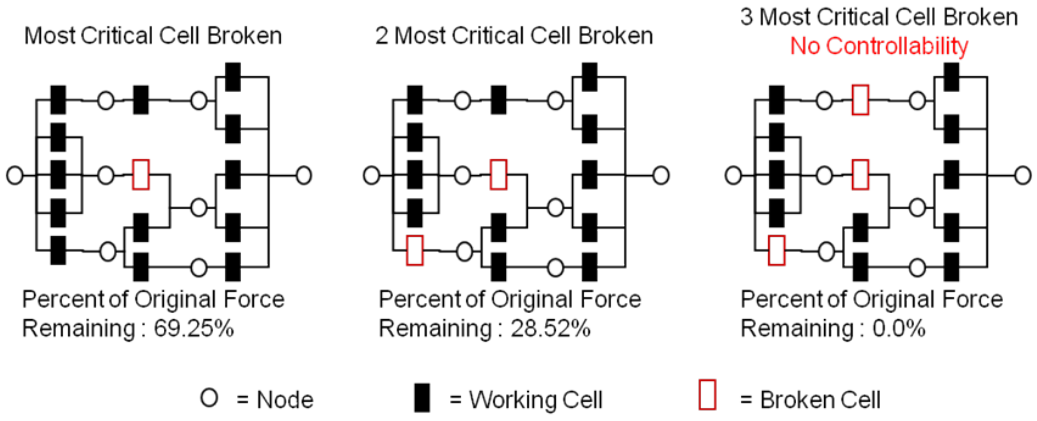


Figure 32: Robustness Measure: Minimum Cell Loss to Uncontrollability = 3.

two measures are shown in Fig. 32. So long as forces on the actuator array are below this value, the actuator array will be able to function after breaking.

5.1.1 Static Force Function

Given a linear actuator with dynamic equations of motion in the form of (16), the force exerted by each endpoint in a static configuration can be calculated by setting the change of internal variables $\dot{\mathbf{X}}$ equal to 0. This simplifies (16) into (183).

$$\begin{aligned} \mathbf{0} &= \mathbf{A} \cdot \mathbf{X} + \mathbf{B} \cdot \mathbf{U} \\ \mathbf{Y} &= \mathbf{C} \cdot \mathbf{X} + \mathbf{D} \cdot \mathbf{U} \end{aligned} \tag{183}$$

Solving the first line for $\mathbf{X}$ then yields (184), the resultant state given a particular input $\mathbf{U}$.

$$\mathbf{X} = -\mathbf{A}^{-1} \cdot \mathbf{B} \cdot \mathbf{U} \tag{184}$$

Plugging (184) into the second line of (183) then yields $\mathbf{Y}$ as shown in given by (185). This is the force function, the resultant force at the endpoints of the array given a particular input $\mathbf{U}$. Further details about how the force function is used will be explained in the forward loop signal distribution chapter.

$$\mathbf{Y} = \left(-\mathbf{C} \cdot \mathbf{A}^{-1} \cdot \mathbf{B} + \mathbf{D}\right) \cdot \mathbf{U} \tag{185}$$

5.2 Forward-Loop Signal Distribution

Just like in human muscle, the robot controller will likely not have direct control over the individual actuation cells in a cellular actuator array, but rather will send a single command signal to the entire array which must then be distributed across the array in a forward-loop manor. The lack of direct control is in order to simplify the controller computational effort and system variables, especially for large numbers of cells, and also carries physical benefits such as simplified wiring. Several methods of forward-loop signal distribution are explored including both probabilistic and deterministic methods.

5.2.1 Probabilistic Signal Distribution

Static results and theorems have been obtained for probabilistic based stochastic broadcast distribution, a method of broadcasting a control percentage across all cells in a muscle and having those cells decide whether to actuate based on comparing an internal random variable to that control signal. This is inspired by the calcium diffusion process in biological muscle where a command signal from nerves connected to the brain is distributed to the individual sarcomeres via the release of calcium ions. The calcium ions are then used in the process of sarcomere contraction. Higher concentrations of the ions causes greater numbers of sarcomeres to contract thus yielding a higher force.

When modeling this effect in the cellular actuator arrays, each cell actuation can be treated as a Bernoulli trial. Due to the linear nature of the system, each cell's actuation has an independent effect on the output force of the system and the cumulative sum of these effects (plus endpoint effects) yields the output force of the system. This was shown in (185), remembering that $\mathbf{U} = \begin{bmatrix} \mathbf{Z} & \mathbf{0} & \mathbf{\Theta} \end{bmatrix}^{\top}$ for the static case.

Given an input probability (p) between 0 and 1, the expected value of the array's output $E\left(\mathbf{Y}\right)$ is given by (186). The $\mathbf{Z}$ components are constant regardless of input probability, but since all cells are independent Bernoulli trials $\mathbf{\Theta}$ (cell input component) scales according to the input probability.

$$E\left(\mathbf{Y}\right) = \left(-\mathbf{C}\cdot\mathbf{A}^{-1}\cdot\mathbf{B}+\mathbf{D}\right)_{(\mathbf{Z},\,\mathbf{0})}\cdot\begin{bmatrix}\mathbf{Z}\\ \mathbf{0}\end{bmatrix} + \left(-\mathbf{C}\cdot\mathbf{A}^{-1}\cdot\mathbf{B}+\mathbf{D}\right)_{(\mathbf{\Theta})}\cdot\begin{bmatrix}\theta_1\cdot p\\ \vdots\\ \theta_v\cdot p\\ \vdots\\ \theta_V\cdot p\end{bmatrix} \tag{186}$$

By the same logic of independent Bernoulli trials, the variance of each cell can be calculated by additionally multiplying the input probability in the expected value of muscle force output by one minus the input probability. This is shown in (187).

$$\sigma\left(\mathbf{Y}\right) = \left(-\mathbf{C}\cdot\mathbf{A}^{-1}\cdot\mathbf{B}+\mathbf{D}\right)_{(\mathbf{\Theta})}\cdot\begin{bmatrix}\theta_1\cdot p\cdot(1-p)\\ \vdots\\ \theta_v\cdot p\cdot(1-p)\\ \vdots\\ \theta_V\cdot p\cdot(1-p)\end{bmatrix} \tag{187}$$

This yields a quadratic function with roots at input probability values of zero and one, and a maximum at an input probability value of one half. Additional details are in [49].

Deterministic methods deal with grouping cells and having a predetermined recruitment function govern which cells are activated for a given input. This can be shown by example. Fig. 28 shows step responses when the cells in the array are non-uniformly grouped as Group 1={Cell 2}, Group 2={3}, Group 3={4, 8}, Group 4={9, 10, 11}, and Group 5={6, 12, 13, 14}. Cells 1, 5, 7 were not used in this case. The progressive activation inspired by the size principle [23] achieves a fine resolution for a small motor command and a more coarse resolution for a larger command. This is known in physiology as signal dependent noise, a commonly observed yet otherwise unexplained phenomena in biological muscle [37]. Additionally, the function of the signal dependent noise can be controlled by choosing appropriate actuator array topologies and/or different cell force levels yielding linear, exponential, or any number of other monotonically increasing profiles. Note that there is a number of ways in connecting actuator units, mechanically, electrically, or a mixture of

these, to realize such actuator groupings. The responses in Fig. 28 show fluctuations from convergence values, mainly due to under damped modes in the system. Muscle forces during contractions also show fluctuations [37]. Although it has not been fully investigated yet, the identification of such fluctuations would provide an interesting insight into the neuromotor variability. These ideas are explored more in [92].

5.3 Static Properties

Fig. 33 shows five actuator arrays which are analyzed; the results are shown below. In the analysis, the force function was selected such that the maximum output force was 1 when the actuator array was not stretched. The cells making up each topology are identical and are based on a simple spring acted upon by a pure force.

Fig. 34 shows the force probability density functions (PDFs) for actuator array D. This graph shows the different forces which can be immediately achieved along with the probability that each point is reached given an input probability. The strips are due to the discrete nature of on-off control of each individual cells. Fig. 35 shows the normalized force function given an input probability and the current array length, and this function is identical for any actuator array using stochastic broadcast distribution. This figure shows how the mean force changes linearly with respect to input probability and displacement. Fig. 36 shows the force variance curves for all of the actuator arrays. Interestingly, the same variance curves were obtained for arrays A, B, and C. Increasing the number of cells in an actuator topology generally results in a lower variance, as in E. Additionally, when cells do not uniformly carry the internal forces, as observed for D, they have a higher variance. The largest variance for any actuator array is obtained when the command signal, or input probability, is 0.5. This can be seen in the examples and is understandable since all of the actuator units will undoubtedly turn on if an input probability of 1 is given and off if an input probability of 0 is given.

Tab. 8 shows the number of cells, actuator array travel, required actuator force/displacement (RAD/RAF), minimum cell loss to uncontrollability (MCLU), and the percentage reduction between the original force function and the worst failure force function (WFFF) after breaking the most critical cell. A had the lowest RAD and RAF while B had the highest. A also had the lowest displacement while B had the highest, showing the trade-off between strength and displacement.

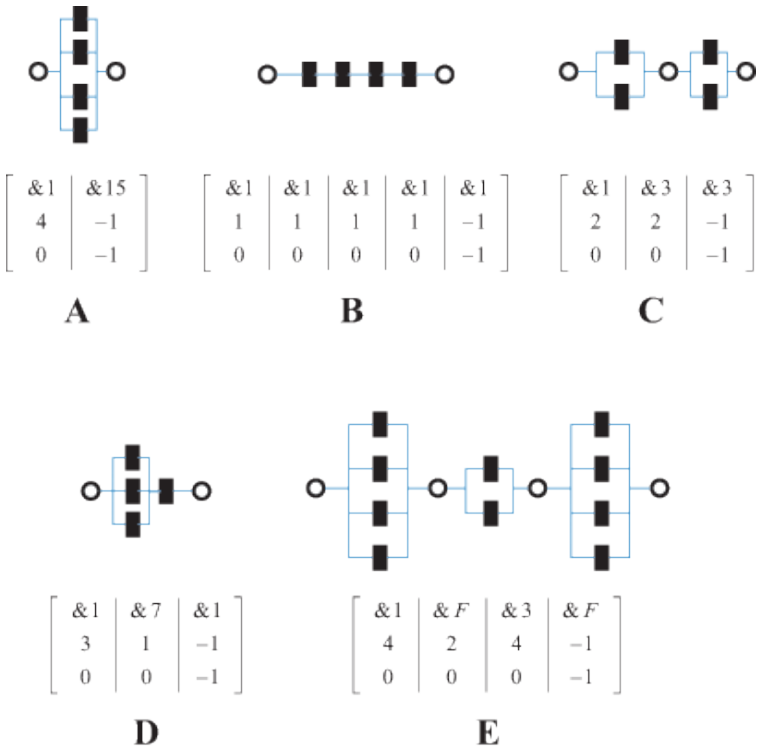


Figure 33: Example actuator arrays analyzed using the fingerprint method

B had a much longer travel than the other systems due to each cell having an additive effect on the overall displacement and since each actuator had to have a greater displacement in order to get the desired force function. Array A had the greatest robustness from both the MCLU and WFFF measurements while C and E were tied as the second best. B and D were both tied for the worst case, despite D having a parallel structure. This shows how having a parallel structure can improve robustness, but does not necessarily do so. E was shown as a contrast to the 4 cell examples to show how additional cells can improve the variance curve, strength (or RAF), or displacement. It also shows how cases with fewer cells can have more desirable properties, such as strength (lower RAF) when E is compared to A or displacement when E is compared to B.

5.4 Non-Linear Cells

The dynamic modeling method can be extended to represent arrays containing cells with non-linear elements so long as only linear elements connect to the endpoints of the cell. This allows for modifying $\dot{\omega}_q^n$ in (19) to replace the summations with a non-linear function $f\left(t, \mathbf{J}, \boldsymbol{\Theta}\right)$ of t (time), $\mathbf{J}$

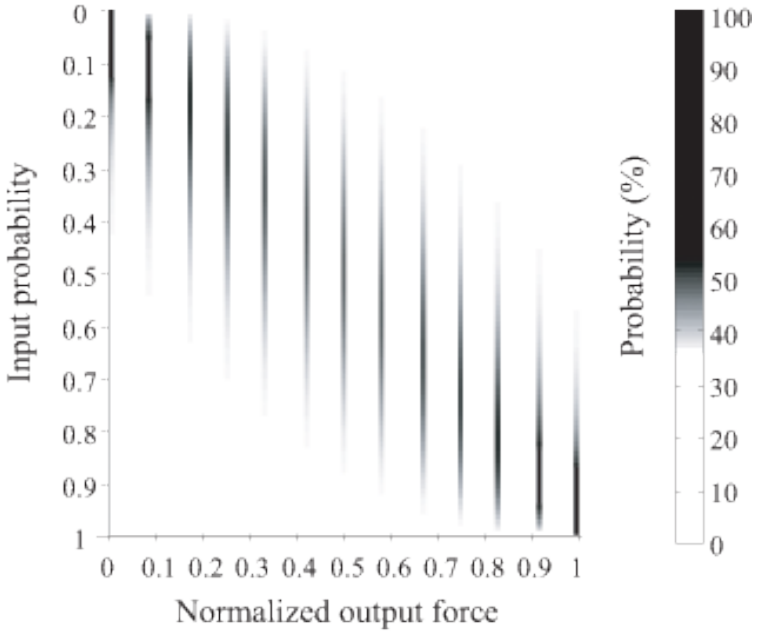


Figure 34: Force probability density function for actuator array D.

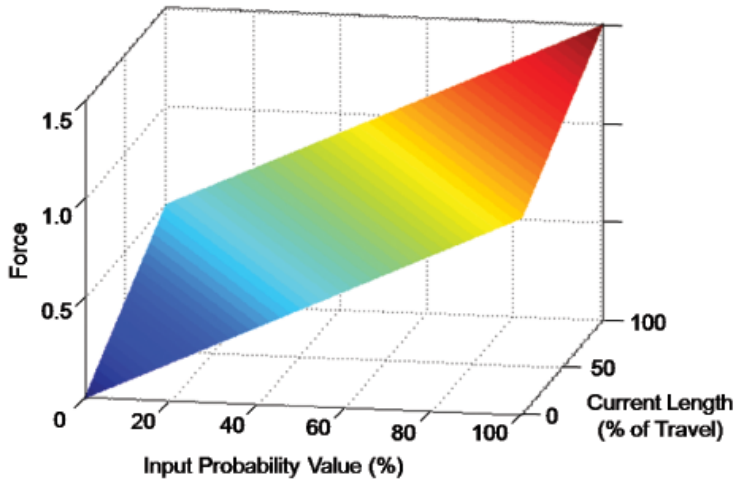


Figure 35: Normalized force function for all array topologies

(the internal variables of the array), and Θ (control inputs for the array). (188) shows the modified equation.

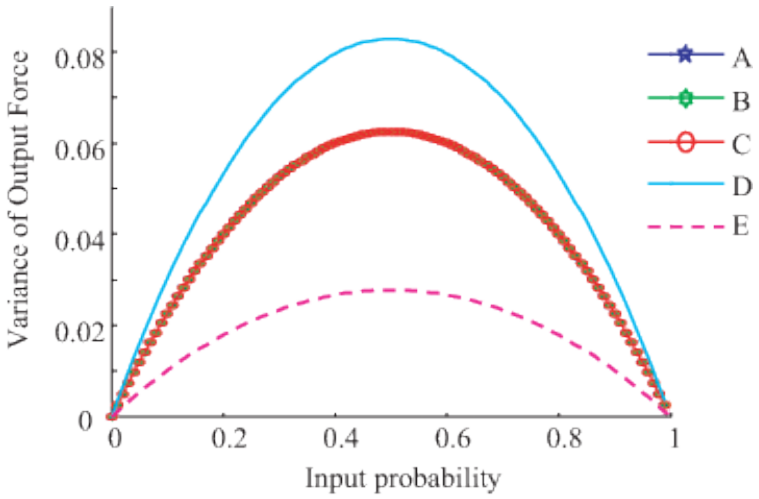


Figure 36: Force variance of example arrays using normalized force function

Table 8: Example Actuator Array Properties

Topology	A	B	C	D	E
Number of Cells	4	4	4	4	10
Actuator Array Travel	0.125	2.0	0.5	0.667	0.5
Required Actuator Displacement	0.250	1.000	0.500	0.667	0.333
Minimum Cell Loss to Uncontrollability	4	1	2	1	2
% Reduction After Losing Most Critical Cell	25.0 %	100.0%	33.3%	100.0%	33.3%

$$\begin{aligned}\dot{w}_q^n =& \text{cow}(p_L)_q^n \cdot \{p_L^n\} + \text{cow}(p_R)_q^n \cdot \{p_R^n\} + \text{cow}(\dot{p}_L)_q^n \cdot \{\dot{p}_L^n\} \\ &+ \text{cow}(\dot{p}_R)_q^n \cdot \{\dot{p}_R^n\} + f\left(t, \mathbf{J}, \boldsymbol{\Theta}\right)\end{aligned} \tag{188}$$

This allows (32) to be represented according to (189) which allows for the limited non-linearity.

$$\begin{aligned}\left[\begin{array}{c}\dot{\mathbf{P}}_v\end{array}\right] &= \left[\begin{array}{cc|c}\mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s\end{array}\right] \cdot \left[\begin{array}{c}\mathbf{P}_v \\ \hline \mathbf{J}\end{array}\right] + \left[\begin{array}{cccc|c}\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d & \boldsymbol{\delta}_s\end{array}\right] \cdot \left[\begin{array}{c}\mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta}\end{array}\right] \\ \left[\begin{array}{c}\dot{\mathbf{J}}\end{array}\right] &= \left[\begin{array}{cc}\boldsymbol{\zeta}_s & \boldsymbol{\eta}_s\end{array}\right] \cdot \left[\begin{array}{c}\mathbf{P}_v\end{array}\right] + \left[\begin{array}{cccc}\boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d\end{array}\right] \cdot \left[\begin{array}{c}\mathbf{Z} \\ \dot{\mathbf{Z}}\end{array}\right] + f\left(t, \mathbf{J}, \boldsymbol{\Theta}\right)\end{aligned} \tag{189}$$

5.4.1 Non-Linear Derivation

The non-linear representation in (189) can be derived directly from (32). First separate equations for $\dot{\mathbf{P}}_v$ from those for $\dot{\mathbf{J}}$ according to (190).

$$\left[\begin{array}{c} \dot{\mathbf{P}}_v \end{array}\right] = \left[\begin{array}{cc|c} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \\ \hline \mathbf{J} \end{array}\right] + \left[\begin{array}{cccc|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d & \boldsymbol{\delta}_s \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta} \end{array}\right]$$

$$\left[\begin{array}{c} \dot{\mathbf{J}} \end{array}\right] = \left[\begin{array}{cc|c} \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s & \boldsymbol{\iota} \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \\ \hline \mathbf{J} \end{array}\right] + \left[\begin{array}{cccc|c} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d & \boldsymbol{\kappa} \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta} \end{array}\right] \tag{190}$$

Due to the endpoints of the cell being connected only to linear elements, the masses of the array are only connected to linear elements. The equations for $\dot{\mathbf{P}}_v$ are thus unchanged and can always be represented by linearly. The equations for $\dot{\mathbf{J}}$, however, will contain both linear and non-linear elements. Since the endpoints of the cell are connected only to linear elements, the effect of $\mathbf{P}_v$, $\mathbf{Z}$, and $\dot{\mathbf{Z}}$ on internal cell states will always be linear. The effects of $\mathbf{J}$ and $\boldsymbol{\Theta}$, however, can be non-liner. (191) separates the always linear components from those which can be non-linear.

$$\left[\begin{array}{c} \dot{\mathbf{P}}_v \end{array}\right] = \left[\begin{array}{cc|c} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \boldsymbol{\alpha}_s & \boldsymbol{\beta}_s & \boldsymbol{\gamma}_s \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \\ \hline \mathbf{J} \end{array}\right] + \left[\begin{array}{cccc|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\alpha}_a & \boldsymbol{\alpha}_d & \boldsymbol{\beta}_a & \boldsymbol{\beta}_d & \boldsymbol{\delta}_s \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \\ \hline \boldsymbol{\Theta} \end{array}\right]$$

$$\left[\begin{array}{c} \dot{\mathbf{J}} \end{array}\right] = \left[\begin{array}{cc} \boldsymbol{\zeta}_s & \boldsymbol{\eta}_s \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{P}_v \end{array}\right] + \left[\begin{array}{cccc} \boldsymbol{\zeta}_a & \boldsymbol{\zeta}_d & \boldsymbol{\eta}_a & \boldsymbol{\eta}_d \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{Z} \\ \dot{\mathbf{Z}} \end{array}\right] + (\boldsymbol{\iota} \cdot \mathbf{J} + \boldsymbol{\kappa} \cdot \boldsymbol{\Theta}) \tag{191}$$

Replacing $(\boldsymbol{\iota} \cdot \mathbf{J} + \boldsymbol{\kappa} \cdot \boldsymbol{\Theta})$ with non-linear function $f(t, \mathbf{J}, \boldsymbol{\Theta})$ completes the derivation and yields (189).

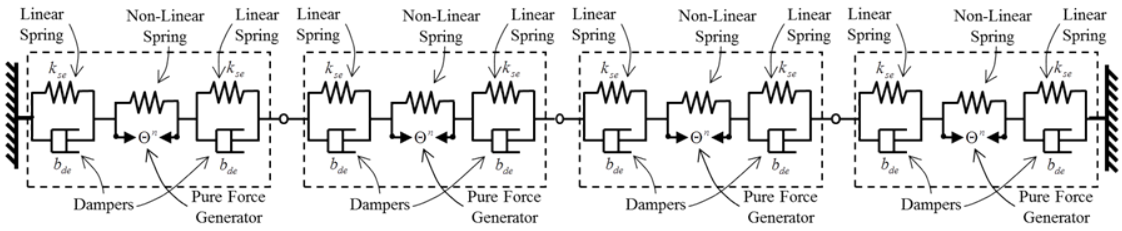


Figure 37: Force variance of example arrays using normalized force function

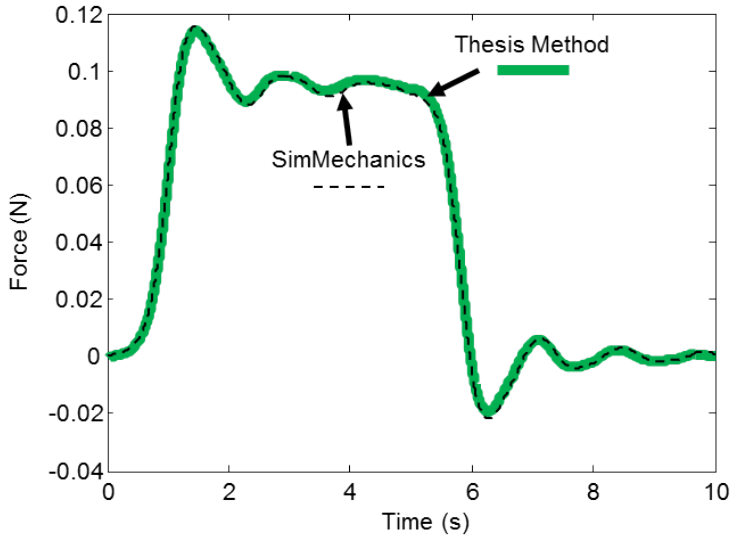


Figure 38: Force variance of example arrays using normalized force function

5.4.2 Non-Linear Numerical Example

The non-linear extension shown above was applied to a numerical example consisting of an actuator with non-linear spring dynamics sandwiched between two sets of linear springs and dampers as shown in Fig. 37. The force delivered by the non-linear spring was $F = K_{nl} \cdot X^2$ where K_{nl} is the spring constant scaling factor and X is the displacement of the spring from equilibrium. This represents a strain stiffening spring. The 4 non-linear cells were placed in series to form the array and the array was simulated in an isometric contraction. The results from simulating the equations of motion from the thesis and those generated from MatLab SimMechanics are shown in Fig. 38.

5.5 Conclusion

This chapter addressed the three challenges of developing static properties to judge engineering utility of actuator arrays, controlling the redundant actuators in an actuator array, and handling non-linear internal cell dynamics. The chapter began with the static modeling portion and provided several useful engineering properties to keep in mind when designing actuator arrays. Since most applications of actuator arrays deal with force control, the force function was given particular attention. Actuator array expected force levels were determined by using a modified version of the dynamic modeling equations. The issue of controlling redundant actuators was then addressed. Two signal distribution methods were presented, one based on probability and the other on deterministic grouping and recruitment of cells. Both methods were biologically inspired, one from the stochastic process of calcium diffusion in muscle and the other on motor unit recruitment. Next a numeric comparison of the static and stochastic properties of several example actuator arrays were presented and discussed to show how the properties can guide design. Finally a non-linear extension to the dynamic modeling method presented previously was given which allows structured non-linearities to be handled in the internal cell dynamics.

This chapter specifically contributed a definition of key static properties for use in actuator array design and developed a method to calculate these properties from the dynamic equations developed in 3. This chapter also showed that the variability in actuator array force when controlled stochastically is topologically dependent, a new result. Finally, this chapter developed a new method that can accommodate well-structured non-linear elements in actuator array cells by separating these elements from the outputs of the cell.

CHAPTER VI

EXPERIMENTAL VALIDATION

This chapter addresses the challenge of experimentally validating the theory presented in this thesis. Three actuator arrays, shown in Fig. 39, ranging from mostly static to highly dynamic were used to test the presented methods. The chapter begins by describing the static solenoid array which was used to validate the static properties theory and demonstrated a stochastic (probabilistic) signal distribution method. This is followed by describing the damped shape memory alloy (SMA) actuator array. This is the most practical of the physical arrays and follows a bio-inspired design based on biological muscle structure. Several guidelines and suggestions are presented which could aid future designers building actuator arrays. The dynamic SMA actuator array is then presented which is used to validate the dynamic aspects of the presented theory. Finally, the results from the two SMA actuator arrays are presented and analyzed.

6.1 Static Solenoid Array Experiment

In order to show the validity of static and stochastic analysis methods, a physical solenoid based actuator array was constructed and compared against numerical results. The during each experiment

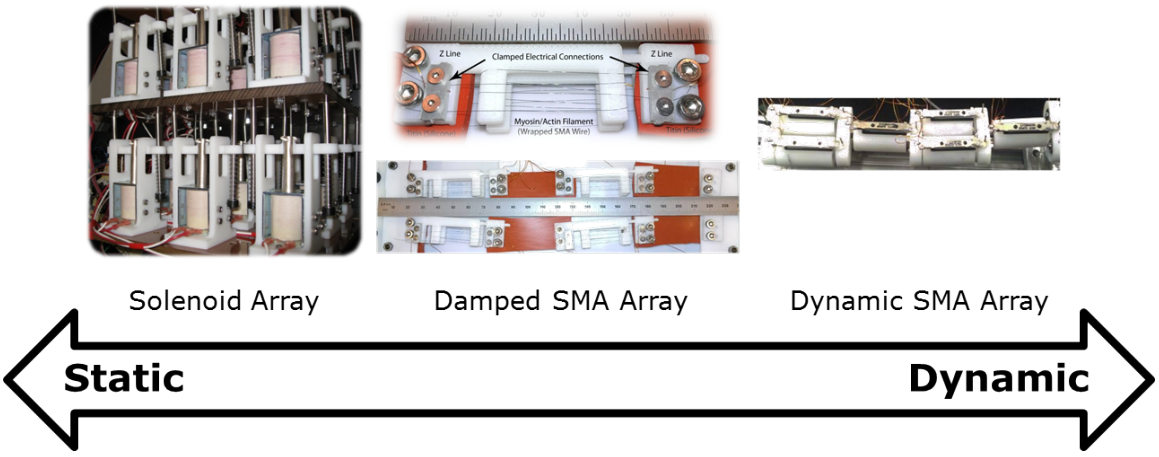
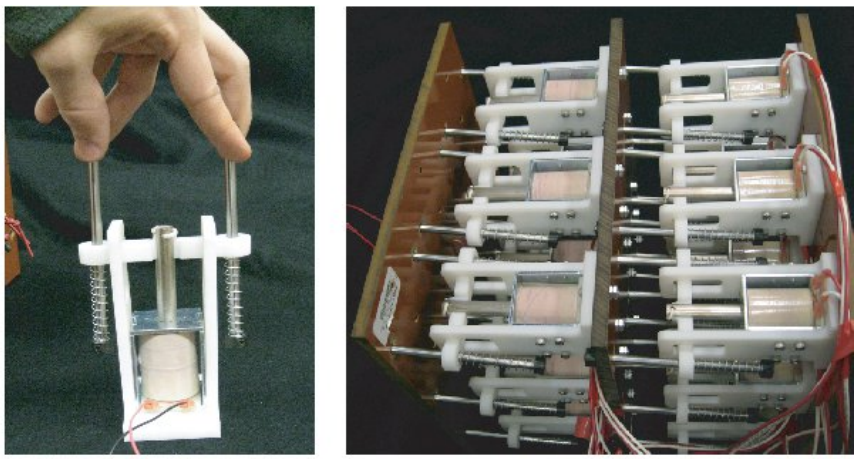


Figure 39: Experimental actuator arrays used to validate the theory presented in this thesis.



(a) Solenoid actuator cell (b) Actuator array

Figure 40: Solenoid actuator array experimental setup.

with the solenoid array, the system was allowed to reach equilibrium before recording data. The control system used with the solenoid array contained a simple stochastic process, thus the results allow for analyzing muscle properties under the influence of a stochastic controller.

6.1.1 Experimental Setup

The array consisted of eighteen Magnet-Schultz of America (MSA) S-06683 solenoids, in a custom designed housing. The housing and connections between the solenoids and springs were printed using a Dimension bst 768 rapid prototyper. The solenoids each acted against two Lee Spring Co. LC 026EE 14M springs set in parallel. Figure 40 (a) shows one solenoid cell. The cells were arranged into layers of three cells by three cells with all of the cells on each layer acting in parallel. Two of these layers were placed in series with one another giving the actuator array shown in Fig. 40 (b). Each cell had a combined spring constant of 438.7 N/m and maximum displacement of 2.5 cm which gave the effect of a 11.1 N pure force acting across the 438.7 N/m spring. Given the layout of the cells, this gave the actuator array a holding force capability of 200 N at its minimum length with an overall displacement capability of 5 cm.

The actuator array was controlled using LabView as shown in Fig. 41. The cells were grouped according to Fig. 42 so that all eighteen cells were controlled with eight signal input channels. This grouping eliminated any moments which could have been generated due to an actuator firing on

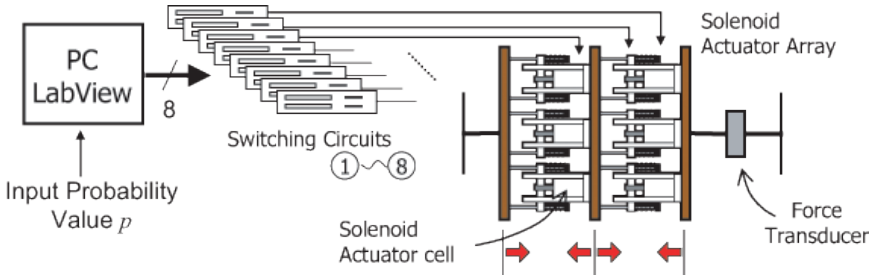


Figure 41: Solenoid actuator array control diagram

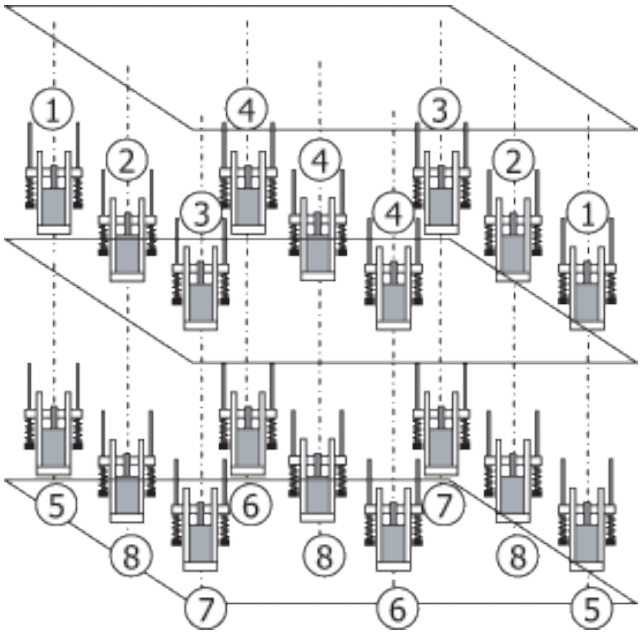


Figure 42: Coupling among solenoid actuator array units. Units with the same number were treated as belonging to the same cell.

only one side of the actuator array, which was necessary to keep with the one-dimensional analysis presented in this paper due to the significant solenoid mass. The simplified fingerprint of this actuator array, given the control groupings, is represented by (192).

$$\left[\begin{array}{c|c|c} \# & \&\mathrm{F} & \&\mathrm{F} \\ 4 & 4 & \# \end{array}\right]. \tag{192}$$

The array took in an input probability value and, for each trial, generated eight random control values. If the random control value for a control channel was less than the input probability value, that control channel would actuate; otherwise it would relax.

Due to solenoid high mass and slow response time, they were not suitable to test using a high frequency stochastic control scheme. Additionally, due to limitations in the solenoid's force capacity at maximum stroke length, each trial began with the actuator array at its unforced relaxed length. An input probability value was input and the cells actuated accordingly. The actuator array was then stretched to a preset length and the force required to achieve this stretch was measured using an OmegaDyne LCM703-50 force transducer. This allowed the experiment to test the actuator based on the holding force capacity of the solenoid as opposed to the significantly lower force at maximum stroke length.

6.1.2 Solenoid Array Experimental Results

Trials were run with input probability values ranging from zero to one with a step size of 0.1 giving a total of 110 trials. The minimum force required to achieve the stretch when no cells were active was subtracted from all data points and the data was normalized such that the maximum force required to achieve the stretch when all cells were active was 1. In Fig. 43(a) and (b), the normalized mean and variance of the forces were calculated from the results for each of the 11 input probability values and are shown along with the expected theoretical results.

The experimental mean force and force variance curves closely matched the values calculated using the presented theory. Errors did exist in both but were expected due to the inherent error in using a probability driven system and due to inaccuracies in the force transducer. Had more trials been run at each of the input probability values, the experimental results would have more closely approached the theoretical values. The errors were low, and the results validate the presented static and stochastic analysis.

6.1.3 Static and Stochastic Analysis

The linearity and constant slope for each actuator array allows a controller to directly command a desired mean force as long as the minimum and maximum forces at the endpoints of the actuator array's travel have been identified and the controller has knowledge of the actuator array's current length. Additionally, combining displacement and force sensors with the actuator array, the current belief of a manipulator and its payload can also be continuously updated by higher level controllers and learning techniques for more accurate manipulation. With these sensors, the actuator array can

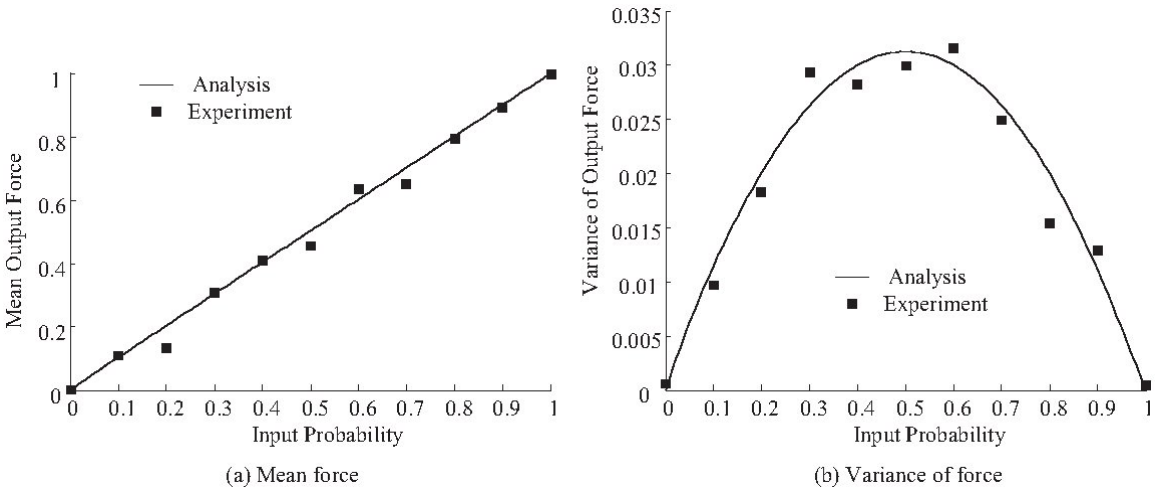


Figure 43: Solenoid actuator array experimental results. (a) shows the mean force generated, which was highly linear. (b) shows the variance in the force which was close to the values calculated using the fingerprint method.

also be continuously calibrated so that it remains robust and accurate even despite multiple cell failures.

The variance of the muscle is a measure of the difference between the force commanded and the force delivered by an actuator array at any given point in time. When cells change position rapidly (200 Hz for piezoelectric actuator arrays), each instantaneous error provides only a small error in the overall impulse delivered and averages out. Increasing the number of cells in an actuator array generally lowers the normalized variance; however variance also scales with increasing force capacity. Cells which uniformly carry the internal forces generally have a lower variance. These observed properties are able to aid designers in creating arrays, however the reverse analysis is especially challenging due to the many design variables which must be manipulated to give any desired set of properties. The possible set of properties is also discrete (non-continuous) for any array when individual cell properties are pre-selected.

It should be noted that while one design criteria may be to minimize the variance, which can made arbitrarily small by adding additional cells and adjusting the topology, arrays could be likewise constructed with larger variances to aid the development of theories regarding natural movement. This can provide evidence in support or against optimization techniques, such as minimization of

jerk or minimization of control signal, contributing to research on the generation of natural movement and serving as a platform for future robots using natural movements.

The relationship between the input probability and variance shown in Figs. 36 and 43(b) is quadratic and does not monotonically increase, while the variance is expected to increase monotonically and proportionally with respect to the command input in biological systems [31]. Maximum voluntary force may not be the maximum that a biological muscle can potentially generate; voluntary forces may be limited at the command level. The association between the input probability and variance could be approximated by a proportional function for a limited range, for example, for $p = 0.0 \sim 0.4$.

6.2 Dynamic SMA Experiments

6.2.1 Damped SMA Array

The damped SMA array experiment shows a practical silicone-based SMA actuator array design that was inspired by the M-lines and Z-lines of sarcomeres in biological muscle and the interaction between the myosin and actin fibers which generate displacement and subsequent contractile force. The actuation of biological muscle is provided by myosin interacting with actin in the presence of ATP to create contractile displacement between the Z-line and M-line on either side of the sarcomere. Additionally, the strands of actin and the connecting titin fibers are flexible allowing the contractile displacement of the sarcomere to be translated into contractile force when under the influence of an external load or blocking force. Individual sarcomeres have a consistent contraction length and additional overall displacement, or similarly force, is built by recruiting or activating more sarcomeres in the muscle, not by further displacing a given sarcomere.

The silicone rubber based SMA actuator array like its Miga NanoMuscle counterpart capitalizes on this idea of recruitment to negate the effect of hysteresis, an inherent drawback with SMA actuators along with many other linear actuation technologies. The SMA wires behave as a hybrid between actin and myosin fibers. When heated via electrical current, the SMA shifts from its martensite phase to its austenite phase which is roughly 3-4% less than its length in the martensite phase. The silicone connecting structure takes the place of the flexible actin and titin fibers and thus provides the translation between displacement and force based on external loading conditions and

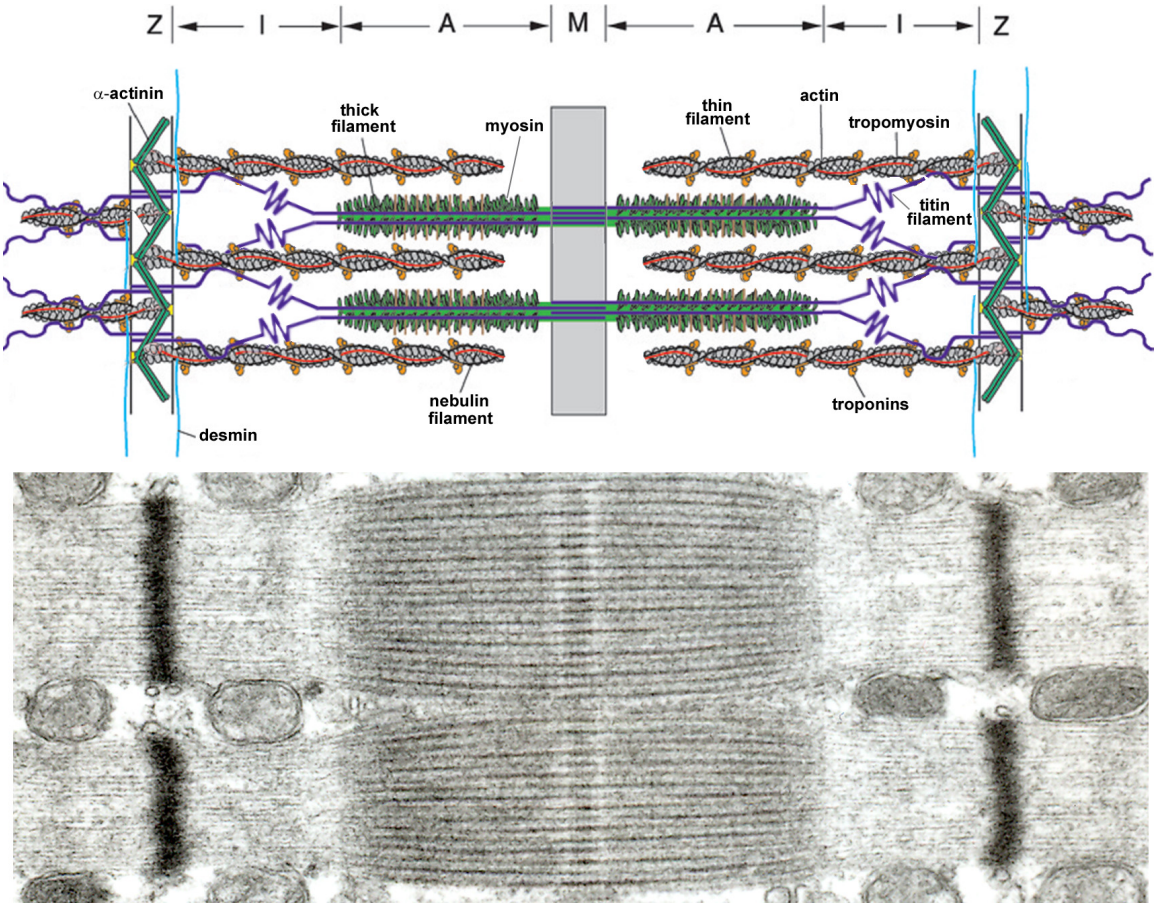


Figure 44: Molecular representation and structure of a sarcomere. Image taken from [60] and used with permission under the creative commons license.

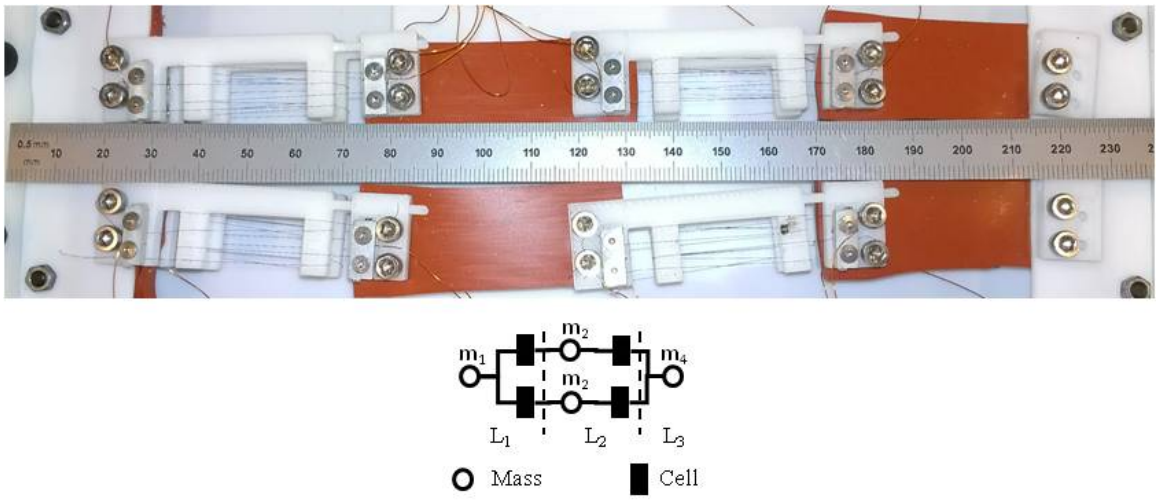


Figure 45: Silicone rubber based actuator array.

the length of other cells. As more force or displacement is desired, additional cells are activated causing additional pre-loading of the silicone 'springs' and thus additional force or displacement to the external environment. In the original design a rigid element existed between two sets of SMA wires for each cell to better emulate the M-line of biological muscle, however this was dropped for construction simplicity as a single set of myosin/actin fibers acting against two Z-line elements has roughly the same dynamics as two sets with a rigid M-line between them.

Flexinol 100 m HT SMA wire produced by Dynalloy Inc. was selected as the actuation material primarily for its balance between force and cooling time. Larger diameter SMA wires can generally produce higher forces, but take significantly longer to cool while smaller diameter wires produce significantly less force but also cool more rapidly. Future research will look into combining larger and smaller diameter wire cells in the same actuator array similarly to how biological muscle has slow and fast twitch sarcomeres, however this was not considered in the current work. The spring constant of the Flexinol 100 m HT SMA wire was determined experimentally to be 2.54N/mm for the 75mm length of wire used in the final cell array. This was determined using a Futek LBS200 five-pound force sensor and two .02mm resolution micro-positioners with the SMA wire in its relaxed Martensite phase. While the spring constant of the SMA wire does vary greatly between the martensite (relaxed) and austenite (active) phase, the relaxed martensite phase is lower than the active austenite phase and both values are an order of magnitude above that of the silicone connecting structure. For this reason, the value of the martensite phase was taken as the actuator's stiffness. During the testing phase the wire was subjected to approximately 8.45 N and remained in its elastic phase. This is roughly five times the force an individual SMA wire experiences in the final actuator array which suggests that, coupled with the silicone connective structure, the SMA wires remain linear and resistant to breaking. In order to increase the force output of a single cell, four wires were used in parallel. Likewise to increase the displacement, the wires were wrapped around a Z-line bracket as shown in Fig. 46. The silicone chosen needed to have a stiffness less than that of the additive SMA wires force for a given displacement so that the SMA wires can always achieve their full displacement. The actual stiffness of the silicone connecting structure is dependent both on the stress-strain properties of the silicone and the geometry of the connecting structure. Several silicone rubber sheets were tested experimentally using the same sensor and micro-positioners as for the

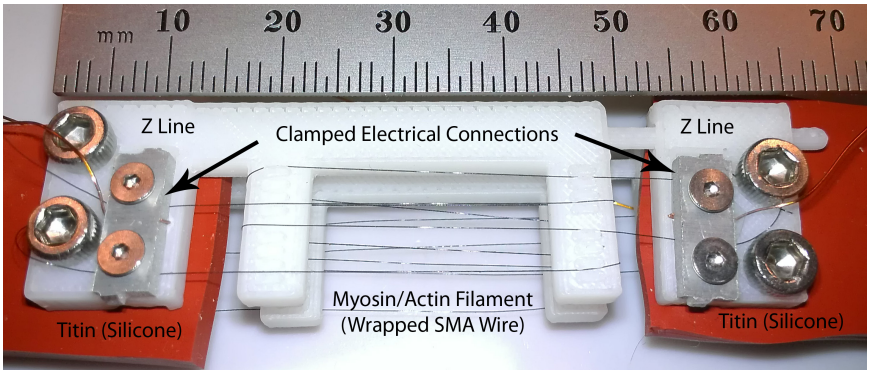


Figure 46: Biological similarity of the silicone rubber based actuator array cell.

SMA testing, and it was determined that shore A 20 durometer silicone rubber provided favorable properties. A 30mm x 62mm x 6mm section of silicone rubber, the size used for the final cells, was tested producing a final spring constant of 0.611 N/mm. A stiffer silicone sheet could have been used and would have increased the strength of the resultant muscle significantly, however this was not done for experimental setup reliability reasons. The Z-line brackets were printed with a 3D rapid prototyper using ABS plastic for ease of construction and plastic's inherent electrical resistivity.

While the four parallel SMA wires in each cell for this experimental setup have a theoretical force capability of 18 N and a displacement capability of 5 mm, manufacturing errors caused the actual displacement to be 0.7 mm and thus the effective control force of the cell was given by $(0.7mm) \cdot (2.54N/mm) = 1.7N$. The effective applied force of the cell was therefore $(0.7mm) \cdot (0.61N/mm) = 0.41N$. While this is significantly lower than what is needed for implementation in a full scale muscle, a different choice of silicone sheet and an improved design and manufacturing process will dramatically increase the properties. The current SMA wires also take roughly a second to fully heat and to fully cool, however introducing additional cooling mechanisms such as forced air or submersion in an oil bath can significantly improve this as well. As the focus of the current research was to validate the presented theory and show viability of the experimental setup, these changes were left to future work.

The properties of a given actuator array are also highly dependent on the topology and choice of internal cell actuation properties (such as SMA wire size). The presented design is highly modular and, depending on the chosen topology, highly robust. Cells are connected together through the silicone structure thus reconfiguring the muscle is as easy as cutting/punching out a different silicone

shape and connecting the Z-line elements to it. Likewise, if a cell fails it can simply be disconnected from the surrounding cells and replaced. Parallel structures add robustness meaning if a cell fails either due to an electrical failure (the cell goes dead but still remains intact) or mechanical failure (cell physically breaks leaving zero stiffness) the remaining cells are still able to carry force. Thus an arm using the muscle may have a reduced force capacity but will still be able to function. Take as an illustrative example a robot arm hammering in a dangerous environment. If a critical number of cells fail, the robot may not be able to continue hammering but would still have the force capacity to secure the area and move itself to safety in order to be repaired. Current systems fall limp after a single motor failure requiring an additional robot or human to 'rescue' the robot, putting additional equipment, and possibly lives, at risk.

6.2.2 Dynamic SMA Array

In order to validate the dynamic aspects of the presented theory, a highly dynamic (under damped) actuator array was created. Each cell in the array consists of two Miga NanoMuscle 704 SMA actuators mounted to a ABS rapid prototyped shell. A compression spring (series elastic element) inside the shell connected to a rod which goes through the spring and out to form the outgoing connection point. The arms of the SMA actuators form the incoming connection points, and are activated together to cancel any moment that would otherwise be generated. The SMA actuators are significantly stiffer than the compression spring, but they do have some flexibility to them. They also take time to heat when activated and time to cool when deactivated. For these two reasons, they are approximated by a pure force acting across a stiff spring (actuator elastic element) and a damper. Magnet wires were chosen to power the SMA actuators in order to reduce the effect of wire stiffness on the results, however some stiffness remains and was accounted for as the parallel elastic element. The mass, stiffnesses, damping (measured from response time), and force output for each cell were measured using a calibrated Futek 9 newton force transducer powered by an Omega signal conditioner and the results were viewed on a LeCroy 600Mhz scope. The spring constants for the parallel elastic, series elastic, and SMA actuator were found to be 20 N/m, 386.52 N/m, and 1000 N/m respectively by stretching each between two 0.02mm resolution micro-positioners and measuring the output force. The damping coefficient was found to be 500 N/(m/s), and the mass of each

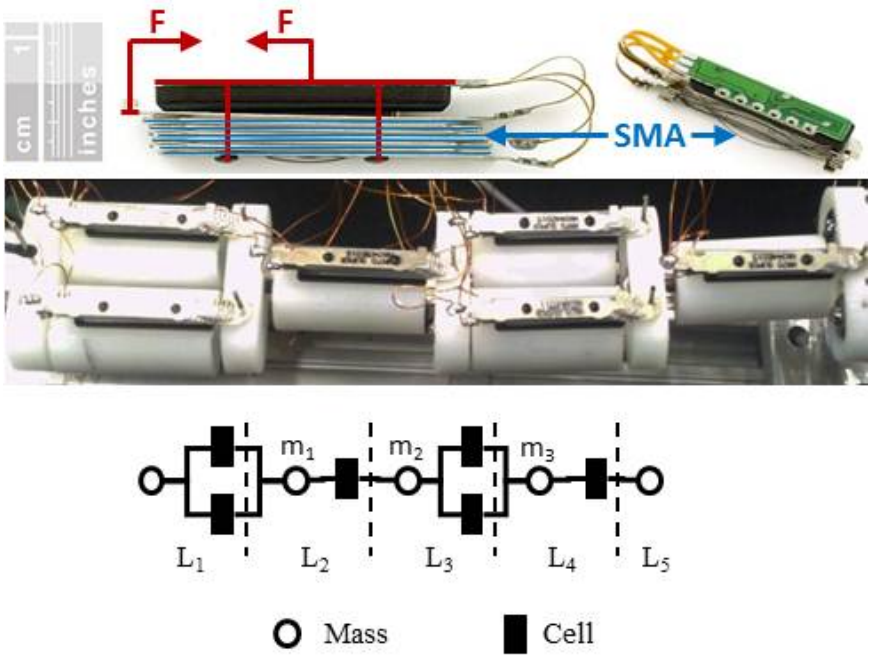


Figure 47: Physical 6 cell array actuator used for experimental validation.

mass was 15.83 grams. 6 cells were set up in an isometric contraction arrangement as shown in Fig. 47. The damper connected in parallel with the series elastic element in the theory was negligible with this experimental setup, allowing the dynamics to play a greater role in validating the theory. Additionally, the simple Miga NanoMuscle SMA actuators can be represented accurately for the all-on all-off case as a spring and damper in parallel with a constant pure-force acting on it as an input. This allows for the contractile element to be less of a black box, again allowing for further validation of the theory. The setup, however, was extremely fragile and while it produced useful dynamic validation data it is not suggested for use in practical applications. Over the course of the experiment the Miga NanoMuscles tore themselves apart repeatedly. As such the experimental results from this setup were somewhat limited in scope as the authors did not wish to spend a large amount of time and resources to continually rebuild the array.

6.2.3 Damped SMA Array Experimental Results

Experimental validation of the above theory was carried out in two stages. The first utilized a highly damped silicone rubber based actuator array shown in Fig. 46 to extend the results of the solenoid experiment to the time domain. In the solenoid experiment the topology of actuator arrays

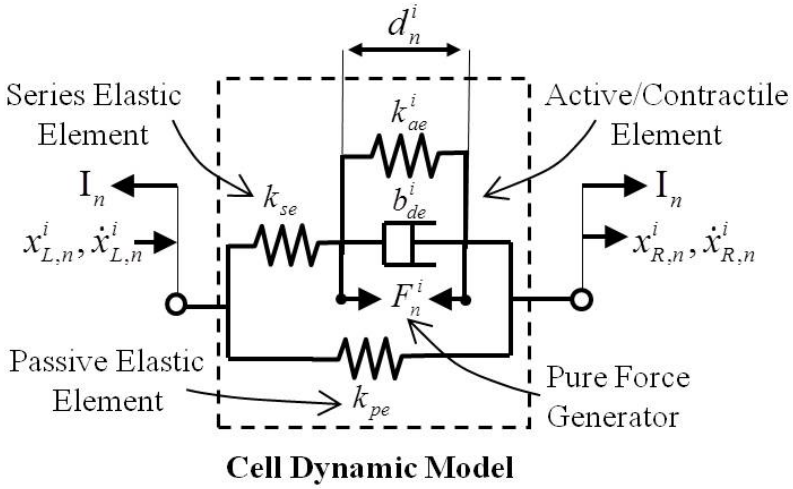


Figure 48: Miga NanoMuscle 704 SMA actuator cell model.

was used to determine the final actuator array properties, such as force level, but all experimental justification used steady state values. In the dynamic experiment with the silicone based SMA array, the four cells were activated from rest at four different levels: 1 cell on, 2 cells on, 3 cells on, and all cells. For each, the force activation profile was the Sigmoid function $\left(F \cdot \left(\frac{1}{1+e^{-5 \cdot t+6}}\right)\right)$ followed by a similar inverse Sigmoid function $\left(F \cdot \left(1 - \frac{1}{1+e^{-5 \cdot (t-t_d)+6}}\right)\right)$ where t_d is the deactivation time and F is the force of the cell. This function was determined experimentally by measuring the SMA wire contraction profile. Fig. 49 shows sample results with the simulation results overlaid. Due to the system's high natural frequency and high damping, the results for the silicone based SMA actuator do not highlight dynamic effects, but they do show that predicted force levels of the theory and simulation match those of the experimental results.

6.2.4 Dynamic SMA Array Experimental Results

The dynamic SMA array shows the dynamic effects much more clearly than the damped array. Fig. 50 shows force results for the dynamic SMA array. In the figure, the upper trace shows the dynamic response when all cells were activated while the lower traces shows the response when half the cells were activated. For each the cells were activated for 3 seconds and then deactivated. As can be seen in the graph, the results track the simulated values very closely including rises and falls in the graph due to individual cell inertia. The track is not perfect, partly because all mass elements were assumed to be uniform in the simulated results whereas in the physical system they

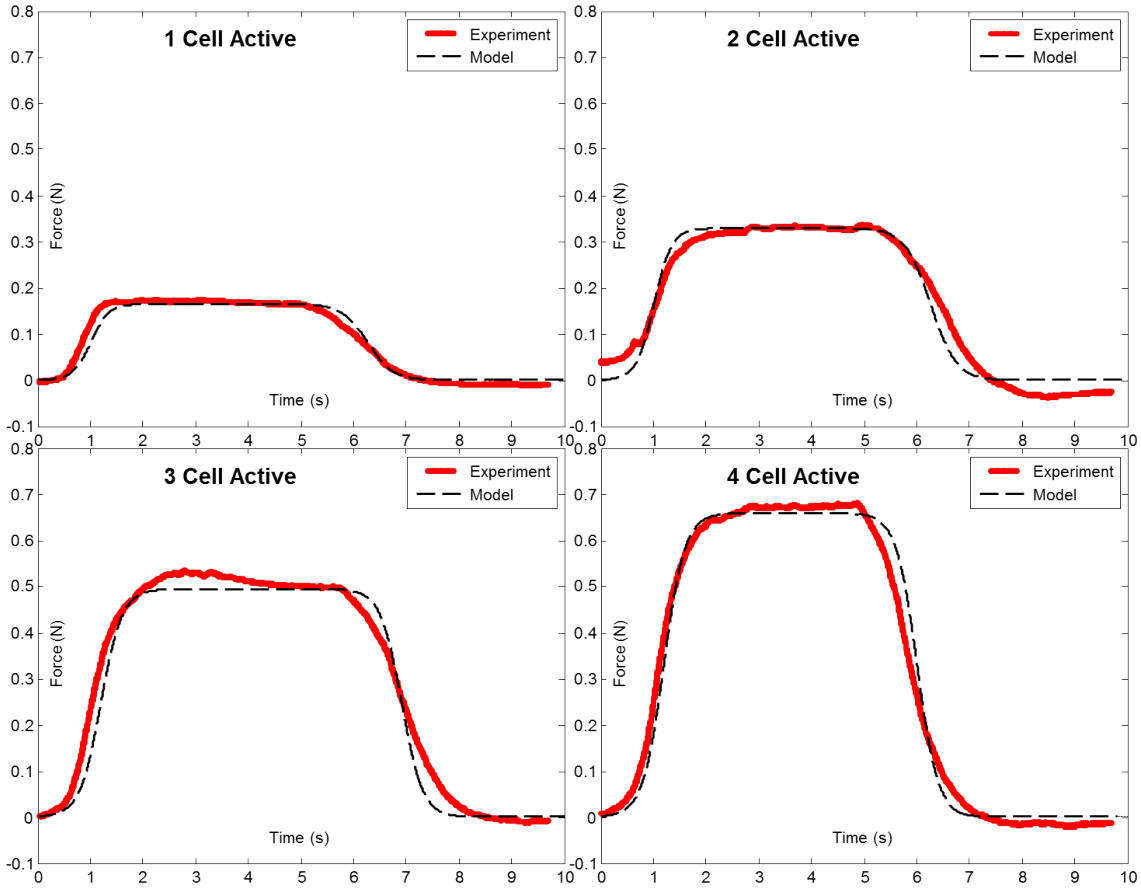


Figure 49: Comparison of 4 cell actuator array physical system and simulated results.

differed between connection points with two outgoing cells and those with only one outgoing cell. While the experimental setup could be upgraded to normalize the masses, usable actuator arrays will likely have non-uniform masses and the results show that this difference can often be small enough to neglect. It should be noted that no scaling of the physical system nor the simulated results was done and all cell properties were calculated prior to conducting any trials to remove any unintended bias. The results validate the dynamic aspects of the presented theory as the system was not damped. Additionally, since the control input was simply a step acting on a damper it can be seen that no input shaping was done to introduce a bias which further validates the theory.

6.3 Conclusion

This chapter addressed the challenge of experimentally validating the theory presented in this thesis. Three new actuator arrays ranging from mostly static to highly dynamic were introduced

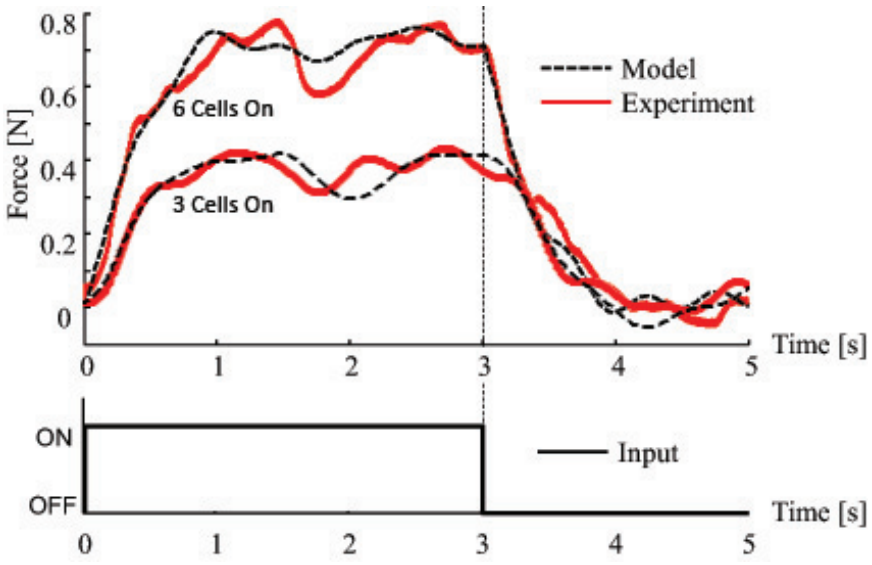


Figure 50: Comparison of 6 cell actuator array physical system and simulated results. All cells were activated for 3 seconds and then deactivated.

and used to test the presented methods. The chapter began by describing the static solenoid array which was used to validate the static properties theory and demonstrate a stochastic (probabilistic) signal distribution method, a new set of results. This was followed by describing the damped shape memory alloy (SMA) actuator array. This is the most practical of the physical arrays and follows a bio-inspired design based on biological muscle structure, a new approach introduced in this work. Several guidelines and suggestions were presented which could aid future designers building actuator arrays. The dynamic SMA actuator array was then presented which was used to validate the dynamic aspects of the presented theory. Finally, the new results from the two SMA actuator arrays were presented and shown to match numerical simulations.

This chapter specifically contributed three new physical actuator arrays and new results showing how the arrays compared to numerical simulations of the arrays based on the presented methods. This chapter also described how an actuator array can be built based on bio-inspired design principles.

CHAPTER VII

CONCLUSION

This thesis presented the representation and mathematical modeling of biologically-inspired robotic muscles called actuator arrays which are made from many small interconnected actuation units (cells). Combining these units together allows them to gain greater force, displacement, robustness, and other properties the original actuators could not achieve. The primary challenges included a) determining a representation for arrangements (topologies), b) exploring possible topologies, c) quantifying differences between array topologies, d) modeling the dynamics of actuator arrays and outside elements connected to them, e) finding engineering properties for actuator arrays to guide design, f) distributing control signals across redundant actuation, g) allowing for non-linear dynamics, and finally h) validating the presented theory experimentally.

Chapter 2 addressed the challenges of representing complex cellular actuator array topologies, exploring possible topologies given certain criteria, and determining the similarity between related topologies. The chapter began by first addressing the main components of an array, the cell and the mass, and restrictions on what types of actuators can be used in each cell of an array. The discussion then moved into representing arrays using a modified incidence matrix from graph theoretic modeling and introduced a reduced representation of well-ordered topologies called a fingerprint. The fingerprint was then used to deterministically identify possible array topologies so that they can be evaluated. The chapter closed by presenting a method for determining similarity between topologies which future work can use to search for topologies with desirable properties.

Chapter 3 addressed the challenge of generating the dynamic equations of motion for varied cellular actuator array topologies with a method that accommodates any valid linear internal cell dynamics. This is needed in order to control and simulate the response of actuator arrays when applied to useful systems. The chapter began by describing a Hill-Type model cell and detailing the equations of motion for that cell. This provided an example as a guide for the following sections which developed the dynamic equations of motion for any generalized actuator array with valid

linear cell dynamics. The process worked much like a plug-and-chug method starting with the incidence matrices from chapter 2 and ending with the dynamic equations in a standardized linear form. An example was then presented to clarify the process for a small Hill-Type model actuator array. Numerical results were presented following the example to show the validity of the presented method compared to MatLab's SimMechanics toolbox, a dynamic modeling suite. Finally, the proof and derivation of the presented method was given to show its validity theoretically.

Chapter 4 addressed the challenge of adding dynamic elements to actuator arrays which do not fit the normal actuator array model. These could be simple mass loads, multi-array robotic arms, or even hierarchical arrangements of actuator arrays. The chapter demonstrated how outside dynamics can be represented relative to an actuator array and then walked through the mathematics behind adapting the actuator array dynamic equations of motion into a complete system including the outside dynamics. An example was provided where the example array from the chapter 3 was attached to a simple mass load, and then numerical results were presented which showed a complex three-layer hierarchical camera positioner actuator array's force response under isometric loading. These numerical results were compared against a SimMechanics model to validate the presented theory.

Chapter 5 addressed the three challenges of developing static properties to judge engineering utility of actuator arrays, controlling the redundant actuators in an actuator array, and handling non-linear internal cell dynamics. The chapter began with the static modeling portion and provided several useful engineering properties to keep in mind when designing actuator arrays. Since most applications of actuator arrays deal with force control, the force function was given particular attention. Actuator array expected force levels were determined by using a modified version of the dynamic modeling equations. The issue of controlling redundant actuators was then addressed. Two signal distribution methods were presented, one based on probability and the other on deterministic grouping and recruitment of cells. Both methods were biologically inspired, one from the stochastic process of calcium diffusion in muscle and the other on motor unit recruitment. Next a numeric comparison of the static and stochastic properties of several example actuator arrays were presented and discussed to show how the properties can guide design. Finally a non-linear extension to the dynamic modeling method presented previously was given which allows structured non-linearities

to be handled in the internal cell dynamics.

Chapter 6 addressed the challenge of experimentally validating the theory presented in this thesis. Three new actuator arrays ranging from mostly static to highly dynamic were introduced and used to test the presented methods. The chapter began by describing the static solenoid array which was used to validate the static properties theory and demonstrate a stochastic (probabilistic) signal distribution method, a new set of results. This was followed by describing the damped shape memory alloy (SMA) actuator array. This is the most practical of the physical arrays and follows a bio-inspired design based on biological muscle structure, a new approach introduced in this work. Several guidelines and suggestions were presented which could aid future designers building actuator arrays. The dynamic SMA actuator array was then presented which was used to validate the dynamic aspects of the presented theory. Finally, the new results from the two SMA actuator arrays were presented and shown to match numerical simulations.

The approach presented in this thesis was designed to aid automation and simulation of cellular actuator arrays, allow for fast recalculation of different cell array topologies, and provide an intuitive base for future controls and physiology work. While the dynamics representing a given cell array actuator could be generated using other means, such as Dymola, SimScape, SimMechanics, or others, the presented method allows the dynamics to be calculated with less human effort, less computational effort, and with greater speed. The method greatly outperforms other methods when needing to test many different actuator arrays as part of a design and engineering method as it allows for efficient recalculation given different array topologies.

7.1 Contributions

This research supports the goals of creating natural motion, overcoming the limits of traditional actuation, and creating human-safe actuators by introducing cellular actuator arrays with redundant and biologically inspired actuation and the mathematical modeling methods needed to represent and design them. The contributions are:

- Chapter 2 specifically contributed two incidence matrix representation for topologies which enables the dynamic equations of motion to be more easily generated than with traditional incidence matrices from graph theoretic modeling. A new compact mathematical representation

of topologies called the 'fingerprint' was also created along with a method to automatically generate array topologies based on the 'fingerprint.' Finally, a mathematical process was then introduced which allows direct generation of the new incidence matrices directly from the 'fingerprint' and vice versa.

- Chapter 3 specifically contributed a mathematical formulation which allows the new incidence matrices from chapter 2 to be combined with the idea of 'stamping' to directly generate the dynamic equations of motion for an actuator array. The process was based on a new analysis of the mathematical structures of actuator arrays which identified the key to simplifying force sums using the linear incidence matrices. The results presented were then verified by numerical simulations which had an error of less than 1%.

- Chapter 4 specifically contributed a mathematical treatment of the separation of internal array dynamics from outside environment dynamics and a way to mathematically describe the interaction between the two. It also defined a method to use the outside dynamics mathematical treatment to represent complex hierarchical systems. Both the outside dynamics and hierarchical systems modeling were verified using a numerical model of a camera positioner which had less than a 3% error.

- Chapter 5 specifically contributed a definition of key static properties for use in actuator array design and developed a method to calculate these properties from the dynamic equations developed in 3. This chapter also showed that the variability in actuator array force when controlled stochastically is topologically dependent, a new result. Finally, this chapter developed a new method that can accommodate well-structured non-linear elements in actuator array cells by separating these elements from the outputs of the cell.

- Chapter 6 specifically contributed three new physical actuator arrays and new results showing how the arrays compared to numerical simulations of the arrays based on the presented methods. This chapter also described how an actuator array can be built based on bio-inspired design principles.

7.2 Future Work

7.2.1 Relate Incidence Matrix to Actuator Array Properties

The methods provided in this thesis can deterministically find possible actuator array topologies and then separately find their properties. This provides the necessary criteria for designing an actuator array, but the only way to ensure an optimal solution has be located would be to test every possible topology. That is a computational expensive challenge. There is currently no direct relationship between changes to the incidence matrix (or fingerprint) and changes to the properties of an array. Having such a relationship would allow efficient search algorithms to be developed which could intelligently explore the possible topologies and determine an optimal solution to a design challenge with less computational effort and time.

7.2.2 Expand Coverage of Dynamic Theory

This thesis allows for a wide range of internal cell dynamics but does not allow for much flexibility in the connecting structure of an actuator array. The theory best represents actuator arrays where the connections between cells have significant mass. As this mass goes to zero, the numbers involved in the calculations of acceleration of the masses become large and calculation errors can be introduced. A useful expansion of the theory would allow for a combination of internal cell dynamics for cells joining to a mass-less connection point such that the acceleration of the point does not need to be calculated.

The theory in this thesis covers limited non-linear applications, but this is due mostly to a notation complexity and limit of time. The same theory could be extended to cover many more non-linear internal cell dynamics cases so long as an easy way to determine cell force exists.

The dynamic theory presented in this thesis is applicable to problems beyond actuator arrays. Many systems in the world contain one dimensional movement with many redundant actors which connect to one another in various topologies. One of the strengths of this theory is its adaptability to changing network structure, so it could be applied to problems like traffic modeling where a local actor is affected only by those nearby but the overall behavior of the system needs to be determined. The network topology of a road is constantly changing, but a model like the one presented in this thesis would be able to efficiently recalculate the system dynamics without large computational load

or human input.

Finally, this thesis separates the dynamics of cells based on the forces they exert on masses distributed throughout the actuator array. Other systems may have different points of separability which can still be exploited by the presented method. Graph theoretic modeling refers to these separation points as terminal equations, and these equations have already be developed for applications ranging from electrical system to hydraulics. Future work could adapt the theory in this thesis to cover those systems as well.

7.2.3 Multi-Degree of Freedom Systems

A common misconception about biological muscle is that it has only a single degree of freedom. A single muscle can have multiple heads connecting to different locations on a bone or even different bones. Each of these heads can pull in a different direction, so treating them all as a single degree of freedom can oversimplify the system. The presented method only deals with a single degree of freedom, but the graph theoretic modeling principles it is based on extend into multi-dimensional spaces. A highly useful extension of the current theory would be to allow multiple degrees of freedom for an array.

This could also take the form of representing motion of an actuator array perpendicular to its direction of travel which might affect the tension force along the array. This is similar to a mass being suspended between two springs. If the mass moves perpendicularly to the springs, it still causes displacement in the springs and thus force. The current theory does not cover this case.

7.2.4 Understanding Natural Motion

Cellular actuator arrays and the mathematical models presented in this thesis provide a well understood physical platform for further research understanding muscle optimality criteria and the control signals used to control human muscles. The signal distribution methods presented here demonstrate the same characteristics as the signal-dependent noise observed in biological muscle, and developing optimal control methods for the cellular actuator arrays will likely lead to greater understanding of the brain's control over biological muscle. Furthermore, since optimality criteria like signal dependent noise is believed to lead to natural motion, actuator arrays which follow this optimality criteria should be capable of achieving natural motion. This enables the creation of

naturally moving and safe robotic systems for use in therapy and rehabilitation, prosthetic devices, human force amplification exoskeletons, and humanoid robotic systems. This can benefit a range of people from those looking for help around the house to those recovering from stroke to those trying to maintain independence and quality-of-life while dealing with conditions like ALS.

REFERENCES

[1] ALEXANDER, M., NELSON, M., and SHAH, A., "Orthotics, adapted seating and assistive devices," *Pediatric Rehabilitation*, pp. 186–187, 1992.

[2] ALON, U., *An introduction to systems biology: design principles of biological circuits*. Chapman & Hall/CRC, 2007.

[3] BONDY, J. and MURTY, U., *Graph Theory with Applications*. London: Macmillan London, 290 ed., 1976.

[4] BORASIO, G. D., VOLTZ, R., and MILLER, R. G., "Palliative care in amyotrophic lateral sclerosis.," *Journal of Neurology*, vol. 19, pp. 829–47, Nov. 2001.

[5] BUCHANAN, T. and SHREEVE, D., "An evaluation of optimization techniques for the prediction of muscle activation patterns during isometric tasks," *Journal of Biomechanical Engineering*, vol. 118, p. 565, 1996.

[6] CALDWELL, D. and TSAGARAKIS, N., "Biomimetic actuators in prosthetic and rehabilitation applications," *Technology and Health Care*, vol. 10, no. 2, pp. 107–120, 2002.

[7] CANFIELD, S. and FRECKER, M., "Topology optimization of compliant mechanical amplifiers for piezoelectric actuators," *Structural and Multidisciplinary Optimization*, vol. 20, no. 4, pp. 269–279, 2000.

[8] CHANDRASHEKAR, M., ROE, P. H., and SAVAGE, G. J., "A Unified Approach to Modelling Photovoltaic Power Systems," *Modeling and Simulation*, vol. 23, no. 4, pp. 313–313, 1993.

[9] CHANDRASHEKAR, M., SAVAGE, G., BIRKETT, S., and MCPHEE, J., "Graph-Theoretic Modeling: Four Decades of Development," *Technical Report: University of Waterloo; Systems Design Engineering*, 1995.

[10] CHURCHLAND, M. M., AFSHAR, A., and SHENOY, K. V., "A central source of movement variability.," *Neuron*, vol. 52, pp. 1085–96, Dec. 2006.

[11] CONWAY, N. J., TRAINA, Z. J., and KIM, S.-G., "A strain amplifying piezoelectric MEMS actuator," *Journal of Micromechanics and Microengineering*, vol. 17, pp. 781–787, Apr. 2007.

[12] CRAIG, J. J., *Introduction to robotics: mechanics and control*. Addison-Wesley series in electrical and computer engineering: control engineering, Pearson/Prentice Hall, 2005.

[13] CROWNINSHIELD, R. D. and BRAND, R. A., "A physiologically based criterion of muscle force prediction in locomotion.," *Journal of Biomechanics*, vol. 14, pp. 793–801, Jan. 1981.

[14] DAVIES, P. M., *Steps to follow: the comprehensive treatment of patients with hemiplegia*. Springer, 2000.

[15] DIAZ-CALDERON, A., PAREDIS, C. J. J., and KHOSLA, P. K., "Automatic generation of system-level dynamic equations for mechatronic systems," *Computer-Aided Design*, vol. 32, pp. 339–354, May 2000.

[16] DICKINSON, M. H., CLAIRE, F. T., FULL, R. J., KOEHL, M. A. R., KRAM, R., and LEHMAN, S., "How Animals Move: An Integrative View," *Science*, vol. 288, pp. 100–106, Apr. 2000.

[17] DISALVO, C. F., GEMPERLE, F., FORLIZZI, J., and KIESLER, S., "All robots are not created equal: the design and perception of humanoid robot heads," in *Proceedings of the 4th conference on Designing interactive systems*, pp. 321–326, 2002.

[18] DOGAN, A., UCHINO, K., and NEWNHAM, R. E., "Composite piezoelectric transducer with truncated conical endcaps cymbal," *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 44, no. 3, pp. 597–605, 1997.

[19] DOGAN, A., XU, Q., ONITSUKA, K., YOSHIKAWA, S., UCHINO, K., and NEWNHAM, R. E., "High Displacement Ceramic Metal Composite Actuators (Moonies)," *Ferroelectrics*, vol. 156, no. 1, pp. 1–6, 1994.

[20] DONNAN, G. A., FISHER, M., MACLEOD, M., and DAVIS, S. M., "Stroke," *Lancet*, vol. 371, pp. 1612–1623, 2008.

[21] DOOB, J. L., *Stochastic Processes (Reprint Edition)*. Wiley-Interscience, 1990.

[22] DRORY, V. E., GOLTSMAN, E., REZNIK, J. G., MOSEK, A., and KORCZYN, A. D., "The value of muscle exercise in patients with amyotrophic lateral sclerosis.," *Journal of the Neurological Sciences*, vol. 191, pp. 133–7, Oct. 2001.

[23] ENOKA, R. M., *Neuromechanics of Human Movement*. Human Kinetics Publishers, 2008.

[24] FU, Y., DU, H., HUANG, W., ZHANG, S., and HU, M., "TiNi-based thin films in MEMS applications: a review," *Sensors and Actuators A: Physical*, vol. 112, pp. 395–408, May 2004.

[25] GAO, W. and WANG, H., "An Automatic Dynamics Generation Method for Reconfigurable Modular Robot," *Advances in Reconfigurable Mechanisms and Robots I*, vol. 5, pp. 551–560, 2012.

[26] GEIJTENBEEK, T., BOGERT, A. J. V. D., BASTEN, B. J. H. V., and EGGES, A., "Evaluating the Physical Realism of Character Animations Using Musculoskeletal Models," *Lecture Notes in Computer Science*, vol. 6459, pp. 11–22, 2010.

[27] HAERTLING, G. H., "Ferroelectric Ceramics: History and Technology," *Journal of the American Ceramic Society*, vol. 82, pp. 797–818, Apr. 1999.

[28] HAMILTON, A. F. D. C., JONES, K. E., and WOLPERT, D. M., "The scaling of motor noise with muscle strength and motor unit number in humans.," *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale*, vol. 157, pp. 417–30, Aug. 2004.

[29] HANRAHAN, K. and KHAN, A., "Adaptive Micro-Actuated Head Gimbal Assembly," 1998.

[30] HARA, S., ZAMA, T., TAKASHIMA, W., and KANETO, K., "Artificial muscles based on polypyrrole actuators with large strain and stress induced electrically," *Polymer journal*, vol. 36, no. 2, pp. 151–161, 2004.

[31] HARRIS, C. M. and WOLPERT, D. M., "Signal-dependent noise determines motor planning," *Nature*, vol. 394, no. August, pp. 780–784, 1998.

[32] HIRIS, E., "Detection of Biological and Non-Biological Motion," *Journal of Vision*, vol. 7, no. 12, pp. 1–16, 2007.

[33] HOSPOD, V., AIMONETTI, J.-M., ROLL, J.-P., and RIBOT-CISCAR, E., "Changes in human muscle spindle sensitivity during a proprioceptive attention task.," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 27, pp. 5172–8, May 2007.

[34] JACOB, S. W. and FRANCONE, C. A., *Structure and function in man*. Saunders (Philadelphia), 1982.

[35] JÄNKER, P., CHRISTMANN, M., HERMLE, F., LORKOWSKI, T., and STORM, S., "Mechatronics using piezoelectric actuators," *Journal of the European Ceramic Society*, vol. 19, pp. 1127–1131, 1999.

[36] JOHANSSON, G., "Visual perception of biological motion and a model for its analysis," *Perception & Psychophysics*, vol. 14, no. 2, pp. 201–211, 1973.

[37] JONES, K. E., HAMILTON, A. F. D. C., WOLPERT, D. M., DIMITRIOU, M., and FRANKLIN, D. W., "Sources of signal-dependent noise during isometric force production," *Journal of Neurophysiology*, vol. 88, no. 3, pp. 1533–1544, 2002.

[38] JULIUS, A. and SAKAR, M., "Stochastic modeling and control of biological systems: the lactose regulation system of Escherichia coli," *IEEE Transactions on Automatic Control*, vol. 53, pp. 51–65, 2008.

[39] KIERNAN, M. C., VUCIC, S., CHEAH, B. C., TURNER, M. R., EISEN, A., HARDIMAN, O., BURRELL, J. R., and ZOING, M. C., "Amyotrophic lateral sclerosis.," *Lancet*, vol. 377, pp. 942–55, Mar. 2011.

[40] KITAMURA, K. and YANAGIDA, T., "Stochastic properties of actomyosin motor," *Biosystems*, vol. 71, pp. 101–110, Sept. 2003.

[41] KOENIG, H., TOKAD, Y., and KESAVAN, H., *Analysis of Discrete Physical Systems*. New Yort: McGraw-Hill, 1967.

[42] KOSTYUKOV, A. I., "Muscle hysteresis and movement control: a theoretical study," *Neuroscience*, vol. 83, no. 1, pp. 303–320, 1998.

[43] KREBS, H. I., CELESTINO, J., WILLIAMS, D., FERRARO, M., VOLPE, B., and HOGAN, N., "A Wrist Extension for MIT-MANUS," *Advances in Rehabilitation Robotics*, vol. 306, pp. 377–390, 2004.

[44] KRISHNAMOORTHY, V., GOODMAN, S., ZATSIORSKY, V., and LATASH, M. L., "Muscle synergies during shifts of the center of pressure by standing persons: identification of muscle modes," *Biological cybernetics*, vol. 89, pp. 152–161, Aug. 2003.

[45] KUSHNER, H. J., "On the stability of stochastic dynamical systems," *National Academy of Sciences*, vol. 53, no. 1, pp. 8–13, 1964.

[46] KUTNER, N. G., ZHANG, R., BUTLER, A. J., WOLF, S. L., and ALBERTS, J. L., "Quality-of-life change associated with robotic-assisted therapy to improve hand motor function in patients with subacute stroke: a randomized clinical trial.," *Physical therapy*, vol. 90, pp. 493–504, Apr. 2010.

[47] KWAKKEL, G., KOLLEN, B. J., and KREBS, H. I., "Effects of robot-assisted therapy on upper limb recovery after stroke: a systematic review.," *Neurorehabilitation and neural repair*, vol. 22, no. 2, pp. 111–21, 2008.

[48] LEE, S. and SANKAI, Y., "Power assist control for walking aid with HAL-3 based on EMG and impedance adjustment around knee joint," in *International Converence on Intelligent Robots and Systems*, no. October, pp. 1499–1504, 2002.

[49] MACNAIR, D. and UEDA, J., "A fingerprint method for variability and robustness analysis of stochastically controlled cellular actuator arrays," *The International Journal of Robotics Research*, vol. 30, no. 5, pp. 536–555, 2011.

[50] MARTINI, F. H. and BARTHOLOMEW, E. F., *Essentials of Anatomy & Physiology*. 4 ed., 2006.

[51] MCPHEE, J., "Dynamics of Multibody Systems: Conventional and Graph-Theoretic Approaches," *Technical Report: University of Waterloo*, 2004.

[52] MCPHEE, J. J., "On the use of linear graph theory in multibody system dynamics," *Nonlinear Dynamics*, vol. 9, pp. 73–90, Feb. 1996.

[53] MITCHELL, J. D. and BORASIO, G. D., "Amyotrophic lateral sclerosis.," *Lancet*, vol. 369, pp. 2031–41, June 2007.

[54] NAGURNEY, A. and QIANG, Q., "Robustness of transportation networks subject to degradable links," *Europhysics Letters (EPL)*, vol. 80, pp. 68001–p1 – p6, Dec. 2007.

[55] NEWNHAM, R. E., DOGAN, A., XU, Q. C., ONITSUKA, K., TRESSLER, J., and YOSHIKAWA, S., "Flextensional "moonie" actuators," in *Ultrasonics Symposium*, pp. 509–513, IEEE, 1993.

[56] NIEZRECKI, C., BREI, D., BALAKRISHNAN, S., and MOSKALIK, A., "Piezoelectric actuation: State of the art," *The Shock and Vibration digest*, vol. 33, no. 4, pp. 269–280, 2001.

[57] NOZAKI, D., NAKAZAWA, K., and AKAI, M., "Muscle activity determined by cosine tuning with a nontrivial preferred direction during isometric force exertion by lower limb.," *Journal of neurophysiology*, vol. 93, pp. 2614–24, May 2005.

[58] ONITSUKA, K., DOGAN, A., TRESSLER, J. F., XU, Q., YOSHIKAWA, S., and NEWNHAM, R. E., "Metal-ceramic composite transducer, the "moonie"," *Journal of Intelligent Material Systems and Structures*, vol. 6, no. 4, pp. 447–455, 1995.

[59] OSU, R., KAMIMURA, N., IWASAKI, H., NAKANO, E., HARRIS, C. M., WADA, Y., and KAWATO, M., "Optimal impedance control for task achievement in the presence of signal-dependent noise.," *Journal of neurophysiology*, vol. 92, pp. 1199–215, Aug. 2004.

[60] OTTENHEIJM, C. A. C., HEUNKS, L. M. A., and DEKHUIJZEN, R. P. N., "Diaphragm adaptations in patients with COPD.," *Respiratory Research*, vol. 9, Jan. 2008.

[61] OZTOP, E., FRANKLIN, D. W., CHAMINADE, T., and CHENG, G., "Human-humanoid interaction: is a humanoid robot perceived as a human?," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 537–559, 2005.

[62] PERRY, J. and ROSEN, J., "Design of a 7 Degree-of-Freedom Upper-Limb Powered Exoskeleton," in *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.*, pp. 805–810, Ieee, 2006.

[63] PLANTE, J.-S. and DUBOWSKY, S., "On the Nature of Dielectric Elastomer Actuators and Its Implications for Their Design," in *Proceedings of SPIE*, pp. 61681J 1–11, 2006.

[64] PRATT, G. and WILLIAMSON, M., "Series Elastic Actuators," *Intelligent Robots and Systems*, vol. 1, pp. 399–406, 1995.

[65] PRILUTSKY, B. I., "Coordination of two- and one-joint muscles: functional consequences and implications for motor control," *Motor Control*, vol. 4, no. 1, pp. 1–44, 2000.

[66] REISMAN, D. S. and SCHOLZ, J. P., "Aspects of joint coordination are preserved during pointing in persons with post-stroke hemiparesis.," *Brain*, vol. 126, pp. 2510–27, Nov. 2003.

[67] ROWLAND, L. P. and SCNEIDER, N. A., "Amyotrophic Lateral Sclerosis," *The New England Journal of Medicine*, vol. 344, no. 22, pp. 1688–1700, 2001.

[68] SANCHEZ, A., MAHOUT, V., and TONDU, B., "Nonlinear parametric identification of a McKibben artificial pneumatic muscle using flatness property of the system," in *IEEE International Conference on Control Applications*, no. September, pp. 70–74, 1998.

[69] SASS, L., MCPHEE, J. M. C., SCHMITKE, C., FISETTE, P., and GRENIER, D., "A Comparison of Different Methods for Modelling Electromechanical Multibody Systems," *Multibody System Dynamics*, vol. 12, no. 3, pp. 209–250, 2004.

[70] SCHERRER, M. and MCPHEE, J., "Dynamic Modelling of Electromechanical Multibody Systems," *Multibody System Dynamics*, vol. 9, no. 1, pp. 87–115, 2003.

[71] SCHIEHLEN, W., *Multibody Systems Handbook*. New York: Springer-Verlag.

[72] SELDEN, B., CHO, K., and ASADA, H. H., "Segmented shape memory alloy actuators using hysteresis loop control," *Smart Materials and Structures*, vol. 15, pp. 642–652, Apr. 2006.

[73] SHAHINPOOR, M., KIM, K. J., and SCHREYER, H. B., "Artificial Sarcomere and Muscle Made with Conductive Polyacrylonitrile (C-PAN) Fiber Bundles," in *Proceedings of SPIE*, vol. 3987, p. 243, 2000.

[74] SHI, P. and MCPHEE, J., "Symbolic Programming of a Graph-Theoretic Approach to Flexible Multibody Dynamics," *Mechanics of Structures and Machines*, vol. 30, no. 1, pp. 123–154, 2002.

[75] SHOOMAN, M. L., *Reliability of Computer Systems and Networks*, vol. 3. 2002.

[76] SIMMONS, G. and DEMIRIS, Y., "Optimal robot arm control using the minimum variance model," *Journal of Robotic Systems*, vol. 22, pp. 677–690, Nov. 2005.

[77] SIMS, N. R. and MUYDERMAN, H., "Mitochondria, oxidative metabolism and cell death in stroke.," *Biochimica et biophysica acta*, vol. 1802, pp. 80–91, Jan. 2010.

[78] SINGER, Y., "Dynamic Measure of Network Robustness," in *Convention of Electrical & Electronics Engineers in Israel*, no. 1, pp. 366–370, Ieee, Nov. 2006.

[79] SRINIVASAN, A. V. and MCFARLAND, D. M., *Smart Structures: Analysis and Design*. Cambridge University Press, 2000.

[80] STEIN, R. B., GOSSEN, E. R., and JONES, K. E., "Neuronal variability: noise or part of the signal?," *Nature reviews. Neuroscience*, vol. 6, pp. 389–97, May 2005.

[81] STELLING, J., KLAMT, S., BETTENBROCK, K., SCHUSTER, S., and GILLES, E. D., "Metabolic network structure determines key aspects of functionality and regulation.," *Nature*, vol. 420, pp. 190–3, Dec. 2002.

[82] STERN, M. D., PIZARRO, G., and RÍOS, E., "Local Control Model of Excitation-Contraction Coupling in Skeletal Muscle," *The Journal of General Physiology*, vol. 110, pp. 415–440, Jan. 1997.

[83] TIMMERMANS, A. A. A., SPOOREN, A. I. F., KINGMA, H., and SEELEN, H. A. M., "Influence of task-oriented training content on skilled arm-hand performance in stroke: a systematic review.," *Neurorehabilitation and neural repair*, vol. 24, no. 9, pp. 858–70, 2010.

[84] TODOROV, E., "Cosine tuning minimizes motor errors," *Neural Computation*, vol. 14, no. 6, pp. 1233–60, 2002.

[85] TODOROV, E., "Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system.," *Neural computation*, vol. 17, pp. 1084–108, May 2005.

[86] TÓTH, A., ARZ, G., FAZEKAS, G., BRATANOV, D., and ZLATOV, N., "Post Stroke Shoulder Elbow Physiotherapy," *Advances in Rehabilitation Robotics*, vol. 306, pp. 391–411, 2004.

[87] TROJE, N. F., "Decomposing biological motion: a framework for analysis and synthesis of human gait patterns.," *Journal of Vision*, vol. 2, pp. 371–87, Jan. 2002.

[88] TSAGARAKIS, N. G. and CALDWELL, D. G., "Development and Control of a Soft-Actuated Exoskeleton for Use in Physiotherapy and Training," *Autonomous Robots Special Issue on Rehabilitation Robotics*, vol. 15, no. 1, pp. 21–33, 2003.

[89] TYREMAN, M. J. A. and MOLLOY, J. E., "Molecular Motors: Natures Nanomachines," *IEE Proceedings – Nanobiotechnology*, vol. 150, no. 3, pp. 95–102, 2003.

[90] UCHINO, K., *Piezoelectric Actuators and Ultrasonic Motors*. Massachusetts: Kluwer Academic, 1997.

[91] UEDA, J., DING, M., MATSUGASHITA, M., OYA, R., and OGASAWARA, T., "Pinpointed control of muscles by using power-assisting device," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Rome, Italy), pp. 3621–3626, Apr. 2007.

[92] UEDA, J., MACNAIR, D., and BROWN, E., "Quantized Control of Compliant Cellular Actuator Arrays for Biological Movement Generation," in *ASME Dynamic Systems and Controls Conference*, 2012.

[93] UEDA, J., MATSUGASHITA, M., OYA, R., and OGASAWARA, T., "Control of Muscle Force During Exercise Using a Musculoskeletal-Exoskeletal Integrated Human Model," in *Experimental Robotics* (KHATIB, O., KUMAR, V., and RUS, D., eds.), vol. 39 of *Springer Tracts in Advanced Robotics*, pp. 143–152, Springer Berlin / Heidelberg, 2008.

[94] UEDA, J., ODHNER, L., and ASADA, H. H., "Broadcast Feedback of Stochastic Cellular Actuators Inspired by Biological Muscle Control," *The International Journal of Robotics Research*, vol. 26, pp. 1251–1265, Nov. 2007.

[95] UEDA, J., ODHNER, L., KIM, S.-G., and ASADA, H. H., "Distributed Stochastic Control of MEMS-PZT Cellular Actuators with Broadcast Feedback," in *International Conference on Biomedical Robotics and Biomechatronics*, pp. 272–277, Ieee, 2006.

[96] UEDA, J., SECORD, T., and ASADA, H. H., "Design of PZT Cellular Actuators with Power-Law Strain Amplification," in *International Conference on Intelligent Robots and Systems*, pp. 1160–1165, Ieee, Oct. 2007.

[97] UEDA, J., SECORD, T., and ASADA, H. H., "Piezoelectric Cellular Actuators Using Nested Rhombus Multilayer Mechanisms," in *Dynamic Systems and Control Conference*, pp. 203–210, 2008.

[98] VAN ZANDWIJK, J. P., BOBBERT, M. F., HARLAAR, J., and HOF, A. L., "From twitch to tetanus for human muscle: experimental data and model predictions for m. triceps surae.," *Biological cybernetics*, vol. 79, pp. 121–130, Aug. 1998.

[99] VAN ZANDWIJK, J. P., BOBBERT, M. F., BAAN, G. C., and HUIJING, P. A., "From twitch to tetanus: performance of excitation dynamics optimized for a twitch in predicting tetanic muscle forces.," *Biological cybernetics*, vol. 75, pp. 409–417, Nov. 1996.

[100] VENEMAN, J. F., KRUIDHOF, R., HEKMAN, E. E. G., EKKELENKAMP, R., VAN ASSELDONK, E. H. F., and VAN DER KOOIJ, H., "Design and Evaluation of the LOPES Exoskeleton Robot for Interactive Gait Rehabilitation.," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 15, pp. 379–86, Oct. 2007.

[101] WONG, C. K., BISHOP, L., and STEIN, J., "A wearable robotic knee orthosis for gait training: a case-series of hemiparetic stroke survivors.," *Prosthetics and orthotics international*, vol. 36, pp. 113–20, Mar. 2012.

[102] YAMAGUCHI, G. T., *Dynamic Modeling of Musculoskeletal Motion: A Vectorized Approach for Biomechanical Analysis in Three Dimensions*. 2005.

[103] YIM, M., SHEN, W.-M., SALEMI, B., RUS, D., MOLL, M., LIPSON, H., KLAVINS, E., and CHIRIKJIAN, G., "Modular self-reconfigurable robot systems [Grand Challenges of Robotics]," *Robotics and Automation Magazine, IEEE*, vol. 14, no. March, p. 9, 2007.

[104] ZINN, M., ROTH, B., KHATIB, O., and SALISBURY, J. K., "A New Actuation Approach for Human Friendly Robot Design," *The International Journal of Robotics Research*, vol. 23, pp. 379–398, Apr. 2004.