Evaluation of Decentralized Reactive Swing-Leg Controllers on Powered Robotic Legs

Alexander Schepelmann

CMU-RI-TR-16-02

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Thesis Committee:
Steven H. Collins
Hartmut Geyer (Chair)
Ralph L. Hollis
Herman van der Kooij, Universiteit Twente

Personal support was provided by the National Science Foundation Graduate Research Fellowship Program (NSF GRFP) (Grant 0946825). Work presented in this thesis was supported by the Eunice Kennedy Shriver National Institute of Child Health & Human Development (Award 1R01HD075492) and the Richard King Mellon Foundation.

Keywords: Locomotion, Neuromuscular Control, Series Elastic Actuators, Nonlinear Springs

For Jackson, my best humanoid to date.

Abstract

We present work to transfer decentralized neuromuscular control strategies of human locomotion to powered segmented robotic legs. State-of-the-art robotic locomotion control approaches, like centralized planning and tracking in fully robotic systems and predefined motion pattern replay in prosthetic systems, do not enable the dynamism and reactiveness of able-bodied humans. Animals largely realize dexterous segmented leg performance with leg-encoded biomechanics and local feedback controls that bypass central processing. A decentralized neuromuscular controller was recently developed that enables robust locomotion for a simulated multi-segmented planar humanoid. A portion of this controller was used in an active ankle-foot prosthesis to modulate ankle torque during stance, enabling level and inclined ground walking. While these results suggest that the neuromuscular controller is a promising alternative control method for both fully robotic systems and powered prostheses, it is unclear if the controller can be transferred to multi-segmented robotic legs. The goal of this thesis is to investigate the feasibility of controlling a multi-segmented robotic leg with the proposed neuromuscular control approach, which may enable robots and powered prostheses to react to locomotion disturbances dynamically and in a human-like way. Specifically, work in this thesis investigates two hypotheses. Hypothesis one posits that the proposed decentralized swing-leg controllers enable more robust foot placements into ground targets than state-of-the-art impedance controls. Hypothesis two posits that neuromuscular swing-leg control enables more human-like motion than state-of-the-art impedance control.

To transfer neuromuscular controls to powered segmented robotic legs, we use a model-based design approach. The initial transfer is focused on neuromuscular swing-leg controls, important for maintaining dynamic stability of both fully robotic systems and powered prostheses in the presence of unexpected locomotion disturbances, such as trips and pushes. We first present the design of RNL, a three segment, cable-driven, antagonistically actuated robotic leg with joint compliance. The robot's size, weight, and actuation capabilities correspond to dynamically scaled human values. Next, a high-fidelity simulation of the robot is created to investigate the feasibility of transferring neuromuscular controls, pre-tune hardware gains via optimization, and serve as a benchmark for hardware experiments. An idealized version of the swing-leg controller with mono-articular actuation, as well as the neuromuscular interpretation of this controller with multi-articular actuation is then transferred to RNL and evaluated with foot placement experiments. The results suggest that the proposed swing-leg controllers can accurately regulate foot placement of robotic legs during undisturbed and disturbed motions. Compared to impedance control, the proposed controls achieve foot placements over a range of ground targets with a single set of gains, which make them attractive candidates for regulating the motion of legged robots and prostheses in the real-world. Furthermore, the ankle trajectory traced out by the robot under neuromuscular control is more human-like than the trajectories traced out under the proposed idealized control and impedance control.

In parallel to this control transfer, a synthesis method for creating compact nonlinear springs with user-defined torque-deflection profiles is presented to explore methods for improving RNL's series elastic actuators. The springs use rubber as their elastic element, which, while enabling a compact spring design, introduce viscoelastic behavior in the spring that needs to be accounted for with additional control. To accurately estimate force developed in the rubber, an empirically characterized constitutive rubber model is developed and integrated into the series elastic actuator controller used by the RNL test platforms. Benchtop experiments show that in conjunction with an observer, the nonlinear spring prototype achieves desired behavior at actuation frequencies up to 2 Hz, after which spring behavior degrades due to rubber hysteresis. These results show that while the presented methodology is capable of realizing compact nonlinear springs, careful rubber selection that mitigates viscoelastic behavior is necessary during the spring design process.

Acknowledgments

First and foremost, thank you to my advisor Hartmut Geyer for overseeing this work, and for his guidance and support throughout my Ph.D. The lessons I learned from you have made me a better researcher and will continue to guide me throughout my career. Thank you also to my committee members Steve Collins, Ralph Hollis, and Herman van der Kooij.

Thank you to all of the master's students who worked with me on the RNL and NLS projects presented in this thesis. These projects were truly a collaborative effort, and this thesis would not exist without you. In order of appearance: Michael Taylor, Kathryn Geberth, Erle Misra, Jessica Austin, and Yin Zhong.

Additional thanks to my lab mates and collaborators who contributed to these projects and offered their time, help, and advice throughout: Justin Haines, Prasana Velagapuddi, David Matten, Ruta Desai, William Martin, Nitish Thatte, Ting Xu, Rajshekar Prabhakar, Anirudh Vasudevan, Aadith Varadarajan, Carmel Majidi, and Christopher Atkeson.

CMU's machinists, Lawrence Hayhurst and James Dillinger, were critical in manufacturing much of the hardware used in this thesis. Thank you for providing feedback on various iterations of the series elastic actuators, robot leg components, and nonlinear spring prototypes, as well as for teaching me about hardware fabrication and manufacturing.

Thank you to Marliese Bonk, our lab group's administrative coordinator, who made sure that we always had the necessary materials on-hand to continue work on the project, and who was always available to chat.

Above all, thank you to my wife Grace, my parents Carla and Heinz, and Grace's parents Young and Ken. Without their unwavering support and encouragement this thesis would not be possible. Finally thank you to my dogs Etzel, Xena, and Ripley, who would let me know that it was time to take a break by making me take them for a walk.

Contents

List of Figures

List of Tables

List of Symbols

Chapter 2

Symbol		Description
$\mathbf{M}$	...	Inertia matrix
$\mathbf{C}$	...	Centrifugal and Coriolis force vector
$\mathbf{g}$	...	Gravitational force vector
τ	...	Joint torque vector
$\mathbf{q}$	...	Generalized system coordinates
$\dot{\mathbf{q}}$	...	Generalized system velocities
$\ddot{\mathbf{q}}$	...	Generalized system accelerations
p_x	...	ZMP location
x	...	COM position
$\dot{x}$	...	COM velocity
$\ddot{x}$	...	COM acceleration
z_h	...	Constrained constant COM height relative to ground
g	...	Gravitational acceleration
E_{LIP}	...	"Orbital energy" of linear inverted pendulum
$x_{capture}$	...	Capture point location of linear inverted pendulum
$\mathbf{F}$	...	Force vector of virtual model control components
$\mathbf{J}$	...	Jacobian
λ	...	Feed-forward stimulation signal of virtual muscles in [41]
u_i	...	Virtual neuron i in [144]

Chapter 3

Symbol		Description
l_{opt}	...	Optimum muscle length
F_{max}	...	Maximum isometric muscle force
v_{max}	...	Maximum muscle contraction velocity
m_t	...	Thigh mass
m_s	...	Shank mass
m_f	...	Foot mass
l_t	...	Thigh length
l_s	...	Shank length
l_f	...	Foot length

Chapter 3 (cont'd)

Symbol		Description
r_t	...	Thigh joint radius
r_s	...	Knee joint radius
r_f	...	Ankle joint radius
τ_{max}	...	Maximum actuator output torque
v_{max}	...	Maximum actuator output velocity
m_r	...	Robot mass scaling factor
m_h	...	Human mass scaling factor
l_r	...	Robot length scaling factor
l_h	...	Human length scaling factor
g_r	...	Robot gravitational acceleration scaling factor
g_h	...	Human gravitational acceleration scaling factor
t_r	...	Robot time scaling factor
t_h	...	Human time scaling factor
F_r	...	Robot force scaling factor
F_h	...	Human force scaling factor
τ_r	...	Robot torque scaling factor
τ_h	...	Human torque scaling factor
v_r	...	Robot velocity scaling factor
v_h	...	Human velocity scaling factor
τ_{load}	...	Applied load torque
k_s	...	Compliant element stiffness
$\Delta\theta$	...	Compliant element deflection
θ_m	...	Motor position
$\dot{\theta}_m$	...	Motor velocity
$\dot{\theta}_{md}$	...	Desired motor velocity
τ_d	...	Desired SEA torque
θ_l	...	Load position
τ_l	...	Load torque
J_l	...	Load inertia
l_{com}	...	Shank center of mass location
J_s	...	Shank inertia
θ_k^{ref}	...	Reference position of knee trajectory
$\dot{\theta}_k^{ref}$	...	Reference velocity of knee trajectory
$\ddot{\theta}_k^{ref}$	...	Reference acceleration of knee trajectory
k_p	...	Reference trajectory position feedback gain
k_d	...	Reference trajectory velocity feedback gain
θ_k	...	Measured joint angle
$\dot{\theta}_k$	...	Measured joint velocity

Chapter 4

Symbol		Description
α_{tgt}, α_{tgt}^{des}	...	Swing-leg controller ground target
α_{tgt}^{meas},	...	Measured ground location at touchdown
ϕ_h	...	Hip angle
$\dot{\phi_h}$	...	Hip velocity
ϕ_k	...	Knee angle
$\dot{\phi_k}$	...	Knee velocity
l_t	...	Thigh length
l_s	...	Shank length
l	...	Single-segment virtual leg length
l_{clr}	...	Single-segment virtual leg clearance length
l_{com}	...	Segment center of mass location
α	...	Angle of single-segment virtual leg with respect to ground
α_0	...	Initial angle of single-segment virtual leg with respect to ground
$\dot{\alpha}$	...	Velocity of single-segment virtual leg
τ_h^{α}	...	Idealized swing-leg controller hip torque command
k_p^{α}	...	Idealized swing-leg controller hip torque proportional gain
k_d^{α}	...	Idealized swing-leg controller hip torque derivative gain
τ_h^{iii}	...	Idealized swing-leg controller late-swing hip torque addition
τ_k^{i}	...	Idealized swing-leg controller stage one knee torque command
k^{i}	...	Idealized swing-leg controller stage one knee torque proportional gain
τ_k^{ii}	...	Idealized swing-leg controller stage two knee torque command
k^{ii}	...	Idealized swing-leg controller stage two knee torque proportional gain
α_{thr}	...	Idealized swing-leg controller stage three trigger angle
$\Delta\alpha_{thr}$	...	Idealized swing-leg controller stage three trigger threshold
τ_k^{iii}	...	Idealized swing-leg controller stage three knee torque command
k^{stp}	...	Idealized swing-leg controller stage three stopping knee-flexion stiffness
$\dot{\alpha_{max}}$	...	Idealized swing-leg controller stage three maximum return velocity
μ_i	...	Coulomb friction coefficient at drivetrain stage i
ν_j	...	Viscous friction coefficient at joint j
g	...	Acceleration due to gravity
t	...	Time
θ	...	Measured joint position
$\dot{\theta}$	...	Measured joint velocity
$\ddot{\theta}$	...	Measured joint acceleration
J	...	Drivetrain stage rotational inertia estimated from CAD
θ_j^{meas}	...	Measured joint angles
τ_j^{des}	...	Desired joint torques

Chapter 4 (cont'd)

Symbol		Description
$\dot{\theta}_m^{des}$	...	Desired motor velocities
$\dot{\theta}_m^{meas}$	...	Measured motor velocities
τ_m^{app}	...	Applied motor torques
F_c^{app}	...	Force transmitted by cable drive
θ_{SEA}^{meas}	...	Measured spring angles
f_c	...	Commanded torque filter frequency
V	...	Motor voltage
i	...	Motor current
R	...	Motor resistance
L	...	Motor inductance
k_{EMF}	...	Back-EMF constant
$\dot{\theta_m}$	...	Simulation motor velocity
τ_f^c	...	Simulation Coulomb friction torque applied to each bearing stage
k_c	...	Simulation cable stiffness
v_c	...	Simulation cable damping
$\bar{\tau}$	...	Normalized commanded joint torques
$\tilde{x}$	...	Horizontal position normalized by leg-length
$\tilde{y}$	...	Vertical position normalized by leg-length
$\tau_{\mathbf{meas}}$	...	Spring torque measurement

Chapter 5

Symbol		Description
F^m	...	Contraction force of muscle m
F_{max}^m	...	Maximum isometric force of muscle m
f_l	...	Force-length relationship
f_v	...	Force-velocity relationship
A^m	...	Activation of muscle m
$\dot{A}^m$	...	Activation rate of change of muscle m
S^m	...	Stimulation of muscle m
S_0^m	...	Pre-stimulation of muscle m
S_{thr}	...	Hamstring activation threshold
τ_{ecc}	...	Excitation-contraction coupling constant
G_n^m	...	Feedback gain contributed by muscle n to muscle m
P_n^m	...	Proprioceptive signal contributed by muscle n to muscle m
t	...	Time

Chapter 5 (cont'd)

Symbol		Description
Δt_n^m	...	Neural transport delay from muscle n to muscle m
L_n^m	...	Proprioceptive length feedback signal from muscle n to muscle m
l_{opt}^n	...	Optimum length of muscle n
l_{ce}^n	...	Contractile element length of muscle n
l_{off}^n	...	Offset length of muscle n
V_n^m	...	Proprioceptive velocity feedback signal from muscle n to muscle m
v_{ce}^n	...	Contractile element velocity of muscle n
v_{off}^n	...	Offset velocity of muscle n
α_{tgt}, α_{tgt}^{des}	...	Swing-leg controller ground target
α_{tgt}^{meas},	...	Measured ground location at touchdown
α	...	Angle of single-segment virtual leg with respect to ground
α_0	...	Initial angle of single-segment virtual leg with respect to ground
$\dot{\alpha}$	...	Velocity of single-segment virtual leg
l_{mtu}	...	Muscle-tendon unit length
l^m	...	Length of muscle m
l_0^m	...	Rest length of muscle m
l_{ce}^m	...	Contractile element length of muscle m
l_{se}^m	...	Series element length of muscle m
l_{sl}^m	...	Slack length of muscle m
l_{off}^m	...	Offset length of muscle m
v_{ce}^m	...	Contractile element velocity of muscle m
v_{off}^m	...	Offset velocity of muscle m
r_h^m	...	Moment arm of muscle m spanning the hip
r_k^m	...	Moment arm of muscle m spanning the knee
ϕ_h	...	Hip angle
$\dot{\phi}_h$	...	Hip velocity
ϕ_k	...	Knee angle
$\dot{\phi}_k$	...	Knee velocity
$\phi_{h,r}^m$	...	Hip angle rest-length offset for bi-articular muscle m
$\phi_{k,r}^m$	...	Knee angle rest-length offset for bi-articular muscle m
α_r^m	...	Rest-length offset expressed in terms of α to calculate bi-articular muscle length
M	...	Selector gain for stage ii neuromuscular swing control
τ	...	Impedance control commanded joint torque
k_1	...	Impedance control proportional gain
k_2	...	Impedance control cubic proportional gain
b	...	Impedance control damping gain
θ	...	Measured joint position

Chapter 5 (cont'd)

Symbol		Description
$\dot{\theta}$	...	Measured joint velocity
θ_{des}	...	Desired joint set-point location
ϕ	...	Mini-V Vectran cable diameter
F_{max}	...	Mini-V Vectran rated force
μ_i	...	Coulomb friction coefficient at drivetrain stage i
ν_i	...	Viscous friction coefficient at drivetrain stage i
J	...	Drivetrain stage rotational inertia estimated from CAD
θ_j^{meas}	...	Measured joint angles
F_{mus}^{des}	...	Desired muscle forces
τ_j^{des}	...	Desired SEA torques generated by muscle
$\dot{\theta}_m^{des}$	...	Desired motor velocities
$\dot{\theta}_m^{meas}$	...	Measured motor velocities
τ_m^{app}	...	Applied motor torques
F_c^{app}	...	Force transmitted by cable drive
θ_{SEA}^{meas}	...	Measured spring angles
$\tilde{x}$	...	Horizontal position normalized by leg-length
$\tilde{y}$	...	Vertical position normalized by leg-length
τ	...	Commanded SEA torque
$\dot{\theta}_{motor}$	...	Commanded SEA motor velocity

Chapter 6

Symbol		Description
$\Delta\theta$	...	Spring deflection
$\Delta\theta_{max}$	...	Maximum spring deflection
(x, y)	...	Cam profile points
(x_i, y_i)	...	Convex cam profile points
(x_0, y_0)	...	Cam point of rotation
(x_n, y_n)	...	Last contact point between cable and cam
$\tau_{\mathbf{des}}$	...	Desired spring torque
$\tau_{\mathbf{des,max}}$	...	Maximum desired spring torque
$\mathbf{F_{elastic}}$	...	Elastic element force
Δs	...	Change in elastic element length
$(r_{spring,x}, r_{spring,y})$	...	Elastic element contact point with NLS housing
$\tau_{\mathbf{NLS}}$	...	NLS torque
r_{cam}	...	Vector from (x_0, y_0) to (x_n, y_n)

Chapter 6 (cont'd)

Symbol		Description
$J(...)$	...	Cost functions used to optimize cam profile
t_{rise}	...	Spring rise time
J_m	...	Reflected SEA inertia at the spring
θ_m	...	Motor position
$\dot{\theta}_m$	...	Motor velocity
$\ddot{\theta}_m$	...	Motor acceleration
τ_{motor}	...	Torque applied by motor
σ_f	...	Rubber tensile strength
A_o	...	Rubber Cross-sectional area
$r_{cam,max}$	...	Maximum cam radius
τ_{max}	...	Maximum spring torque
F_{req}	...	Rubber force required to achieve τ_{max}
β	...	Angle between $r_{cam,max}$ and F_{req}
F_{rubber}	...	Force generated in rubber
λ	...	Rubber stretch
L_o	...	Unstretched rubber rest length
ϕ	...	Polyethylene cable diameter
F_{max}	...	Polyethylene maximum force
f	...	Frequency of sinusoidal position commands
η_{motor}	...	Motor efficiency
η_{gears}	...	Gear efficiency
k_i	...	Constitutive rubber model stiffness of element i
η_i	...	Constitutive rubber model damping of element i
μ	...	Constitutive rubber model viscosity
δ	...	Constitutive rubber model deflection
$\dot{\delta}$	...	Constitutive rubber model deflection velocity
θ	...	Actuator testbed spring deflection
F	...	Constitutive rubber model force
$\dot{F}$	...	Rate of change of constitutive rubber model force
σ	...	Stress
ϵ	...	Strain
$k*$	...	Instantaneous stiffness of constitutive rubber model
δ_i	...	Deflection set-point i of constitutive rubber model at which to change stiffness
τ_{rubber}	...	Torque developed in rubber
n	...	Gear reduction between motor and spring
k_t	...	Motor torque constant

Chapter 6 (cont'd)

Symbol		Description
I, i	...	Motor current
$\mathbf{x}$	...	State vector
$\dot{\mathbf{x}}$	...	Derivative of state vector
$\mathbf{y}$	...	Output vector
$\mathbf{u}$	...	Control vector
$\mathbf{A}(\theta)$	...	State matrix as a function of θ
$\mathbf{B}$	...	Input matrix
$\mathbf{E}(\theta)$	...	Offset matrix as a function of θ
$\mathbf{C}$	...	Output matrix
L	...	Observer gain matrix
$\hat{x}$	...	State vector estimate
$\hat{\dot{x}}$	...	Derivative of state vector estimate
N	...	Observability matrix
$\mathbf{A_{obs}}$	...	Observable state matrix
$\mathbf{A_{unobs}}$	...	Unobservable state matrix
$\mathbf{C_{obs}}$	...	Observable output matrix
$\mathbf{C_{unobs}}$	...	Unobservable output matrix
$\tau_{\mathbf{des}}$	...	Desired torque
$\tau_{\mathbf{obs}}$	...	Observed torque
$\tau_{\mathbf{actual}}$	...	Actual torque
$k_{s,soft}$	...	Soft linear spring stiffness
$k_{s,stiff}$	...	Stiff linear spring stiffness

Chapter 1

Introduction

This thesis presents work to transfer controllers based on decentralized neuromuscular control strategies of human locomotion onto powered, segmented robotic legs. Chapter 2 surveys current legged locomotion control approaches and proposes that controllers based on decentralized neuromuscular control of human locomotion can improve performance of legged robotic systems. The requirements and mechanical design of dynamically scaled robots that match the performance envelope of human legs is discussed in Chapter 3. The robots developed in this chapter are subsequently used to evaluate decentralized swing-leg controllers on robotic hardware. A swing-leg placement controller based on local feedbacks is evaluated in simulation and hardware in Chapter 4. The neuromuscular interpretation of this controller is evaluated in Chapter 5, where its performance is also compared to state-of-the-art swing-leg controls. To improve performance of the robot's series elastic actuators (SEAs), a design methodology to create compact nonlinear springs is presented in Chapter 6, and the performance of an SEA using a prototype spring created using this methodology is compared to traditional SEAs that use linear, metal springs. Chapter 7 presents conclusions and future extensions to this work.

1.1 Motivation

Locomotion control of powered, segmented robotic legs is critical to humanoid robot and prosthetic applications. Current control approaches of such devices primarily use centralized

planning and tracking, a technique commonly used in humanoid robots, or mimic predefined motion patterns extracted from normal gait, a technique commonly used in prosthetic and orthotic devices. While both of these approaches produce functional gaits, neither approach has identified control strategies that can match the stability, maneuverability, and adaptability exhibited by able-bodied humans.

State-of-the-art centralized planning and tracking-based locomotion controllers for humanoid robots rely on the concept of "Zero Moment Point" (ZMP), originally proposed by Vukobratović *et al.* nearly fifty years ago [162]. The operating principle behind ZMP control is to align the robot's center of mass (COM) with the foot's center of pressure (COP) so that no moments are applied to the robot during ground contact; this generates a statically stable robot pose whenever the robot is in contact with the ground. Walking is realized by pre-computing COM trajectories that satisfy the ZMP criterion, and then executing the trajectories with the robot via full body motion control using a combination of inverse dynamics and inverse kinematics [31][66]. This approach lends itself well to both position and force controlled robots [49], and has even been shown to enable running behavior, indicated by a short flight phase during locomotion [67]. To realize these controllers, a highly accurate dynamic model of the system, as well as an accurate estimate of the system's full state is necessary; therefore, this approach cannot be applied to robotic prosthetic and orthotic devices where the human user's state is unknown and cannot be fully controlled.

Due to the inability to accurately estimate a human's full state, methods which replay predefined motion patterns extracted from healthy human gait are prevalently used in human-in-the-loop robotic locomotion applications. Replayed patterns include joint impedance [8][139][140][143], joint motion [50][1][155][89], or a combination of the two [4]. These approaches enable individuals wearing robotic assistive devices to walk over level and sloped ground at various speeds. However, they are not well suited to highly dynamic situations which require motions not explicitly coded into the device's motion library, like trip recovery, for example.

Heuristic locomotion controllers that encode desired motions via control laws are often used

to control fully robotic systems. Heuristic controllers essentially encode motion libraries for a broad set of motions. As a result, they are able to react dynamically to terrain variations and other disturbances. Many heuristic controllers exist, including ones inspired by central pattern generators identified in biological systems [58], and controllers which enforce desired robot poses during various states of a finite state machine [175]. Other types of heuristic controllers based on the spring-mass model and its variants [9][83] capture human center of mass dynamics and ground reaction forces [9][38][39], and are highly dynamic and robust to disturbances [173]. A decentralized, reflex-based neuromuscular model of human locomotion based on these models [35] also exhibits improved robustness and dynamism compared to approaches that rely on centralized planning and joint motion replay [133]. A portion of this controller has been used in an active ankle-foot prosthesis to modulate ankle torque during stance, allowing a wearer to walk at 1.0 m/s across level ground and traverse up and down 11^o slopes [26]. These results suggest that neuromuscular controls may enable highly dynamic performance in both humanoid robots and powered prostheses. However, it is unknown if this control approach can be transferred to real-world systems with multiple leg segments.

This thesis investigates the feasibility of controlling multi-segmented robotic legs with the proposed neuromuscular control scheme by transferring its decentralized swing-leg control to robotic hardware. Its use may enable more robust foot placement in the presence of unexpected locomotion disturbances compared to traditional control approaches, important for maintaining dynamic stability for both fully robotic systems and powered prostheses, and enable legged robotic systems to react to locomotion disturbances in a human-like way. Specifically, work in this thesis investigates two hypotheses. Hypothesis one posits that the proposed decentralized swing-leg controllers enable more robust foot placements into ground targets than state-of-the-art impedance controls. Hypothesis two posits that neuromuscular swing-leg control enables more human-like motion than state-of-the-art impedance control.

1.2 Approach

A model-based design approach is used to transfer the swing-leg controllers to robotic hardware. The hardware platforms "Robotic Neuromuscular Leg" (RNL) are developed to transfer and evaluate control strategies based on decentralized neuromuscular control on powered, segmented robotic legs. The platforms draw inspiration from human musculature by actuating joints via cables attached to SEAs, similar to the mechanical arrangement of tendons and muscles. Performance requirements of each actuator are defined by the primary muscle groups used in a planar, neuromuscular human walking model [35]. High-fidelity simulations of the RNL hardware platforms are created to investigate the feasibility of transferring neuromuscular controls, pre-tune hardware gains via optimization, and serve as a benchmark for hardware experiments.

An idealized version of the swing-leg controller with mono-articular actuation is transferred to RNL first and evaluated with foot placement experiments. The neuromuscular swing-leg controller is then transferred to RNL. To accomplish this, the robot is modified to include bi-articular actuators, which enables it to meet its actuation targets while maintaining a human-like mass distribution and exhibit more human-like leg behavior. The performance of these controllers, specifically their ability to robustly place feet into desired ground targets at touchdown and the similarity of ankle point trajectories traced out by the robot using each of the controllers, is then compared to a version of impedance control, a state-of-the-art control method for fully robotic systems and powered prostheses.

Hardware experiments conducted with the RNL platforms reveal the benefits of variable stiffness actuators for humanoid robots, which overcome torque resolution and actuation bandwidth tradeoffs present in classical SEAs with single-stiffness, linear springs. In size and mass constrained systems like the RNL testbeds, variable stiffness actuation can be implemented with passive nonlinear rubber springs that encode a single desired torque-deflection profile. A synthesis method for compact nonlinear springs with user-defined torque-deflection profiles that use rubber as their compliant element is presented, along with state estimation methods to account

for viscoelastic rubber behavior. Benchtop experiments compare the performance of an SEA with the developed nonlinear spring to SEAs with single stiffness, linear, metal springs.

1.3 Thesis Contributions

Work presented in this thesis contributes to the state-of-the-art knowledge in the fields of legged robotic system control and series elastic actuator design and control. The contributions described in this thesis are:

- **Transfer and evaluation of decentralized neuromuscular swing-leg control strategies onto robotic hardware.** While developed decentralized neuromuscular control strategies enable robust locomotion in simulation and a portion of the controller has been used in a powered ankle-foot prosthesis, it is unclear if the control approach can be transferred to multi-segmented robotic systems. This thesis investigates the feasibility of controlling a multi-segmented robotic leg with the proposed neuromuscular controller by transferring decentralized swing-leg controls to robotic hardware and evaluating its performance with foot placement experiments.
- **Design and control of compact nonlinear springs with user-defined torque-deflection profiles for series elastic actuators.** SEAs, popular in robots that require high force and torque control accuracy, often use linear metal springs as the torque transmitting elements in their drivetrains. Using linear springs requires design compromises between torque resolution and actuation bandwidth. Nonlinear springs overcome this tradeoff, but a synthesis method to create compact nonlinear springs for use in small and existing SEAs does not exist. This thesis presents an optimization-based synthesis method for compact nonlinear springs with user-defined torque-deflection profiles that use rubber as their compliant element. Compared to metal springs, rubber springs exhibit viscoelastic behavior like hysteresis, creep, and strain-rate stiffening. To overcome this shortcoming, state estimation

methods to account for rubber hysteresis are presented. The performance of an SEA with a nonlinear spring prototype created with the proposed synthesis method, which uses the developed observer on-line, is compared to the performance of SEAs with single stiffness, linear, metal springs through benchtop experiments.

Chapter 2

Background

2.1 Overview

Humanoid robots have been an actively researched for nearly half a century [163], yet still do not come close to matching the stability, maneuverability, and adaptability exhibited by able-bodied humans. This performance gap can be attributed both to mechanical performance limitations of robotic systems, as well as control approaches used to generate robot motions. This chapter presents an overview of existing bipedal locomotion control approaches for robotic systems and argues that, based on comparisons of reported performance of each controller type in simulation and hardware, benefits inherent to neuromuscular-inspired locomotion controllers may further bridge the human-robot performance gap compared to traditional locomotion control approaches.

2.2 Introduction

Most locomotion controllers can be divided into three broad categories. Controllers in the first category pre-plan full-body trajectories based on the robot's state, tracking generated trajectories with inverse dynamics and kinematics [137]. Such control methods, here referred to as "centralized planning and tracking" controllers, are often used in humanoid robots where the system's full state can be accurately estimated. Controllers in the second category, again used

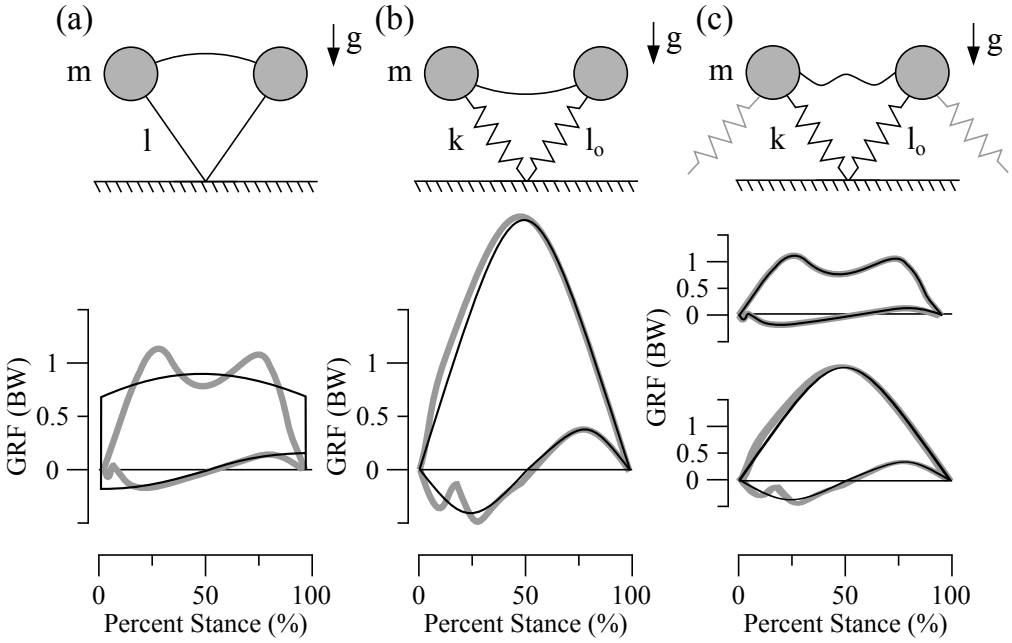

Figure 2.1: Simplified walking models. a) The linear inverted pendulum/compass gait model. b) Spring-mass model for running and hopping. c) Bipedal spring-mass model. Adapted from [39]. Despite their few parameters, simple models capture important aspects of human locomotion. The bipedal spring-mass model captures center of mass trajectories and ground reaction forces (GRFs) exhibited by humans during walking and running. Black: Simplified model GRFs. Gray: Human locomotion GRFs.

in humanoid robots, encode desired behaviors into control equations that generate robot motion commands, and are here referred to as "heuristic" controllers [23]. Controllers in the third category are common in human-in-the-loop robotics applications, and replay pre-defined motion patterns extracted from human gaits based on the current state of a robotic prosthetic or orthotic device [61]. These control approaches are discussed in the subsequent sections, where advantages and disadvantages of each approach are highlighted.

2.3 Simplified Models

Simplified models are the basis of both centralized planning and tracking and heuristic locomotion controllers. These models capture key aspects of human bipedal locomotion with few system states, such as ground reaction forces (GRFs) and center of mass (COM) dynamics,

which makes them attractive candidates for modeling bipedal locomotion of robotic systems. The following section describes the simplified walking models used by centralized and heuristic locomotion controllers.

The earliest simplified model, the compass gait/inverted pendulum model [82], models human walking, and is comprised of a lumped mass attached to an incompressible leg (Fig. 2.1a). These two parameters fully describe the model. Despite its simplicity, the model captures features of the COM's potential and kinetic energy exchange during locomotion at optimal walking speed for both bipeds and quadrupeds [15]. Additionally, the model reaches its highest point relative to the ground during midstance [77], a key characteristic of human locomotion. An extension of this model, the linear inverted pendulum model [63], constrains the COM to travel along a plane at constant angular velocity, which linearizes COM dynamics. Due to its simplified dynamics, the linear inverted pendulum model is used to reason about the COM in zero moment point (ZMP) walking implementations [162][66][67][63][65] (discussed in section 2.4). One drawback to the inverted pendulum model is that it is not able to match GRF patterns exhibited by humans during walking [93]. This discrepancy is explained by the model's inability to reproduce compliant limb behavior seen in humans due to its incompressible leg [77]. A later extension of the linear inverted pendulum model replaces the point mass with a flywheel, more closely modeling the dynamics of a physical bipedal system [101].

The spring-mass/spring-loaded inverted pendulum model for running and hopping [9] is a later model that accurately captures COM dynamics observed in human running, reproducing both experimentally observed COM trajectories and GRF patterns (Fig. 2.1b) [53]. The model has three system variables: a mass, a constant stiffness spring, and a spring rest length. Despite its simplicity, the spring-mass model for running and hopping provides a unifying theory for hopping-based robotic controllers that enables highly dynamic running and bounding behavior [109] (discussed in section 2.5). This model was later expanded to include two "legs" modeled as springs, and unified human walking and running behavior by reproducing COM trajectories and GRF patterns exhibited during both types of locomotion [39] (Fig. 2.1c).

2.4 Centralized Planning and Tracking Approaches

Centralized planning and tracking controllers pre-plan desired robot trajectories, and then execute these trajectories via full-body motion control using a combination of inverse dynamics and kinematics. The equations of motion for a bipedal robot provided by the Euler-Lagrange equations take the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau \tag{2.1}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are the generalized coordinates, velocities, and accelerations of the system, $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is the vector of centrifugal and Coriolis forces, $\mathbf{g}(\mathbf{q})$ is the vector of gravitational forces, and τ is the vector of joint torques [128]. Centralized planning and tracking approaches manipulate this equation to generate desired robot motions; therefore, a highly accurate dynamic model of the system, as well as accurate knowledge of the full system state is necessary to realize this control.

Centralized planning and tracking approaches lend themselves well to both position and force controlled robots [49], and can even enable running behavior, indicated by a short flight phase during locomotion [66]. The following subsections describe state-of-the-art centralized planning and tracking controllers in humanoid robot locomotion applications.

2.4.1 Zero Moment Point (ZMP)

The earliest bipedal walking controller resulted from stability analysis for bipedal systems [163] and centered around the concept of the zero moment point (ZMP), originally proposed by Vukobratović [162]. The ZMP describes the point where no moments are applied to a system during ground contact, and is realized with full-body poses that align the system's COM projected onto the floor with the foot's center of pressure (COP). The support polygon is defined by the convex hull of the system's ground contact points. To be stable, the ZMP must remain within

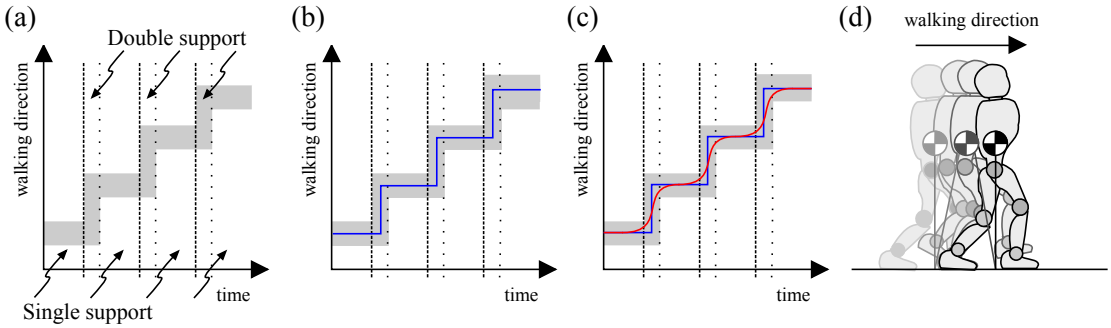

Figure 2.2: ZMP Walking. a) The support polygon trajectory is calculated based on the robot's desired foot trajectory. Gray: Support polygon trajectory. Dashed: Touch down. Dotted: Lift off. b) A ZMP trajectory that remains within the support polygon trajectory is designed to realize the desired motion. Blue: ZMP trajectory. c) A COM trajectory is generated to track the ZMP trajectory. Red: COM trajectory. d) The COM trajectory is tracked by the robot via inverse kinematics or dynamics.

the system's support polygon.

The ZMP principle can realize bipedal walking in two ways [62]. An early method executed desired leg trajectories and used a compensating mass in the robot's trunk to perform corresponding ZMP patterns to keep the biped stable. Modern implementations instead use the robot's body to execute desired ZMP patterns. In these implementations, walking is accomplished as follows (Fig. 2.2):

1. The support polygon trajectory is calculated based on the desired foot trajectory.
2. Based on the width of the support polygon, a ZMP trajectory is designed to realize the desired motion while remaining stable.
3. From this ZMP trajectory, a COM trajectory that satisfies the ZMP balance criterion is calculated. Methods exist to compute the COM trajectory from a desired ZMP trajectory in both the frequency and time domains [66]. However these methods require *a priori* knowledge of the full ZMP trajectory; therefore COM trajectories must be computed off-line and played back on the robot, making the system unable to react to disturbances or change its walking path. To overcome this limitation, [66] introduced the concept of preview control, formulated as a servo control problem, to bipedal walking. Preview control uses information about the upcoming ZMP trajectory to dynamically calculate the robot's

COM trajectory. While knowledge of the upcoming ZMP trajectory is necessary to generate COM trajectories, the preview period is short, on the order of individual steps. This allows the robot to change direction during locomotion and avoid obstacles.

4. Full-body motions to realize the desired COM trajectory are then generated through inverse dynamics or kinematics consistent with equation 2.1.

For complex systems, using the robot's full state to reason about its COM or desired footstep locations becomes computationally intractible [168]. Simplified models are therefore used in ZMP-based implementations, specifically the linear inverted pendulum model [63][66][65]. With this model, it is possible to calculate the ZMP in one dimension as

$$p_x = x - \frac{z_h}{g}\ddot{x} \tag{2.2}$$

where p_x is the ZMP location, x is the COM position, $\ddot{x}$ is the COM acceleration, z_h is the constrained constant height of the COM relative to the ground, and g is the gravitational acceleration. Saggital and frontal plane ZMP locations can be calculated separately; therefore the linear inverted pendulum model extends easily to 3D.

ZMP walking lends itself well to both position controlled robots [49], as well as force controlled robots with feedback control on joint angles [31], and has even been shown to enable running behavior, indicated by a short flight phase during locomotion [67]. This method can be combined with footstep planning techniques to avoid both static and dynamic obstacles [16]. Drawbacks to ZMP-based walking include the required computational power to compute inverse dynamics, which in turn requires a high-fidelity model of the system, the need to maintain active control over each joint at all times, and the necessity to have large feet to realize a sufficiently-sized support polygon.

2.4.2 Foot Placement Strategies

Proper foot placement is necessary to maintain dynamic balance during locomotion [150]. Foot placement strategies based on simplified locomotion models have been identified that allow humanoids to react dynamically to disturbances [101][169][94][153][154]. Of these strategies, the "capture point" and "singular LQ preview regulation" methods represent the state-of-the-art stepping strategy controllers applied to ZMP-walking humanoid robots to date, both of which make use of the linear inverted pendulum model [63]. These strategies are discussed below.

Humanoid robots can compensate for low magnitude push-disturbances by actively applying resistive torques at the hip and ankle joints [135]. High magnitude push-disturbances require the robot to take a step to remain balanced [136]. The capture point describes the ground location in which a robot must step in order to come to a complete stop after a high magnitude push [101]. By representing the humanoid robot as a linear inverted pendulum, it is possible to calculate the linear inverted pendulum's "orbital energy" E_{LIP} [64] as

$$E_{LIP} = \frac{1}{2}\dot{x}^2 - \frac{g}{2z_h}x^2 \tag{2.3}$$

derived by integrating the linear inverted pendulum model's equation of motion. Solving this equation for $E_{LIP} = 0$ yields the capture point $x_{capture}$ [101]

$$x_{capture} = \dot{x}\sqrt{\frac{z_h}{g}}, \tag{2.4}$$

the required footstep location to dissipate the system's orbital energy. This concept can be expanded to multiple steps for instances when the capture point is outside of the robot swing-leg's reach [104]. Furthermore, the capture point concept can be expanded to a "capture region" by incoporating a flywheel into the linear inverted pendulum model, which is able to impart angular momentum and accelerate the COM [101]. With this extension, the linear inverted pendulum model more closely models real-world robotic systems.

In simulation, foot placement control based on the capture region concept allows a simulated robot with a 25 kg trunk mass, negligible leg mass, 1.2 kgm^2 trunk inertia, and 0.93 m COM height to stop within 350 ms when walking at 0.5 m/s [101]. The controller can be combined with online learning techniques to improve performance on more complicated simulated robots [111].

In hardware, capture region control allowed M2V2, a 12 DOF human-sized bipedal robot to recover from 21 Ns frontal and lateral pushes while balancing on one leg [102]. Larger pushes, requiring multi-step recovery, could not be tested due to the robot's actuation limits. Formulating capture point control as an online optimization problem enabled a human-sized 33 DOF hydraulically actuated robot to recover from 18-24 Ns pushes using both single- and multi-step recoveries [137].

Using the linear inverted pendulum model in conjunction with online optimization is currently the ideal method to realize push-recovery on ZMP-based bipedal robots. [153] describes a two stage, optimal control-based method to generate foot placements online, which can be used to execute multi-step strategies to counteract push-disturbances. The linear inverted pendulum model is again used to represent the robot during ZMP calculations. In stage one, sets of ZMP-COM trajectory pairs are calculated based on the current state of the robot. As discussed in section 2.4.1, calculated COM trajectories must be consistent with the desired ZMP trajectory to create stable motion. [153] describes a method to explicitly calculate sets of non-divergent ZMP-COM trajectory pairs in real-time by modifying the preview control problem, formulated as a linear quadratic optimal control problem. In this formulation, the cost to control is set to zero. This allows for arbitrarily large torque commands, and enables the desired ZMP trajectory to be tracked within one control cycle. However, actuation limits do not allow tracking of arbitrary ZMP trajectories. Therefore, dynamically feasible trajectories are selected from the calculated ZMP-COM pairs in stage two. Dynamically feasible trajectories are defined by the maximum allowable change in the desired ZMP trajectory based on robot actuation limits, and can be found either through numerical optimization [153] or a simple iterative heuristic [154].

With this method a 12 DOF, 53 kg robot recovers from 22 Ns pushes and can continue walking when running into a 10 kg obstacle [154].

2.4.3 Virtual Model Control

Virtual model control is an alternative to ZMP-based control techniques to realize bipedal robotic locomotion [98]. Intended as a mechanically intuitive way to generate robot motion, virtual model controllers interconnect reference frames defined on the robot's structure with simulated virtual components such as springs, dampers, and energy fields [99] (Fig. 2.3). Forces $\mathbf{F}$ generated by these virtual components between reference frames are translated to joint torques τ using the Jacobian $\mathbf{J}$

$$\tau = \mathbf{J}^T\mathbf{F}. \tag{2.5}$$

Each pair of interconnected frames contains one action and one reaction frame. Forces act on the action frame; the reaction frame defines the second attachment point of the virtual component. Multiple, parallel virtual components can be interconnected between frames, allowing for the creation of complex movements by stacking simple virtual mechanisms together [98]. Virtual model control can realize classical control techniques, such as PD joint pose tracking, through sequential stacking of virtual spring-dampers with different set points. It can also be used to build virtual mechanisms that generate desired behaviors, such as "support structures" to maintain robot balance [99].

Walking behavior is generated using a finite state machine with different virtual components in each state [99][103][100]. Finite state machine-based controllers divide the gait cycle into distinct actions. Transitions between these actions are based on system variable states; finite-state machines are further discussed in section 2.5.2. Virtual model control has enabled two planar bipedal robots to walk [98][103]. Spring Turkey, a 10 kg, 0.6 m tall bipedal robot with point feet and series elastic actuators, used virtual model control to walk at 0.5 m/s over level

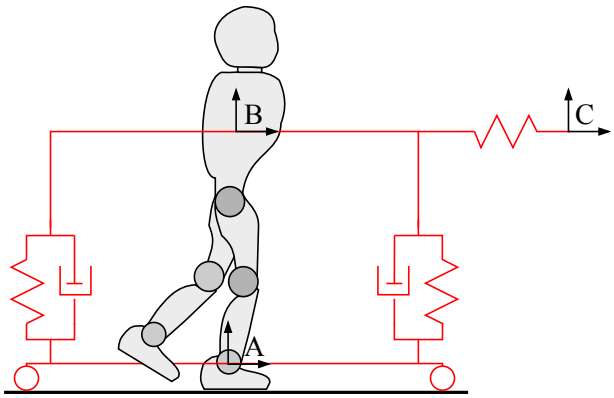

Figure 2.3: An illustration of virtual model control. Virtual components span between reference frames defined on the robot and its surrounding environment, generating virtual forces between frames. In the frame pair {A,B}, A is the action frame, and B is the reaction frame. The virtual spring-damper walker mechanism between the two frames maintains robot balance. In the frame pair {B,C}, B is the action frame, and C is the reaction frame. The virtual spring between the two frames creates forward motion. Virtual forces acting on each frame are converted to joint torques via the Jacobian.

ground while maintaining its torso pitch between $\pm 5.2^o$ [98]. Spring Flamingo, a 14 kg, 1m tall bipedal robot with actuated hips, knees, and ankles used virtual model control to walk at 0.6 m/s over level ground while maintaining its torso pitch between $\pm 2.1^o$ [103]. Furthermore, the robot could walk up and down slopes of 5^o without needing to modify properties of the controller's virtual components, and recover from external disturbances that resulted in temporary walking speed changes of 25%.

2.5 Heuristic Control Approaches

Heuristic locomotion controllers encode desired limb motion into a set of control laws based on a reduced set of state variables. State variables are not solely dependent on the robot's full pose, but other states, such as system energy and foot position to maintain constant forward velocity [106], leg angle of attack during touchdown based on time of flight [123][24], or states of virtual actuators in a robot model that differ from the physical robotic system [35]. Therefore, they do not require a high-fidelity model of the system, which makes them advantageous in appli-

cations where a robot's full system state cannot be known, as well as for complex robotic systems, whose high-dimensional full state make centralized control computationally intractable. These controllers generate robust locomotion both in simulation [175][35] and in hardware [24][26], allowing robotic systems to run and jump [109] and resist significant disturbances to their locomotion [110][25]. The following subsections present a summary of state-of-the-art heuristic locomotion controllers.

2.5.1 Hopping Robot Control

Compliant leg behavior captured by the spring-mass model for running and hopping is the basis of several control policies that realize highly dynamic and robust locomotion in robotic systems. The earliest robotic controller to achieve dynamically stable locomotion was designed for a planar, one-legged hopper that implemented its compliant limb as a prismatic air spring [109]. The controller, composed of three different control loops, was based on observations that biological legs store energy in their muscles and tendons during hopping [108]. The first control loop manipulates hopping energy of the system, maintaining constant system energy by compensating for frictional losses through active thrusting of the prismatic leg. Hopping energy is calculated by modeling the system as a spring-mass-damper [106][14]. Hopping height is modulated by injecting and dissipating system energy. The second and third control loops regulate forward motion, adjusting forward velocity and actuating the hip to maintain balance. Constant forward velocity is maintained through foot placement about the locus of the center of gravity during stance, the location over which the center of mass passes while the leg is in contact with the ground. This locus is called the "CG print." Placing the foot before or after this locus results in faster or slower forward motion. The final controller regulates attitude, or body posture during running, and is implemented as a hip torque control loop that receives feedback from the robot's hip angle. This controller maintains the robot's hip angle to be horizontal relative to the ground during stance. The three control loops are implemented via a finite state machine [109].

In hardware, this heuristic controller enables the hopper to jump 0.25 m into the air, roughly 50% of its uncompressed leg length, run up to 1.2 m/s with strides up to 0.6 m, and remain stable when subjected to horizontal disturbances [109].

This piece-wise control approach can be extended to create 3D running controls, allowing a 17 kg robot to hop in place and run over level ground at 2.2 m/s [107]. Aspects of this controller have been used in a 3D quadruped robot, which is able to navigate unstructured terrain, walk up inclines, jump 1.1 m, and carry up to 154 kg [110][25].

In contrast to this piece-wise hopping policy, analysis of the theoretical spring-mass model has also resulted in highly dynamic, robust robotic hopping and running controls for simple robots. Through a simulation of the spring-mass model for hopping and running, [36] showed that landing at a fixed angle of attack can result in self-stabilizing running behavior for a conservative spring-mass system. This controller enables a simulated spring-mass robot with an 80 kg point-mass trunk and a 1 m uncompressed spring-leg length to locomote over uneven terrain equivalent to 20% of the uncompressed spring-leg length or more. By combining self-stabilizing behaviors for various angles of attack, it is possible to derive a hopping controller that is able to regulate its hopping height and speed, and can therefore trade-off between navigating an environment quickly or overcoming large obstacles [122][123]. This behavior can be encoded by a heuristic-based controller that modulates the robot's angle of attack based on time of flight since reaching hopping apex.

The controller can also be extended into 3D. In simulation, a controller derived from analysis of the theoretical spring-mass model has resulted in time-based deadbeat control policies for a system with a human-like 80 kg mass and 1m uncompressed leg length. With this controller, the model can run at 5 m/s, tolerate ground disturbances equivalent to 30% of its uncompressed spring-leg length, and execute sharp turns in simulation [173].

In hardware, this hopping control heuristic has been applied to a simple, single-actuator robot hopper. The heuristic allows a 0.35 kg robot with a 0.19 m uncompressed leg length to cope with vertical obstacle heights equivalent to its leg length [24].

2.5.2 Keyframe-Based Locomotion Control

Finite state machines are a common method to achieve bipedal walking in simulation and robotic hardware [109][103][17][175][35]. Finite state machine-based controllers divide the gait cycle into distinct actions. Transitions between these actions are based on system variable states. During each state, various types of control are executed to realize the desired motion, including executing empirically determined control strategies [109], enabling and disabling actuators [17], and virtual model control [103]. Keyframe-based locomotion controls use desired joint poses as the control setpoint during each finite state. Two of these methods, SIMBICON and the λ-model, are discussed below.

Simple Biped Control (SIMBICON) is a state machine-based method used in the animation community to develop various gait and locomotion behaviors [175]. As in other finite state machine-based controllers, state transitions occur based on time spent in each state or sensed foot contacts with the ground. Control within each state is achieved via PD control that realizes desired robot target poses by tracking pre-defined interjoint angles. This control is applied to all robot joints, except for those located at the hip, which are responsible for controlling torso orientation, decoupling swing foot positions from the current torso orientation, and modulating the hip angle of the swing-leg to maintain balance. The latter is accomplished via a control law that uses a desired static swing-leg angle as a setpoint, and modulates around it based on the position and velocity of the robot's center of mass. Gaits can be both manually designed or learned from motion capture, and can be combined with adaptive control techniques to execute desired motions robustly while using low tracking gains.

In simulated robots, SIMBICON produces robust locomotion. Despite the controller's intuitive simplicity of simply tracking desired joint motions while maintaining balance, it is robust to 60 Ns pushes during walking and model variations of 10% leg length, can walk down steps with a height roughly equivalent to 20% of its leg length, and locomote up and down 6^o slopes [175]. Furthermore, the controller is able to robustly transition between various gait types, including

walking, high-stepping, hopping, skipping, and running when walking over level ground, and can be used for both 2D and 3D bipeds. The controller can be adapted to work with various biped morphologies and more closely emulate human walking characteristics, such as ankle actuation and passive knee swing [164]. SIMBICON's balance control law has also been combined with reflex-based neuromuscular controls (discussed in section 2.5.4), resulting in simulated locomotion that matches human locomotion recorded with motion capture [165]. To date, SIMBICON locomotion has not been implemented in hardware.

The walking model proposed in [41] interprets keyframe-based locomotion control in a neuromuscular framework. Instead of ideal torque sources, virtual biological muscles, modeled as interconnected contractile, parallel elastic, and series elastic elements attached to a virtual skeleton, are used to generate movement. Muscle dynamics are governed with a stimulation signal, which includes a feed-forward motor command λ for each of the model's 28 virtual muscles. λ sets encode sets of joint angles, the keyframes of the model. Locomotion is achieved with only two keyframes in addition to a continuously running balance controller that modifies the nominal λs of the stance leg's hip muscles. Keyframes are switched using a finite state machine, with transitions occurring based on the swing-leg's ankle position relative to the stance leg.

In simulation, the model is able to tolerate 2^o downhill and 0.5^o uphill slopes without parameter adjustment. Furthermore, walking speed can be slightly modulated between approximately 1.0-1.1 m/s by adjusting the muscle stimulation feedback gain. While the model is not able to execute highly dynamic behavior, results demonstrate the advantage granted by heuristic controls: being able to control a complex walking model by modulating a small set of parameters. Several other, non-keyframe based neuromuscular locomotion controllers exist. These are further discussed in section 2.5.4.

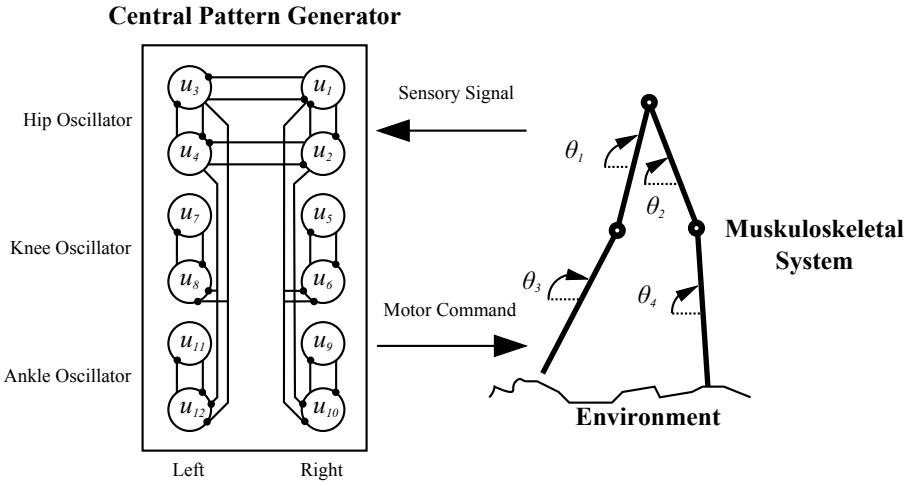

Figure 2.4: Taga's neural oscillator walking model. Reproduced from [144]. Six interconnected self-inhibitory neuron oscillators, which receive sensory feedback from a simulated biped's musculoskeletal system interacting with the ground, generate robust locomotion behavior. Running behavior can be triggered by adjusting a single parameter corresponding to the neuron's constant stimulation level.

2.5.3 Human Motor Inspired Control

Biologically inspired control policies based on central pattern generators (CPGs) have also been investigated for use on robotic systems, and are able to achieve robust locomotion and a range of locomotive behaviors with few control variables. In biology, CPGs are groups of interacting neurons that produce a rhythmic output without sensory or central input [54]. The existence of such neural circuits was first suggested in 1911, through observing locomotive behaviors of quadrupedal mammals whose nerves had been partially severed [11]. Later experiments demonstrated the existence of such locomotion controls on decerebrate cats and rats, who, despite possessing no brain control over their legs, were able to adapt to different locomotion speeds on a treadmill and transition between gaits [127][18][88]. Research on crustaceans and insects, whose central nervous system is relatively simple compared to mammals, showed that it is possible for such animals to perform complex escape-behaviors with few neurons [72][114]. The ability to execute complex behaviors with decentralized controllers that contain few variables make CPGs an interesting candidate for robotic locomotion controls.

While the existence of locomotion-governing CPGs in humans is contested [79][160], CPG-

based controls are able to realize bipedal walking behavior in simulation and hardware. Using a mathematical model of six interconnected self-inhibitory neuron oscillators [81] (Fig. 2.4), it is possible to create a bipedal locomotion controller that is able to walk over level ground and shallow slopes, is robust to various ground stiffness, and can transition between walking and running with a single parameter change [146]. This robustness can further be improved by coupling CPG-based approaches with feedback from ground contact sensors to reset the phase of the oscillators [3]. The drawback to CPG models is that no systematic methods to tune their parameters exist, however this problem has been largely overcome with optimization techniques enabled by increasing amounts of computing power [113][27][134][29].

CPGs have been used to control locomotion in sub-human sized, servo driven robots. CPG-based control was used to generate walking patterns for a 20 DOF robot, which allowed it to walk over level ground with an 8 cm step length, as well as down stairs [125]. This approach was later extended to include reflex behaviors intended to stabilize the robot when sudden disturbances occurred [176]. These additions allowed a 60 cm, 8.8 kg robot to continue walking in the presence of sudden ground changes, sudden load increases of 0.5 kg, and non-planar pushes. Another CPG-based control implementation on a servo-actuated, 8-link 3D biped, resulted in a controller that could modulate the robot's walking velocity between 0-0.2 m/s, enabling it to walk over terrain height deviations of 3 mm, and walk laterally up slopes and slippery surfaces while compensating for disturbances by adjusting its step period [28]. CPG-based control has also enabled a planar, 7 segment, 24 cm tall robot to walk at 0.39 m/s and 0.73 m/s, and dynamically transition between these speeds [80]. Furthermore, this robot could walk up an 8^o incline at a speed of 0.5 m/s.

2.5.4 Neuromuscular Locomotion

Neuromuscular controllers are another type of biologically inspired locomotion control policy. These types of control model the musculoskeletal system at various levels of detail, from

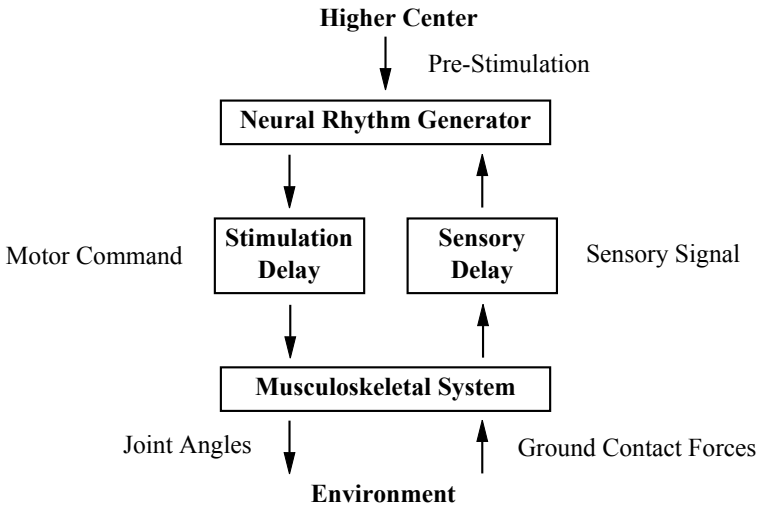

Figure 2.5: Schematic of the Taga neuromuscular locomotion model. Adapted from [144]. Motor commands issued by the neural oscillators, whose structure is shown in figure 2.4, are delayed before being applied to the model of the musculoskeletal system. Similarly, sensory feedback from the musculoskeletal system's interaction with the environment are delayed before being applied to the neural oscillators. This is done to simulate time delays present in biological systems that result from various sources such as nerve conduction velocities and muscle activations.

simple sensory time delays [144] to full biomechanical systems with virtual muscles [35], and can achieve robust 2D and 3D walking without the need for central processing. The majority of these models rely on principles described by CPGs and/or muscle reflexes.

The earliest neuromuscular controller [144] resulted from the study of CPGs [146]. Motivated by the fact that humans are able to walk robustly and dexterously across unknown environments despite having passive degrees of freedom, [146] proposed that locomotion is not achieved through feedforward control, but instead through a dynamic interaction between a CPG for walking, which is able to adapt to a changing environment through entrainment, i.e. synchronization resulting from subsystem interaction, and the musculoskeletal system. In this model, the sensed state of the musculoskeletal system, whose dynamics are partially constrained by the environment, are used as feedback to the CPG (Fig. 2.5). In biological systems, time delays exist both in the sensory feedback pathway to the CPG as well as the motor command pathway to the joint actuators. These delays result from various sources such as nerve conduction velocities and muscle activations. Muscle-like actuation was implemented in this model as delayed muscle

feedback and motor commands. Despite the presence of such delays, the simulated model could walk over level terrain with a 70 ms time delay, and over rough terrain with a 30 ms time delay, showing that global entrainment between the CPG, musculoskeletal system, and the environment can lead to stable locomotion.

The entrainment concept led to a planar neuromuscular locomotion controller for a simulated biped with 8 segments and 20 muscles, some biarticular, which could walk over level ground and was robust to parameter changes [145], as well as a 3D neuromuscular locomotion controller which was robust to 16 Ns pushes in the frontal plane and 4 Ns pushes in the saggital plane [86]. However, neither of these neuromuscular models included time delays.

Several other CPG-based neuromuscular locomotion models exist. Unlike the above model, whose muscles are ideal torque sources subject to time delays, the model described in [44] is 3D, and uses virtual mono- and bi-articular muscles to actuate each joint. Muscle activations in this model are not subject to time delays. The model can be decomposed into three sub-systems. The first system defines the mechanical and inertial characteristics of each body segment. The second describes muscle dynamics of the virtual actuators. The third describes the neuronal model which captures the CPG, empirically determined reflexes, as well as a rough model of reciprocal innervation (described below).

In simulation, this model achieves steady-state walking over level ground at a velocity of 1.4 m/s and a stride length of 1.5 m, which closely corresponds to characteristics exhibited during human locomotion. Furthermore, the model is robust to 16 Ns lateral pushes and can transition from walking to stable running at 3.0 m/s. The model could achieve a top running speed of 5.5 m/s, but stability was inversely proportional to running speed.

The model in [90] also incorporates mathematically implemented virtual mono- and bi-articular muscles that are excited by alpha motor neurons. Like the model described in [144], excitation is subject to time delays in addition to muscle dynamics similar to those in [44]. These muscles actuate a simulated seven link planar biped. Under the assumption that human locomotion patterns are generated to minimize energy consumption using gravity, alpha motor neurons

in this model are excited through a combination of a CPG and reciprocal innervation. Reciprocal innervation describes the phenomenon that the state of a muscle inhibits the excitation of its antagonist [126]. This muscle state encodes information about the state of the joint, through which gravity-induced motion can be inferred. Thus, walking not only relies on CPG-based locomotion patterns, but also afferent signals processed locally between antagonistic muscle pairs that provide information about the biped's state. While the system is able to generate locomotive behavior over level ground, walking is not as robust as the locomotion generated in [144], falling down when a 30 Ns push is applied and CPG parameters are slightly modified. This results from both a lack of global entrainment, as well as a "lazy" toe resulting from a parameter optimization to minimize consumed energy, which scuffs the ground when a perturbation is applied. Based on these results, [90] suggests that higher-level controls, in addition to both CPG and reciprocal innervation, contribute to highly-dynamic locomotion behavior.

While higher-level controls and CPGs may be necessary to generate locomotion in biological systems, it is possible to generate robust locomotion with muscle-like actuators using only reflexes, implemented as local, intermuscle feedbacks. The neuromuscular walking model in [35] is based off of the bipedal spring-mass model, and is used to control a virtual seven segment planar biped, whose legs are each actuated by seven mono- and bi-articular Hill-type muscles [48] that include both sensory and actuation delays resulting from muscle dynamics. Beginning with the bipedal spring mass model, which reproduces human COM dynamics and ground reaction force patterns seen during walking and running, the model is expanded to include two multi-segmented legs and a trunk. Muscles and corresponding intermuscle control laws are then added to generate compliant leg behavior during stance, prevent knee hyperextension, balance the trunk, and execute swing-leg behavior. Stance and swing-leg controllers are switched between based on ground contact sensors located in the ball and heel of each foot. Reflex gains are tuned manually to generate human-like walking behavior.

In simulation, the model walks steadily over level-ground at 1.3 m/s. Additionally, it can walk up and down 4 cm steps and 4^o slopes without varying reflex gains. Joint kinematics, dynam-

ics, and ground reaction force patterns are similar to those produced during human locomotion. Compared to human muscle activation patterns, the model's muscle activation patterns exhibit a correlation of at least R=0.85 during stance for all muscles, and half of the model's muscles exhibit a correlation of at least R=0.87 during swing. These findings suggest that the interplay between joint mechanics and motor control, instead of feed-forward CPG output, dominates the output of some muscles during locomotion, and, therefore, CPGs are not required to generate robust locomotive behavior.

Expansions to this model allow it to regulate its walking speed and execute speed transitions [130], as well as walk in 3D [131]. By optimizing neuromuscular parameters to achieve steady state-walking at various speeds, speed transitions between 0.8-1.8 m/s could be achieved by switching between parameter values corresponding to steady-state walking solutions during locomotion. This speed range corresponds to the typical adult human walking speed range [12]. Hip abductor and adductor muscles were added to each leg in order to realize 3D walking [131]. Desired lateral torques during leg swing were generated via SIMBICON balance control laws [175]. Like the planar model, the 3D model was able to achieve steady-state walking at 1.3 m/s, and generated GRF patterns that matched key characteristics of GRF patterns seen during human walking.

Further extensions were made to the model's swing-leg controller. Instead of implementing intermuscle control laws based on empirically observed swing-leg behavior, swing-leg control laws were refined based on the dynamics of a double physical pendulum [21], and were then reinterpreted in a neuromuscular framework [22]. Ideal-torque control of a double pendulum model, whose mechanical and inertial characteristics matched that of a human leg, resulted in a state machine-based controller that was able to robustly place its legs into arbitrary targets for a wide range of initial conditions that exceeded those typically experienced by humans [21]. This controller was able to produce motions whose kinematics and joint torque patterns were qualitatively similar to those observed in humans, achieving foot placement into desired ground targets, defined by a desired touchdown angle of a virtual leg, with an error typically less than

5^o. Furthermore, the controller was able to maintain foot placement accuracy in the presence of disturbances in both early and late swing. The neuromuscular interpretation of this controller, which includes both mono- and bi-articular muscles, achieves similar performance, while matching patterns of human muscle activation patterns observed during walking and running [22]. The ideal torque control swing-leg controller was combined with the planar neuromuscular model; after parameter optimization, the model walks steadily at 1.4 m/s, traverses rough terrain with a maximum ground height deviation of +12/-9 cm, and climbs 15 cm stairs, corresponding to 15% of its leg length [133].

2.6 Human-in-the-Loop Locomotion Controllers

A goal of studying bipedal locomotion for robotic systems is to develop powered prosthetic and orthotic devices that can re-enable disabled humans to walk. Such applications present a significant control challenge, as they place a human being in-the-loop with a robot. In fully robotic systems, the system's entire state can be estimated and controlled. In prosthetic applications, and to some extent exoskeletons, the system's full state cannot be estimated and controlled, as humans are autonomous of the robot. This necessitates alternative control approaches for human-in-the-loop applications.

Most exoskeletons augment the capabilities of able-bodied users to increase their ability to carry loads or decrease the amount of energy required to perform an action. Mechanically complex exoskeletons, like the Berkeley Lower Extremity Exoskeleton [177] and Hybrid Assistive Leg [69] use external sensors like foot switches or electrodes that measure EMG activity to estimate user intent, and then apply joint torques to assist with the inferred motion. Comparatively simple exoskeletons, like the soft lower-extremity exosuit developed in [167], are designed to reduce the energetic cost of walking, and impose joint actuation based on measured gait phase. All of these control approaches require an able-bodied human in-the-loop, and can therefore not be applied to prosthetic devices.

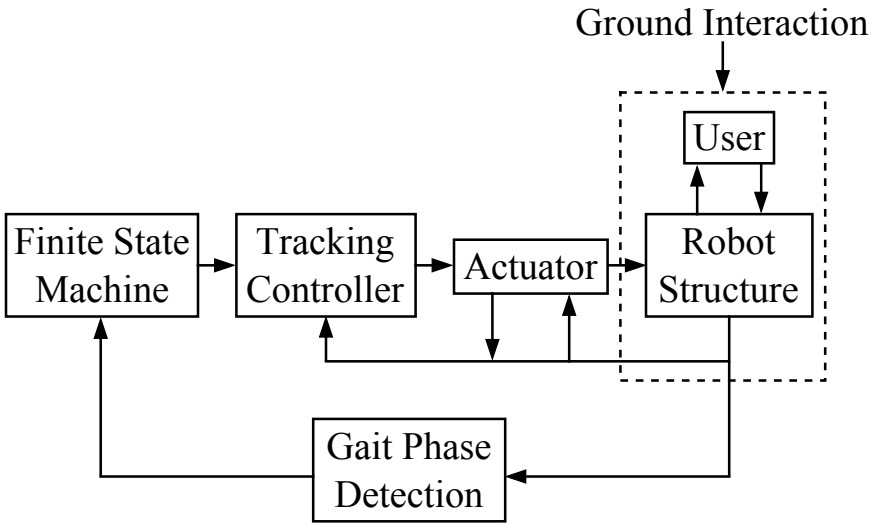

Figure 2.6: General schematic of locomotion controllers for human-in-the-loop applications. Adapted from [61]. A pre-defined motion characteristic is replayed on the prosthetic or orthotic device based on the user's gait phase.

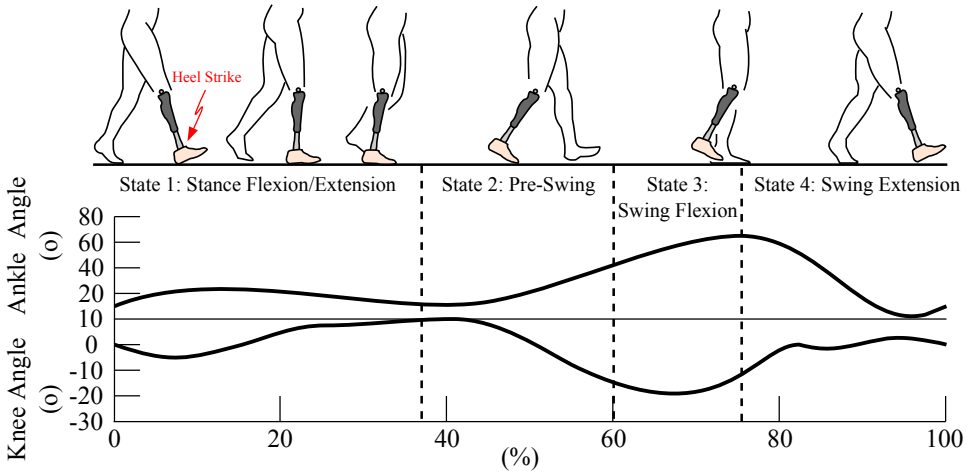

Figure 2.7: An example set of finite states during the gait cycle. Adapted from [139]. A finite state machine, initiated during prosthesis heel-strike, generates a desired joint trajectory throughout the gait cycle by tracking desired joint impedances extracted from normal human gait. Traces: Generated joint trajectory for the knee and ankle as a function of percent stride.

Controllers which replay pre-defined motion patterns extracted from normal human gait are the predominant control method for robotic prosthetic devices and exoskeletons that target paraplegic users to date [45][61][166]. The general framework for such controllers is shown in figure 2.6. The operating principle behind these control approaches is that walking is a periodic motion. Based on the current gait phase, which can be detected through a combination of pressure, joint angle, joint velocity, and joint torque [140][91][10][4], the controller uses a finite state machine to replay a pre-defined motion pattern to execute desired stance and swing-leg behavior (Fig. 2.7). Pre-defined motion patterns include joint position [50][91][52], joint torque generated by joint impedance tracking [10][139][140], or a combination of these patterns [4]. To enable a variety of actions with a single prosthetic device, such as ground level walking, performing sit-to-stand motions, and stair climbing, intent recognition has been implemented on robotic prosthetic devices [159][141][142]. Using motion libraries specifically developed for different ground slopes, amputee subjects were able to walk up inclines up to 10^o [143].

While replay of pre-defined motion patterns produces a functional gait, this approach is only suitable in situations where an appropriate trajectory is encoded into the device's motion library. Such approaches fail in highly dynamic situations and instances where normal gait is disturbed, for example during tripping and foot scuffing. Methods to dynamically generate joint trajectories have been proposed [71], but require keyframes, which makes them ill suited for situations that require a dynamic response. Approaches that encode joint behavior using control laws instead of trajectory examples may be better suited for locomotion applications, as control laws can be formulated to deal with a larger range of conditions experienced during locomotion. To this end, portions of the muscle reflex-based neuromuscular controller discussed in section 2.5.4 have been used in an active ankle-foot prosthesis to modulate ankle torque during stance [26]. Due to its modular construction and ability to achieve robust locomotion behavior despite lacking central processing, the controller is an attractive candidate to control both robotic assistive devices and fully robotic systems. With its muscle reflex-based controls, the prosthesis allows a subject to walk at 1.0 m/s across level ground, up and down 11^o slopes, run in place, and hop

	Capture Point	Preview Regulation	Virtual Model Control
Type	Centralized (ZMP)	Centralized (ZMP)	Centralized
Implementation	Hardware	Hardware	Hardware
Application	Humanoid	Humanoid	Humanoid
Size	Human	Human	Human
Mass (kg)	35	54	14
Max. vel (m/s)	0.21	1.1	0.6
Max. disturbance	21 Ns (frontal/lateral)	22 Ns (frontal)	Unspecified
Ground type	Level	Level	Sloped

Table 2.1: Centralized planning and tracking best performance comparison.

	Spring-Mass Hopping	Keyframe	CPG	Neuromuscular	Motion Replay
Type	Heuristic	Heuristic	Heuristic	Heuristic	Replay
Implementation	Hardware	Simulation	Hardware	Simulation	Hardware
Application	Humanoid	Humanoid	Humanoid	Humanoid/Prosthesis	Prosthesis
Size	Human	Human	Sub-Human	Human	Human
Mass (kg)	17	90	9	80	70
Max. vel (m/s)	2.2	Unspecified run	0.7	1.4	1.2
Max. disturbance	Unspecified	60 Ns	0.5 kg mass increase	0.1 m stairs	10^o slopes
Ground type	Rough	Sloped	Rough	Rough	Sloped

Table 2.2: Heuristic and human-in-the-loop control best performance comparison.

[46]. However, while this device illustrates the ability to use this muscle-reflex based controller to control prosthetic devices and enable highly dynamic motions, it does not explore the full potential of this controller, as it only actuates a single joint.

2.7 Performance Comparison & Thesis Motivation

The previous sections present a summary of state-of-the-art locomotion control approaches for both fully robotic and human-in-the-loop legged systems. The best reported performance of each control method is summarized in tables 2.1 and 2.2. Direct comparison between all methods is difficult due to the variability in application and maximum system complexity that each method can control. However, assuming that methods which function on complex systems are better, since they are more suited to control hardware capable of operating outside of a controlled laboratory setting, each method can be grossly compared in terms of robustness and dynamism.

Based on reported performance, preview regulation is the best performing locomotion controller to date, allowing the most massive robot to walk at the fastest speed and tolerate the largest disturbance (Tab. 2.1). This control approach was most recently used in the DARPA Robotics

Challenge Trials [152] by Team SCHAFT. Using this approach, Team SCHAFT finished in first place, besting 11 other competitors [20]. While preview control represents the current state-of-the-art for fully robotic systems that can use centralized planning and tracking-based controllers, it cannot be used for prosthetic devices, as the full human-robot system state cannot be estimated and controlled.

The spring-mass hopping controller presented in [107] enables the most dynamic locomotion in robotic hardware systems to date. Its formulation, however, only allows it to be used on simple systems with limited degrees of freedom. While keyframe-based locomotion approaches enable the widest range of behaviors and can tolerate the largest disturbances, they have only been tested in simulation. Such pose tracking methods also require active control over all joints, again making them exclusive to fully robotic systems. Motion replay techniques, like those presented in [143] offer functional control of human-in-the-loop systems, but motions not explicitly coded into a prosthetic device's library cannot be executed. Therefore, this control approach offers limited performance when encountering locomotion disturbances such as trips.

Of the presented approaches, reflex-based neuromuscular control is the only approach that has been used to control both fully humanoid systems, albeit only in simulation [35], as well as prosthetic hardware during stance [26]. In both cases, the controller enables robust locomotion. In simulation, the neuromuscular controlled model can walk at speeds up to 1.4m/s, as well as over rough terrain and up stairs [133]. Since the reflex-based locomotion controller does not require central processing and can be localized to stand-alone robotic devices, it also performs well in single degree of freedom prosthetic applications, allowing a user to walk across level and inclined ground. Whereas these results suggest that the neuromuscular controller is a promising alternative control method for both fully robotic systems and powered prostheses, it is unclear if the controller can be transferred to multi-segmented robotic legs. The goal of this thesis is to investigate the feasibility of controlling a multi-segmented robotic leg with the proposed neuromuscular control approach, which may enable both robots and powered prostheses to react to locomotion disturbances dynamically and in a human-like way.

To evaluate the feasibility of neuromuscular control of robotic hardware, work in this thesis focuses on neuromuscular swing-leg controls, important for maintaining dynamic stability of both fully robotic systems and powered prostheses in the presence of unexpected locomotion disturbances, such as trips and pushes. Specifically, work in this thesis investigates two hypotheses. Hypothesis one posits that proposed decentralized swing-leg controls enable more robust foot placements into ground targets than state-of-the-art impedance controls. Hypothesis two posits that neuromuscular swing-leg control enables more human-like motion than state-of-the-art impedance control. The transfer of neuromuscular control to hardware and its comparison to state-of-the-art impedance control is accomplished in the following chapters as follows:

- Chapter 3 presents the design of RNL, a three segment, cable-driven, antagonistically actuated robotic leg with joint compliance. The robot's size, weight, and actuation capabilities correspond to dynamically scaled human values. Experiments presented in this chapter test if it is possible to design and control a robot that matches human-like performance. The hardware platform developed in this chapter is used in subsequent experiments to transfer and evaluate the neuromuscular swing-leg controller.
- Chapter 4 transfers and evaluates an idealized version of the swing-leg controller with mono-articular actuation to RNL. A model-based design approach is used to transfer the swing-leg controller. A high-fidelity simluation of the robot is created to investigate the feasibility of transferring neuromuscular controls, pre-tune hardware gains via optimization, and serve as a benchmark for hardware experiments. The swing-leg controller is then transferred to RNL and evaluated with foot placement experiments. The results suggest that the swing-leg controller can accurately regulate foot placement of robotic legs during undisturbed and disturbed motions, although discrepancies exist between human motions and those executed by the robot.
- Chapter 5 and transfers and evaluates the full neuromuscular swing-leg controller on RNL. The model-based design approach is extended to this setup, culminating in hardware foot

placement experiments to validate the controller's performance. To compare the controller's performance to state-of-the-art locomotion controls, impedance control is then implemented on the robot and evaluated with the same set of foot placement experiments. Results of these experiments suggest that both the idealized and neuromuscular swing-leg controllers improve over the current state-of-the-art by enabling robots to robustly place feet into a range of ground targets using only a single set of gains. Furthermore, the ankle trajectory traced out by the robot under neuromuscular control is more human-like than the trajectories traced out under the proposed idealized control and impedance control.

- Chapter 6 describes a design methodology of compact nonlinear springs with user defined torque-deflection profiles to improve the performance capabilities of RNL's SEAs. A prototype spring is created and evaluated on an actuator testbed. Methods to estimate spring torque, necessary to compensate for viscoelastic effects of the rubber, are discussed and evaluated via simulation and hardware experiments. To investigate potential advantages of nonlinear springs to series elastic actuators, testbed experiments are conducted that compare the torque tracking ability of low stiffness linear springs, high stiffness linear springs, and the developed nonlinear springs.
- Chapter 7 summarizes the work presented in this thesis and discusses possible future extensions to realize a fully-walking bipedal system, whose motion is regulated using decentralized neuromuscular control.

Work in this thesis demonstrates that neuromuscular swing-leg controls can be transferred to robotic hardware and generate human-like leg behavior. This work validates neuromuscular controls as an alternative control method of multi-segmented robotic legs, which could enable existing and future powered, segmented legged systems to exhibit more robust, dynamic, and human-like locomotion.

Chapter 3

Design and Development of RNL, a Testbed for Robotic Neuromuscular Controllers

Material in this chapter is partially based on:

A. Schepelmann, M.D. Taylor, and H. Geyer
Development of a testbed for robotic neuromuscular controllers.
Proceedings of Robotics: Science and Systems, VIII:385-392, 2012. [117]

A. Schepelmann, J. Austin, and H. Geyer
Evaluation of decentralized reactive swing-leg control on a powered robotic leg.
Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems: 381-386, 2015. [119]

Decentralized neuromuscular control strategies of human locomotion may improve the reactiveness and dynamism of powered, segmented robotic legs compared to traditional control approaches. A hardware platform that matches the performance envelope and actuation method of human legs is necessary to evaluate these control strategies and test their full advantages to legged robotic systems. Such a hardware platform is not currently available. To overcome this limitation, the Robotic Neuromuscular Leg (RNL) test platforms are created, whose development is outlined in this chapter. RNL1 investigates the feasibility of building and controlling a robot whose size, weight, and actuation capabilities match the performance envelope of human legs. The robot is an antagonistically actuated two segment leg with translational joint compliance. Experimental results show that the robot can track fast motions corresponding to 87% of the maximum performance limit of human muscle, indicating that it is possible to build and

control a hardware platform with human-similar performance. Based on these results, RNL2 and RNL3 are created. These robots are full powered, segmented robotic legs with three segments and seven mono- and bi-articular series elastic actuators, which are used to evaluate the decentralized swing-leg controls to be discussed in the subsequent chapters.

3.1 Design Requirements

The RNL platforms are designed to test decentralized neuromuscular control strategies stemming from [35] on robotic hardware. The model in [35] (Fig. 3.1), is a planar humanoid with a trunk and two three-segment legs, actuated by virtual mono- and bi-articular Hill-type muscles [48][37]. The model interprets the bipedal spring-mass model, which unifies human center of mass (COM) dynamics during walking and running, in a human morphological framework. Muscles in [35] generate locomotion behavior primarily through intermuscle feedbacks, termed "muscle reflexes," and represent major muscle groups active during human walking. Components of the model have since been extended to regulate walking speed and enable speed transitions [130], achieve robust swing-leg placement in the presence of large disturbances [21][22], navigate rough terrain and steep inclines [133], and realize 3D walking [131]. Due to its versatility, robustness, and decentralized nature which makes it interesting to human-in-the-loop prosthesis applications, the controller represents a promising alternative to existing locomotion controllers for powered, segmented robotic legs.

Implementing the model's muscle-like actuation scheme on robotic hardware may provide several additional benefits. First, an antagonistic actuation scheme enables joint co-contraction, important for energy regulation during stance [15], especially while running [14]. Second, robot segment mass distributions enabled by distributed actuators may match human segment mass distributions more closely compared to traditional co-located actuators at the robot's joints. Third, biomechanics imparted by muscle-like actuation may decrease high-level control complexity.

Currently, no legged system exists that matches the performance envelope and actuation

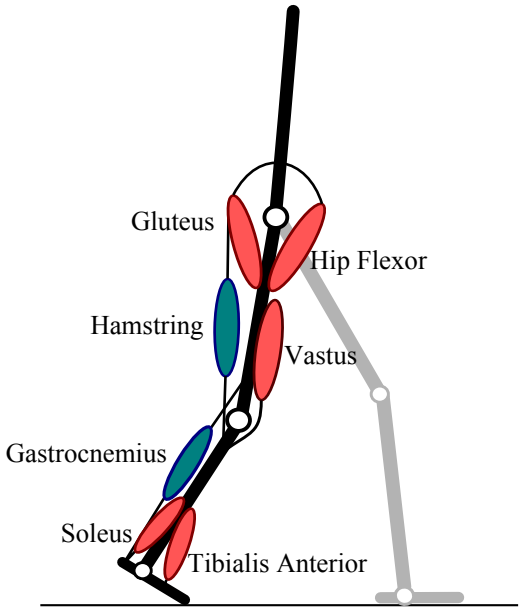

Figure 3.1: Hill-type muscles of the decentralized neuromuscular walking model in [35]. The model represents a planar humanoid with a trunk and two three-segment legs, actuated by virtual mono- and bi-articular Hill-type muscles. Black: Leg segments and trunk. Red: Mono-articular muscles. Blue: Bi-articular muscles. Gray: Other leg. (Muscles not shown.)

	Gastroc.	**Hamstring**	**Hip Ext.**	**Hip Flex.**	**Soleus**	**Tibialis Ant.**	**Vastus**
$\mathbf{l}_{opt}$ (cm)	5	10	11	11	4	6	8
$\mathbf{F}_{max}$ (N)	1500	3000	1500	2000	4000	800	6000
$\mathbf{v}_{max}$ (l_{opt}/s)	12	12	12	12	6	12	12

Table 3.1: Muscle parameters of [35]'s planar, reflex-based neuromuscular walking model.

	$\mathbf{m}_t$ (kg)	$\mathbf{m}_s$ (kg)	$\mathbf{m}_f$ (kg)	$\mathbf{l}_t$ (cm)	$\mathbf{l}_s$ (cm)	$\mathbf{l}_f$ (cm)	$\mathbf{r}_t$ (cm)	$\mathbf{r}_s$ (cm)	$\mathbf{r}_f$ (cm)
Human	8.5	3.5	1.25	50	50	20	10	5	5
RNL	2.125	0.875	0.3	25	25	10	5	2.5	2.5

Table 3.2: Human segment mechanical properties vs. RNL segment targets. Human segment mechanical properties taken from [35].

	Gastroc.	**Hamstring**	**Hip Ext.**	**Hip Flex.**	**Soleus**	**Tibialis Ant.**	**Vastus**
τ_{max} (Nm)	9.38	37.5	18.75	25	25	5	45
$\mathbf{v}_{max}$ (rpm)	162	162	178	178	65	194	217

Table 3.3: Actuator requirements, specified at RNL's joints. Torque requirements based on dynamically scaled maximum isometric force of each muscle in table 3.2, and RNL joint radii. Speed requirements based on dynamically scaled maximum joint speeds reported in [171].

scheme of [35]'s neuromuscular model (Tab. 3.1). Yet, such a hardware platform is necessary to test the full advantages of neuromuscular control to legged robotic systems. The RNL hardware platforms are designed to overcome this limitation. Their design requirements are driven by three themes: dynamic similarity, antagonistic actuation, and leg compliance.

3.1.1 Dynamic Scaling

The RNL platforms aim to match human leg performance, but constructing a full-sized humanoid is costly and difficult for individual researchers to work with. For cost and safety considerations, RNL is designed to be half the size and a quarter of the weight of a human leg. Due to this size and weight reduction, it is necessary to ensure that the dynamic behavior exhibited by RNL matches that of a full-sized human leg. This is accomplished through dynamic scaling.

Dynamic scaling uses fundamental physical variables to define relationships between a system's quantities at different scales. This approach was formalized by Buckingham [13], and is often applied in aerospace and fluid engineering applications. In mechanical systems, fundamental units are mass, length, and time. The robot's mass and length targets define these scaling factors as m_r/m_h (kg)/(kg) = 1/4 and l_r/l_h(m)/(m) = 1/2, respectively. Since the robot is exposed to the same gravitational field as humans, the relationship g_r(m/s^2)=g_h(m/s^2) must hold.

With these three scaling factors, dynamic scaling factors for time, force, torque, and velocity are calculated through dimensional analysis. RNL's time scaling factor must have units

$$\frac{t_r}{t_h} = \frac{s}{s}. \tag{3.1}$$

It is possible to resolve these units with the above fundamental units as

$$\frac{t_r}{t_h} = \sqrt{\frac{g_r/l_r}{g_h/l_h}} = \sqrt{\frac{1}{2}}. \tag{3.2}$$

Peforming similar calculations results in force, torque, and velocity scales of F_r/F_h(N)/(N)

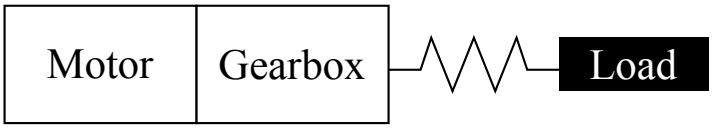

Figure 3.2: General schematic of a series elastic actuator (SEA). SEAs are characterized by the presence of a compliant element between the motor and load, and offer several advantages for torque controlled robots compared to actuators without a compliant element.

= 1/4, τ_r/τ_h(Nm)/(Nm) = 1/8, and v_r/v_h(m/s)/(m/s) = $\sqrt{2}/2$. A comparison of human segment and RNL mechanical properties calculated using these scaling factors is shown in table 3.2. The optimum length, maximum contraction velocity, and maximum isometric force of each muscle in [35] can also be used to calculate dynamically scaled RNL joint actuator no load speeds and stall torques, which are shown in table 3.3.

3.1.2 Antagonistic Actuation

The RNL platforms use cable driven series elastic actuators (SEAs) to enable multi-articular, muscle-like actuation. SEAs were originally developed by Pratt and Williamson [97] and are common in bipedal robots [103][92]. SEAs contain a compliant element between motor and load (Fig. 3.2), which offers several benefits compared to rigid actuators. First, the compliant element can be used to estimate applied load torque more accurately than using motor current. Motor current is often noisy. Strain gauges and encoders, on the other hand, can provide cleaner measurements of compliant element deflection. With this deflection measurement, applied load torque τ_{load} can then be calculated via Hooke's law as

$$\tau_{load} = k_s \Delta\theta \tag{3.3}$$

where k_s is the compliant element stiffness and $\Delta\theta$ is the compliant element deflection. Furthermore, the compliant element is often located after the drivetrain's gear stages and attached to the load; as such, no dissipative friction occurs between the measurement and torque-application in-

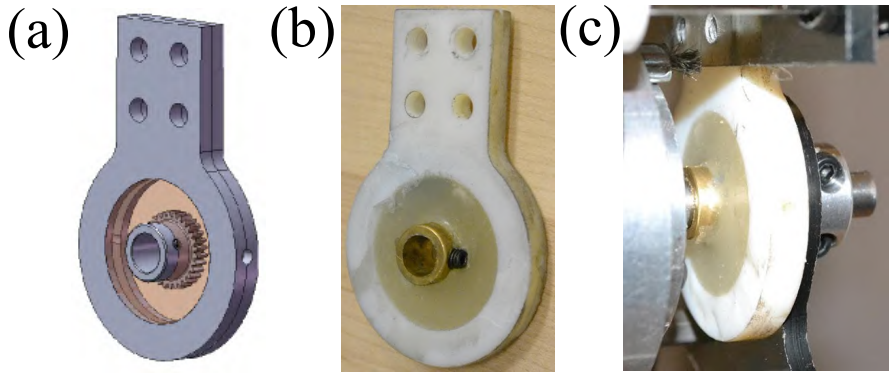

Figure 3.3: Translational joint compliance. a) CAD rendering. b) Hardware implementation of joint for RNL1. c) Close-up of installed RNL1 joint.

terfaces. Finally, the compliant element decouples motor and load inertia, which enables precise torque control with highly geared motors (at the expense of system bandwidth), including zero torque [115].

Cable drives have been used to actuate robots where size, mass distribution, and low segment inertias are critical [7][42][19]. SEAs have previously been combined with cable drives in legged systems [40] and exoskeletons [161]. In RNL, cable-driven actuators help to meet several design requirements. First, they enable human-like segment mass distributions, since actuators do not need to be co-located with a joint. Second, they allow actuation across translationally compliant joints, since cables can go slack. Cable slack also enables truly passive dynamics since they can be actively commanded to be slack. Finally, cable drives enable antagonistic actuation, where multiple actuators act on a single joint, enabling joint co-contraction.

3.1.3 Joint Compliance

Humans are not rigidly coupled kinematic chains, possessing interjoint cartilage and soft tissue around bones. To capture this aspect in RNL, translationally compliant joint designs are incorporated into the robots' structure (Fig. 3.3). The joints connect the RNL thigh and torso, as well as the shank and thigh. Joints are composed of two clamping plates, which hold a custom-molded compliant element (Shore A Hardness 70). The joints' shafts and compliant elements

	Human	RNL	Scaling Factor
Vastus Max Force (N)	6000	1500	1/4
Vastus Max Vel. (m/s)	0.96	0.68	1/4
Hamstring Max Force (N)	3000	750	1/4
Hamstring Max Vel. (N)	1.2	0.84	1/4
Max Joint Torque (Nm)	368	45	1/8
Max Joint Vel. (rpm)	153	217	$\sqrt{2}$

Table 3.4: Maximum human mechanical performance vs. RNL SEA targets

are coupled with a compliance retainer, constructed from spur gears. This allows bi-directional translation between the robot's segments, while restricting rotational motion to only occur in the robots' distal segments.

3.2 Robotic Neuromuscular Leg 1 (RNL1)

It is unclear if a dynamically scaled robot with human-like size, weight, and actuation capabilities can be built and controlled. RNL1 is created to investigate this question. RNL1 is a half-human sized, quarter-human weight, two segmented, antagonistically actuated robotic leg with joint compliance (Fig. 3.4). The following subsections detail the electromechanical design of RNL1, describe the velocity-based SEA control scheme used to control the robot's actuators, and present experiments used to evaluate whether the robot meets its "human-like" performance goals.

3.2.1 RNL1 Series Elastic Actuators

The vastus muscle is able to produce the largest maximum isometric force of all leg muscles active during locomotion. Its antagonist, the hamstring, is among the fastest muscles active during locomotion. If it is possible to meet the most aggressive human actuation goals while maintaining dynamically scaled length and mass targets, it should be possible to construct a full leg that matches RNL's overall performance goals, as the additional actuators necessary to

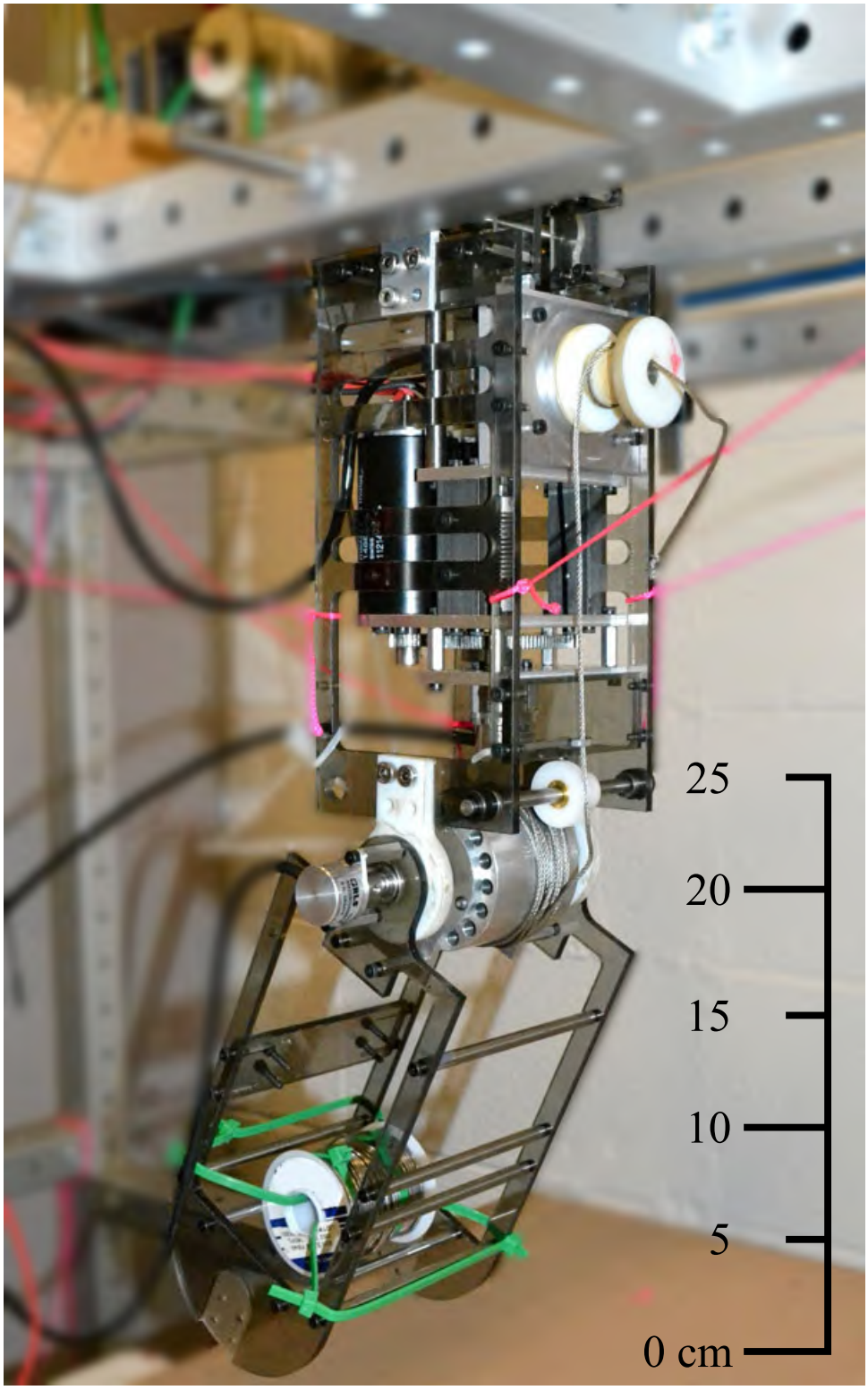

Figure 3.4: The Robotic Neuromuscular Leg 1 (RNL1) test platform.

realize the full system are smaller and less powerful. Therefore, RNL1 focuses on meeting the performance envelope defined by the vastus and hamstring in a robotic system. Table 3.4 shows human mechanical performance limits and corresponding RNL1 dynamically scaled mechanical performance goals.

The vastus and hamstring antagonistically actuate the knee joint. Therefore, RNL1 is constrained to have only this degree of freedom, accomplished by rigidly coupling the robot's thigh segment to its mounting cage. As a result, the hamstring, which in a full neuromuscular system is bi-articular (Fig. 3.1), is instead mono-articular in RNL1. The vastus SEA is located in the robot's thigh; the hamstring is located on the mounting cage, acting as the robot's "torso." With future technology transfer to prosthetic and orthotic devices in mind, RNL1's motor selection is limited to DC motors. Due to author preference, motor selection is further limited to the Maxon Motor AG catalog. Within these constraints, the only single motor able to meet the speed and torque requirements of the vastus and hamstring SEAs is the 250-watt RE65. Using this motor immediately exceed the 2 kg weight limit of the robot's thigh segment. Therefore, RNL1's SEAs use multiple DC motors geared in parallel, which yield similar motor power at reduced actuator weight. Several motor configurations match RNL1's SEA design targets (Tab. 3.5). The optimal combination of low weight, low motor inertia, torque, and speed is accomplished with four mechanically coupled RE30 motors. To limit actuator complexity RNL1's SEAs instead use two mechanically coupled RE40 motors (RE40: 148877; Maxon Motor AG), at the expense of

	4x RE30	**2x RE40**
Gear ratio	40	36
Total weight (g)	952	960
Rotor inertia (kgm^2)	0.02	0.03
Nominal torque (Nm)	14.1	13.2
Stall torque (Nm)	163	180
No load speed (rpm)	212	211
Nominal speed (rpm)	194	194

Table 3.5: RNL1 SEA motor configuration comparison. Optimal configuration is comprised of four RE30s. Two RE40s meet the same performance criteria with lower mechanical complexity, at the expense of increased rotor inertia.

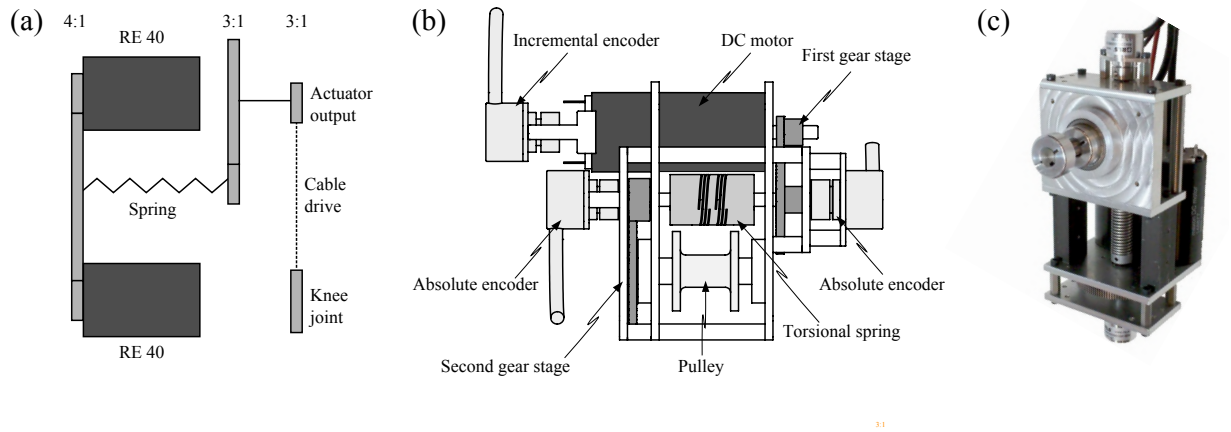

Figure 3.5: RNL1 SEA drivetrain. a) Drivetrain schematic. b) Drivetrain layout. c) Hardware implementation.

increased rotor inertia.

For compactness, the SEAs incorporate a three stage drivetrain (Fig. 3.5). The first stage mechanically couples the motors with a 4:1 reduction; the second stage aligns the output shaft's axis of rotation and the robot's joint with a 3:1 reduction. An additional 3:1 reduction is located between the SEA output shaft and joint, connected via cable drive, for a total 36:1 gear reduction in the drivetrain. The gear ratio of the external stage can be modified, which enables the same SEA design to be used for muscles with different properties.

A rotary spring coupling serves as the SEA's compliant element. For compactness, it is located between the actuator's first and second drivetrain stages. Torque measurements are realized with two absolute rotary encoders (RM22B: magnetic, analog; Renishaw PLC) located on either side of the spring. Encoders are interfaced to the host PC using an analog data acquisition card (Model 626; Sensoray Co., Inc.), yielding an effective encoder resolution of 9 bits. Off-the-shelf spring couplers are used as SEA compliant elements. To enable high-accuracy torque tracking, the SEAs must have a minimum joint torque resolution of 0.5 Nm. A 1.75 Nm/rad spring coupling with a 1.4 Nm maximum torque rating (A5Z26M0606; SDP-SI) is used to realize the desired torque resolution with the selected encoders. An incremental encoder on the shaft of one RE40 (RM22I: 9 bit, magnetic, digital; Renishaw PLC) measures motor velocity.

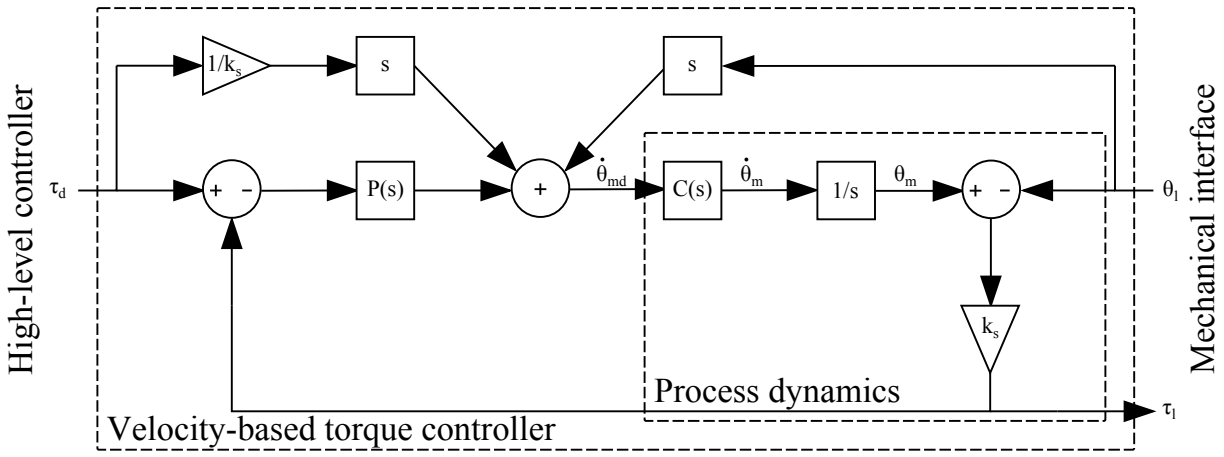

Figure 3.6: Velocity-based SEA controller schematic. P(s) is a feedback loop to compensate for model uncertainty. C(s) represents the motor controller.

This measurement is used as feedback to the SEA controller detailed in the next section. One motor controller (SOL-WHI-20/60E02; Elmo Motion Control) supplies the same current to both RE40s. Motor controllers are interfaced via RS-232 to a host PC that runs Real-Time Windows Target (Mathworks, Inc.), and receive velocity commands asynchronously at approximately 1 kHz.

3.2.2 RNL1 Joint Compliance

RNL1's floating knee joint design is shown in figure 3.3. The joint's rapid prototyped clamping plates (VeroWhitePlus; Object Ltd.), hold the compliant element, made from a two-part PMC-744 urethane rubber mixture (Smooth-on Inc.) that is cast into the plates. An RM22B encoder is mounted on the joint shaft. The encoder body is located on the shank. Relative motion between the encoder head and body, enabled by the compliance retainer design discussed previously, measures the knee position during leg movement.

3.2.3 Velocity-Based SEA Control

RNL1's SEAs generate desired torques using a velocity-based SEA control scheme [147] (Fig. 3.6). The original SEA controller proposed by [97] modulates load torque using motor torque as a control target. Recently, an alternative SEA controller was proposed [174], which modulates load torque τ_l using motor velocity $\dot{\theta}_m$ as a control target instead of motor torque

$$\dot{\theta}_m = \tau_l(\frac{1}{J_l s} + \frac{s}{k_s}) \tag{3.4}$$

where J_l is the load inertia and k_s is the stiffness of the compliant element. Velocity-based SEA control is advantageous to SEAs where the compliant element is not directly connected to the load. Since motor velocity corresponds exactly to velocity at the drivetrain output, the velocity loop automatically compensates for losses without additional tuning of the outer control loop. In addition, the controller's inverse dynamics terms only require the first derivative of motor position, which leads to increased system bandwidth, since low-pass filters with higher cutoff frequencies can be used. Velocity-based SEA control can also enable passive control while maintaining an integral gain to counteract steady-state error, when it is realized as a cascaded controller with an inner velocity loop that generates desired motor velocities via PID control that operates on the difference between desired and measured load torque [155]. Realizing passive control that incorporates an integral gain is not possible with the torque-based SEA controller originally proposed in [97].

The velocity-based SEA control formulation in equation 3.4 requires load inertia to be known. For legged systems, it is not clear what load inertia is, as load dynamics constantly change due to joint position and gait phase. However, knowledge of load inertia is not necessary in the formulation of a velocity-based SEA controller. Starting with Hooke's law (Eq. 3.3), it is possible to formulate torque control as velocity-based control by writing the motor position as a function of motor velocity $\theta_m = \dot{\theta}_m / s$, which resolves Hooke's law to

$$\dot{\theta}_m = \tau_l/(k_s s + \theta_l s). \tag{3.5}$$

Here J_l does not need to be known. Figure 3.6 shows a schematic implementation of this velocity control, in which equation 3.5 is implemented as feedforward compensation, $P(s)$ is PID feedback compensation for model uncertainty, and $C(s)$ represents the motor controller and actuator plant.

3.2.4 Hardware Experiments

RNL1's electromechanical design and control implementation is verified with motion tracking experiments. The experiments test whether the robot can meet desired performance goals by tracking fast, commanded knee trajectories with high fidelity. To simulate the inertial effects of a foot segment, a weight is attached to the bottom of RNL1's shank, increasing its total mass to 1.1 kg. The resulting mechanical properties of the shank segment are l_{com}=0.107 m, and J_s=0.005 kgm^2, where l_{com} is the distance of the shank's center of mass position from the knee pivot and J_s is the shank inertia about the knee joint.

Knee trajectories θ_k^{ref} observed in human walking (tabulated in [171]) are sped up until the joint velocity, calculated via numerical differentiation, reaches the maximum human knee joint velocity defined in table 3.4. Velocities are median filtered (filter order = 10) to eliminate artifacts resulting from differentiation. This results in references trajectories with speeds between 1.0$\times$ and 2.0$\times$ of the nominal trajectory. Calculated joint velocities and accelerations are then dynamically scaled to generate reference trajectories for RNL1 joint velocity $\dot{\theta}_k^{ref}$ and acceleration $\ddot{\theta}_k^{ref}$.

With these references, a tracking control that outputs desired actuator torques is designed. The controller includes feed-forward torque trajectories $J_s\ddot{\theta}_k^{ref}$, gravity compensation, and PD feedback compensation $k_p(\theta_k^{ref} - \theta_k) + k_d(\dot{\theta}_k^{ref} - \dot{\theta}_k)$, where k_p=15 and k_d=0.01k_p are the position and velocity feedback gains. θ_k and $\dot{\theta}_k$ are measured by an absolute encoder on RNL1's knee

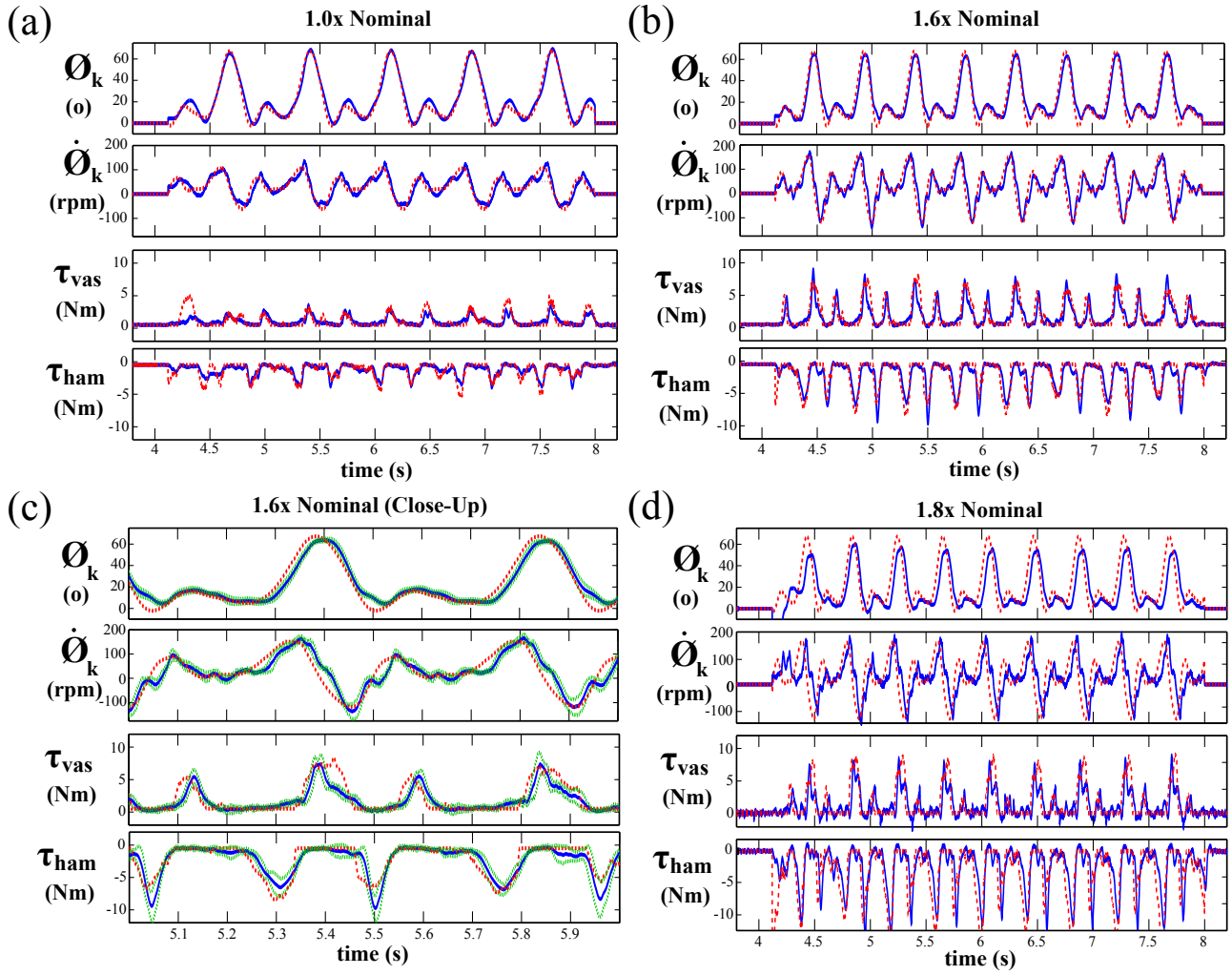

Figure 3.7: High fidelity motion experiment results. RNL1 knee position, velocity, and antagonistic actuator torques for walking trajectories corresponding to a) 1.0×, b),c) 1.6×, and d) 1.8× nominal walking speed trajectories. In experiments, motion occurred between 4 s and 8 s. Plots show mean±std. for 10 repetitions. Red lines: commanded trajectories. Blue lines: measured trajectories. Green lines ±std. of measured trajectories.

	R_{θ_k}	$R_{\dot{\theta}_k}$	$R_{\tau_{vas}}$	$R_{\tau_{ham}}$
1.0×	0.97±0.01	0.87±0.04	0.71±0.06	0.90±0.02
1.2×	0.97±0.01	0.88±0.02	0.67±0.04	0.90±0.02
1.4×	0.94±0.02	0.82±0.04	0.80±0.02	0.82±0.05
1.6×	0.95±0.02	0.85±0.02	0.80±0.01	0.80±0.03
1.8×	0.80±0.06	0.64±0.08	0.77±0.04	0.62±0.04

Table 3.6: Mean correlation coefficients (R) for trajectory following trials. n =10 for all speeds.

	t_{θ_k}	$t_{\dot{\theta}_k}$	$t_{\tau_{vas}}$	$t_{\tau_{ham}}$
1.0×	9.5±1.5	18.4±2.6	13.1±1.7	4.8±0.9
1.2×	7.2±0.9	11±0.7	10.3±1.2	5.1±0.7
1.4×	15.9±1.4	19.5±1.8	9.1±1.0	9.0±1.6
1.6×	12.3±2.4	15.2±2.3	10.7±0.8	9.3±1.7
1.8×	25.0±3.9	25.2±4.4	9.5±2.1	14.9±0.9

Table 3.7: Mean time delays (t) in ms for trajectory following trials. n =10 for all speeds.

joint. The four components are summed to generate net joint torque. The resulting actuator flexion and extension torques are commanded to the corresponding flexion and extension SEA via the velocity-based control scheme described in section 3.2.3. To avoid cable slack, each SEA applies a minimum of 0.5 Nm torque to the joint at all times.

Figure 3.7 summarizes the results of the motion tracking experiments. Tables 3.6 and 3.7 list the mean cross-correlation coefficients and signal time delays over the trials for all traces. The executed joint position and velocity trajectories closely follow the desired trajectories for trial up to 1.6×, with position and velocity correlation coefficients greater than 0.90 and 0.80, respectively. The largest differences between desired and commanded position occur during periods of knee extension at 1.6×, with a maximum error of 7.3^o. Velocity shows a similar tracking quality. Top speeds achieved 160 rpm in 1.6× trials. Additionally, there is a high degree of repeatability throughout all trials, with a maximum standard deviation of 7.4^o and 44 rpm from the mean values over all trials. (The first leg swing was not included in these calculations, as impulse accelerations from rest are not representative of system dynamics.) Higher joint speeds could not be tested, because the compliance retainer in the knee joint failed during the high speed trials.

The motion experiments validate RNL1's electromechanical and control design, demonstrating that it is possible for a robotic leg to achieve human-like actuation capabilities. Experimental results show that RNL1 can reliably generate human-like leg motions with high positional accuracy for joint speeds up to 160 rpm and with lesser position accuracy for joint speeds up to 190 rpm, approximately 90% of the maximum designed for joint velocity. Furthermore, the presence of joint compliance does not hinder robot performance. While the clamping plates failed, indicating the need to construct them out of a stronger material, the compliant element did not show any wear throughout the experiment. Experiments also reveal that RNL2's SEAs will need to incorporate stiffer springs with a higher torque rating, as RNL1's SEA springs limited actuator bandwidth to 25 ms and imposed a joint torque limit of 12.6 Nm, approximately 30% of the actuation goal at the joint level.

In the long term, the linear springs in the SEA drivetrain could be replaced with a nonlinear stiffening spring. Stiffening nonlinear springs enable high precision zero torque control with high bandwidth responses at large commanded torques. Additionally, nonlinear springs in antagonistic actuators enable joint co-contraction and stiffness modulation [158], important for energy regulation during running [15][14]. Advantages of nonlinear springs to series elastic actuation are widely recognized [97][84]; however a systematic method for developing compact springs with custom torque-deflection profiles has not been proposed. To address this shortcoming, a synthesis method for compact nonlinear springs with user-defined torque-deflection profiles is presented and evaluated in chapter 6 for inclusion in a future hardware revision.

3.3 Robotic Neuromuscular Leg 2 (RNL2)

RNL1 experiments validated that a dynamically scaled, antagonistically actuated robotic leg can achieve human-like performance. Based on these results, RNL2, a full, three-segmented, multi-articular robotic leg is created to test neuromuscular control strategies (Fig. 3.8a). The majority of RNL2's electromechanical design and control is identical to RNL1's electromechan-

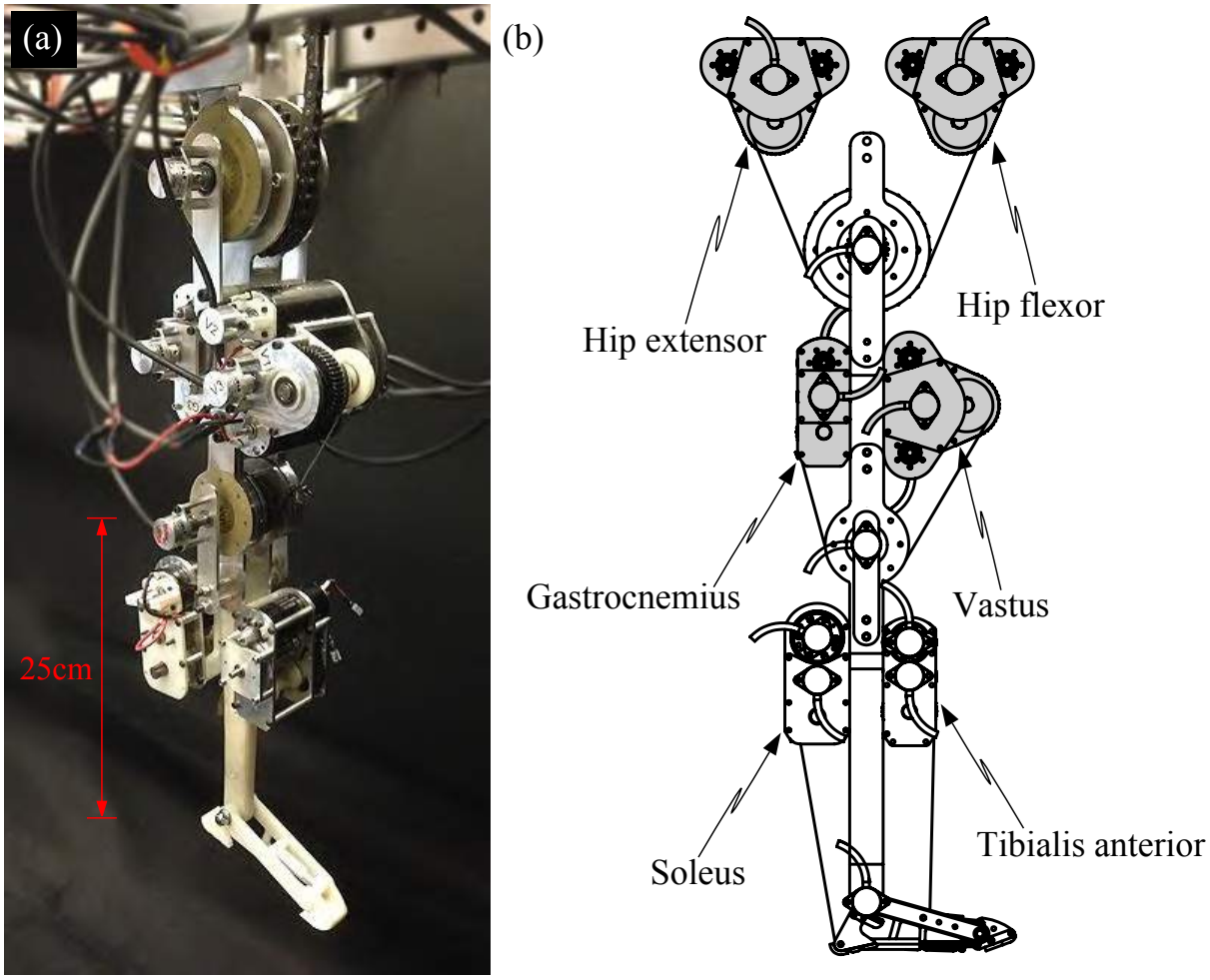

Figure 3.8: The Robotic Neuromuscular Leg 2 (RNL2) test platform. a) Hardware implementation. b) Actuation schematic. Solid: Mono-articular actuator cables. Gray: Actuators used to evaluate ideal swing-leg control in chapter 4.

ical design and control. Design differences implemented in RNL2 are described in the following sections.

3.3.1 RNL2 Series Elastic Actuators

RNL2 is designed to accomodate both both mono- and bi-articular series elastic actuators, emulating the muscle configuration of [35]'s walking model. This will allow the robot to implement neuromuscular locomotion controls. As a step to transferring neuromscular swing-leg controls to RNL2, an idealized swing-leg controller is first evaluated on hardware. (Experiments

	Gastroc.	Hamstring	Hip Ext.	Hip Flex.	Soleus	Tibialis Ant.	Vastus
Stage 1 (Int)	3.6:1	4:1	4:1	4:1	2.5:1	3.6:1	4:1
Stage 2 (Int)	2.9:1	3:1	3:1	3:1	2.8:1	2.9:1	3:1
Stage 3 (Ext)	4.1:1, 4.1:1	5:1, 2.5:1	5:1	5:1	4.1:1	4.1:1	2.5:1
k_s **(Nm/rad)**	23.9	44.5	44.5	48.9	49.3	23.9	49.3

Table 3.8: RNL2 SEA drivetrain stage gear ratios and spring constants

evaluating this swing-leg controller are described in chapter 4.) For this controller, RNL2 uses only mono-articular actuators. The robot's actuator layout for these experiments is shown in figure 3.8b. The hip extensor and hip flexor antagonistically actuate the robot's hip. The vastus and gastrocnemius actuators antagonistically actuate the robot's knee, and act as the robot's knee extensor and knee flexor, respectively.

The hip extensor, hip flexor, and vastus actuators have the same interal drivetrain layout as RNL1's series elastic actuators. The external gear stage, defined by RNL2's hip and knee radii, change these SEAs' stall torques and no load speeds, allowing the same actuator design to meet their respective performance requirements shown in table 3.3. Tibialis anterior actuator requirements are less than gastrocnemius requirements. As such, the two actuators share the same design. These SEAs use a single RE30 motor (RE30: 184858; Maxon Motor AG). The soleus has a unique actuator design. This SEA will be used in future stance and locomotion experiments. It uses one RE40 (RE40: 148877; Maxon Motor AG). Drivetrain gear ratios of all actuators are shown in table 3.8.

Rotary spring couplings again serve as the SEAs' compliant elements, and are located between the actuators' first and second drivetrain stages for compactness. RNL1 revealed that spring coupler torque rating is critical for meeting torque actuation targets. Coupler torque ratings are proportional to the amount of material used in the spring design; couplers with a high torque rating are larger than couplers with a low torque rating. Coupler torque rating is also proportional to stiffness; couplers with a high torque rating are stiffer than couplers with a low torque rating. To meet torque actuation targets, RNL2 SEAs need to incorporate spring couplers with a higher torque rating. To still meet the 0.5 Nm knee torque resolution goal, RNL2 therefore

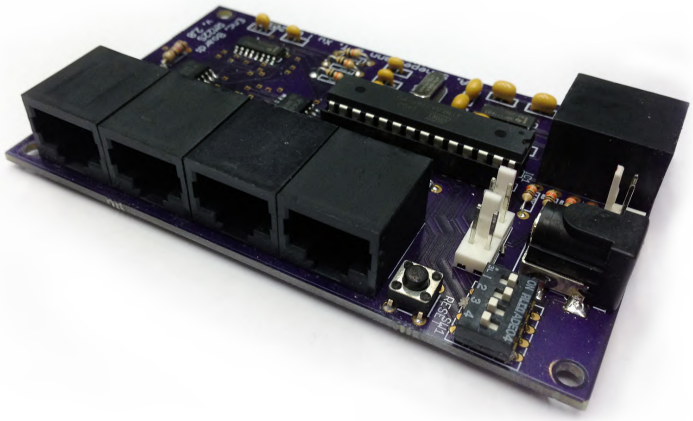

Figure 3.9: RNL2 encoder interface PCB.

needs higher resolution encoders than what can be provided by the data acquisition card used to interface RNL1's encoders to the host PC. Due to author preference, encoder selection is limited to the Renishaw PLC catalog. To ensure compatibility with the existing SEA design, encoder selection is further limited to Renishaw's RM22 line. The highest resolution absolute rotary magnetic encoder in this product family is 13 bit (RM22S: 13 bit, magnetic, digital: Renishaw PLC). For these encoders, no compatible interface options with the robot's host PC exist. A custom PCB is designed to interface the encoders with the host PC (Fig. 3.9). The PCB, based around the ATmega328-PU microcontroller, decodes the RM22S encoder readings and transmits sensor measurements to the host PC via RS-232. The following spring couplers are selected for each actuator: hip extensor, hip flexor, hamstring, soleus, and vastus use a spring coupling with an advertised torque rating of 5.1 Nm (MWS25-6-6-SS; Ruland Manufacturing Co.); gastrocnemius and tibialis anterior use a spring coupling with an advertised torque rating of 2.9 Nm (FSMR19-5-5-A; Ruland Manufacturing Co.). The torsional stiffness of each coupling is calculated from testbed experiments using analog compression load cells (FC22: 100lbf; Measurement Specialties) (Fig. 3.10). Identified spring constants, calculated via linear regression, are shown in table 3.8. These springs provide an approximate joint torque resolution of 1.0 Nm at the robot's hip and 0.5 Nm at the robot's knee.

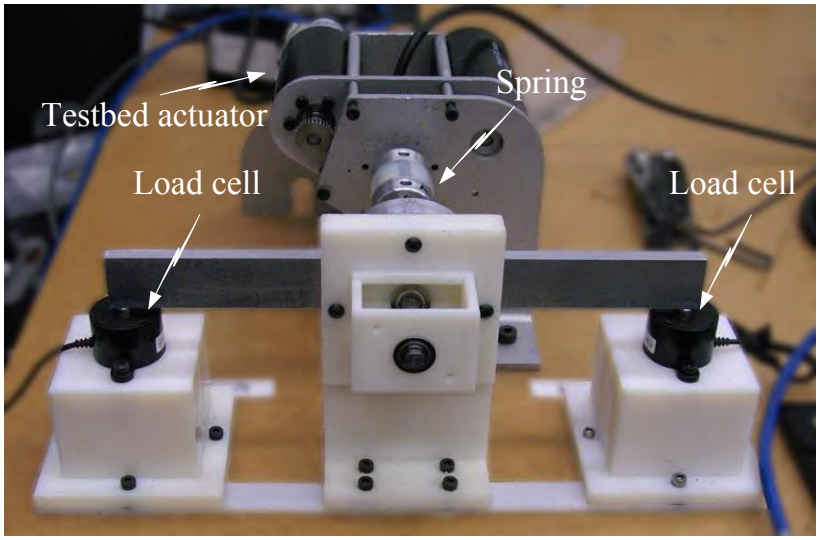

Figure 3.10: Spring characterization testbed.

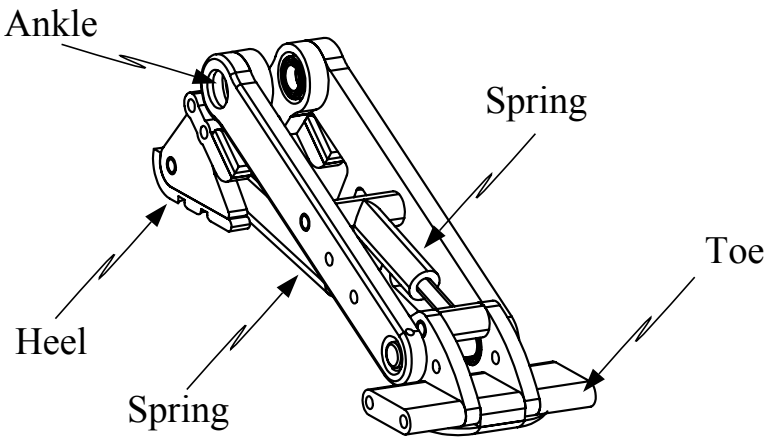

Figure 3.11: RNL2 foot schematic.

The host PC's control software is upgraded to xPC Target (Mathworks, Inc.) to guarantee synchronous 1 kHz control. RNL2's motor controllers (DZEANTU-020B080B; Advanced Motion Controls) communicate with the host PC via EtherCAT (Beckhoff Automation).

3.4 RNL2 Foot

RNL2 has a three-degree of freedom foot with a passive toe and arch (Fig. 3.11). It's design is based on the windlass mechanism found in human feet, which passively stiffens the foot arch

during push-off [47]. Simulation studies using [35]'s neuromuscular model showed that the windlass mechanism leads to a 15% energetic cost reduction during normal walking compared to rigid feet [129]. In RNL2, the windlass mechanism is implemented using two passive linear springs, which connect the robot's toe to and the top of its foot, as well as its toe and heel. The latter spring simulates a human's plantar fascia connective tissue. The foot segment will be used in later hardware experiments of neuromuscular stance behavior and neuromuscular locomotion.

3.5 Summary

This chapter presented the design and evaluation of the Robotic Neuromuscular Leg (RNL) test platforms, designed to evaluate decentralized neuromuscular control strategies of human locomotion on robotic hardware. Motion tracking experiments with RNL1 confirmed that it is possible to construct and control a dynamically scaled robotic leg that matches human size, weight, and actuation capabilities. Based on these results, RNL2, a full, powered segmented robotic leg was constructed. This robot will be used in subsequent chapters to transfer neuromuscular control ideas to robotic hardware, and evaluate the ability of a decentralized, neuromuscular swing-leg controller to accurately place the robot's feet into desired ground targets for undisturbed and disturbed motions.

Chapter 4

Transfer and Evaluation of Decentralized Reactive Swing-Leg Control on a Powered Robotic Leg

Material in this chapter is partially based on:

A. Schepelmann, J. Austin, and H. Geyer
Evaluation of decentralized reactive swing-leg control on a powered robotic leg.
Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems: 381-386, 2015. [5]

Animals and robots balance dynamically by placing their feet into proper ground targets. While foot placement controls exist for both fully robotic systems and powered prostheses, none enable the dynamism and reactiveness of able-bodied humans. A control approach was recently developed for an ideal double pendulum dynamical system that places feet into ground targets for a wide range of initial conditions and in the presence of significant locomotion disturbances [21]. Whereas its performance in simulation make it an attractive candidate to control legged robotic systems, it is unclear if the approach can be used on real-world systems. In this chapter, the approach is transferred and evaluated on the robotic hardware test platform RNL2, described in the previous chapter. To ease transfer of the control approach to robotic hardware, a model-based design approach is adopted and a high-fidelity simulation of the robot is created. The simulation is used as a tool to evaluate if the approach can be applied to robotic systems, how specific components need to be implemented to account for hardware constraints, and to pre-tune

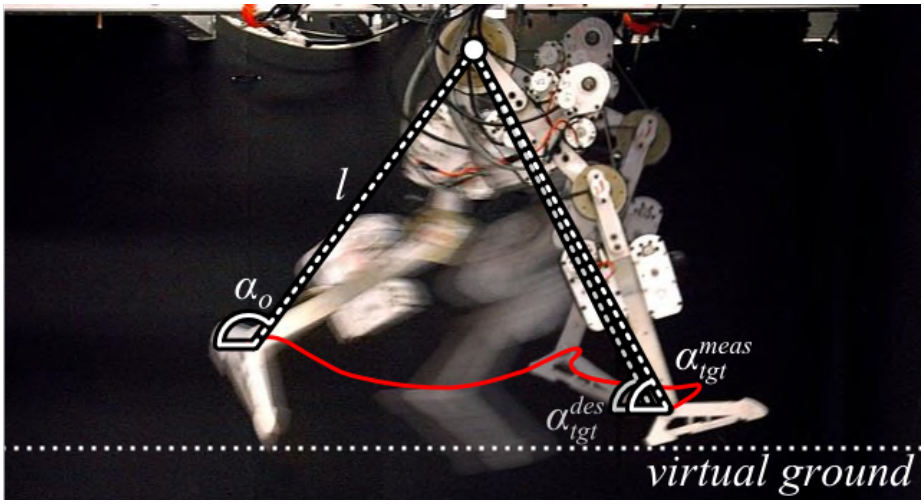

Figure 4.1: Swing-leg control experiments. The controller issues joint torque commands to regulate the length l of a single segment, virtual leg between the robot's hip and ankle, moving it from an initial position α_0 to a target position α_{tgt} when making contact with a virtual ground (dotted). Solid: Trajectory traced out by ankle point during experiment.

mid- and high-level control gains. Hardware results show that the controller can be transferred to robotic systems and achieve foot placements into desired ground targets with comparable accuracy to an ideal double pendulum simulation, both when the robot's motion is undisturbed, as well as when it encounters obstacles in early, mid, and late swing.

4.1 Introduction

Animals and legged robots balance dynamically by placing their feet into proper ground targets [112][107]. Ground targets that stabilize locomotion in the presence of disturbances can be identified with simple locomotion models, like the linear inverted pendulum model for walking [150][63] and the spring-mass model for running [9][123]. (Details of these models are further discussed in section 2.3.) Robotic systems then place their feet into identified ground targets in several ways.

Fully robotic systems pre-plan and execute full-body trajectories using inverse dynamics and kinematics [66][101]. (Details of centralized planning and tracking techniques are further dis-

cussed in section 2.4). While these approaches enable robots to climb stairs [66] and react to push disturbances [102][153], they require estimates of the robot's full state and an accurate system model. Such approaches can therefore not be used to control robotic assistive devices where the human user's state is unknown.

Robotic locomotion controls for human-in-the-loop applications primarily replay motion patterns extracted from healthy human gait, including joint impedance [8][143], joint motion [50][89], or a combination of the two [4]. (State-of-the-art human-in-the-loop control methods are further discussed in section 2.6). Replayed motion patterns contain proper foot placements for level and sloped ground walking at various speeds. Some research has explored extensions to impedance-based controls that could identify stumbles and alter prosthesis motion [76][75], but no conclusive results have been demonstrated.

Heuristic controllers based on simple locomotion models represent alternative control approaches for fully robotic systems and powered prostheses. (Heuristic approaches are further detailed in section 2.5). Commands are not dependent on a system's full state, rather metrics such as system energy [106], leg angle of attack during touchdown [123], and states of virtual muscle actuators [35]. Such controllers allow simple robots to traverse rough terrain equivalent to their leg length [107][24] and have been used in an active ankle-foot prosthesis to regulate joint torque during stance [26].

Recently, a swing-leg control approach based on double pendulum dynamics was proposed in [21]. When applied to an ideal frictionless pendulum simulation, the proposed controller places feet into ground targets for a wide range of initial conditions and in the presence of significant locomotion disturbances. It was later re-formulated to use virtual muscle actuators instead of ideal torque sources [22], and incorporated into a planar, muscle-reflex based locomotion model [133]. Despite lacking central processing, this model walked steadily over level terrain, rough terrain, and up stairs without parameter changes. The control approach's performance in simulation make it an attractive candidate to control both fully robotic systems and powered prostheses, but its functionality on robotic hardware is unknown.

Work in this chapter transfers this swing-leg control approach to robotic hardware (Fig. 4.1), and quantifies its ability to execute foot placements into desired ground targets, both when the leg's motion is undisturbed and when unknown obstacles are in the leg's path. A summary of the swing-leg controller is presented first. Next, a simulation-based approach to transfer the swing-leg controller to robotic hardware is presented. Simulation experiments first validate that the swing-leg controller functions on a dynamically scaled system. A high-fidelity simulation of RNL2 and its control system is then presented. A dynamic calibration method for RNL2's absolute encoders to account for encoder swash is presented next. Finally, simulation and hardware experiment results are presented. Hardware experiments show that the controller can be transferred to robotic systems and achieve foot placements into desired ground targets with comparable accuracy to the ideal double pendulum simulation, both when the robot's motion is undisturbed, as well as when it encounters obstacles in early, mid, and late swing.

4.2 Swing-Leg Controller

The swing-leg controller proposed in [21] is a high level controller based on ideal double pendulum dynamics to achieve foot placements into specified ground targets α_{tgt} via combined hip and knee control. Hip and knee angles ϕ_h and ϕ_k are reinterpreted as a length l and angle α of a single-segment virtual leg (Fig. 4.2a). Assuming equal thigh and shank lengths $l_t = l_s$, $\alpha = \phi_h - \frac{\phi_k}{2}$ and $l = 2l_t \sin(\frac{\phi_k}{2})$.

Three sequential tasks must be accomplished to place the foot during swing: leg flexion to achieve ground clearance, leg advancement to a placement target, and leg extension until ground contact (Fig. 4.2b). While this sequence could be realized by tracking predefined trajectories, heuristic controls are used for three reasons. First, tracking would require extensive predefined motion libraries, which may not handle disturbances. Second, heuristic controllers can exploit passive dynamics to lower joint torques required to realize swing-leg motions. Third, joint controls can be decoupled to modularize the controller for use in multiple prosthesis configurations.

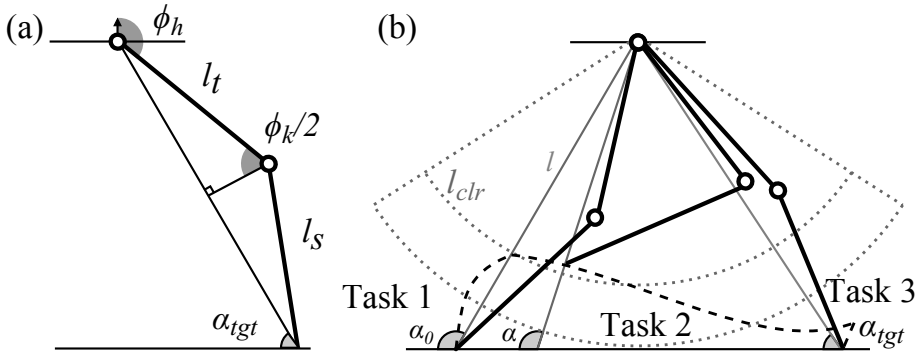

Figure 4.2: Swing-leg control. a) Model geometry. b) Task sequence.

For this reason, the controller is structured as two functionally distinct hip and knee joint controllers.

4.2.1 Hip Control

Hip control is active throughout swing. Its primary function is to move the leg angle into α_{tgt}, given by

$$\tau_h^\alpha = k_p^\alpha(\alpha_{tgt} - \alpha) - k_d^\alpha \dot{\alpha} \tag{4.1}$$

where k_p^α and k_d^α are proportional and derivative gains. During leg extension in late swing, the hip control receives an additional input τ_h^{iii}, detailed in the next section.

4.2.2 Knee Control

The knee control's primary function is to regulate l. Its control is separated into three tasks, shown in figure 4.2a, and can be realized as a finite state machine.

Passive dynamics are exploited to accomplish task one, leg flexion to achieve ground clearance during swing. Hip control (eq. 1) initially generates negative hip accelerations to drive the leg toward α_{tgt}, resulting in passive knee flexion as long as α increases, i.e. $\dot{\alpha} > 0$. If $\dot{\alpha} \leq 0$, there is no passive knee motion; in such cases, active knee flexion, proportional to how fast the

leg moves forward, is applied. The resulting control of task one is

$$\tau_k^i = \begin{cases} k^i \dot{\alpha} & \dot{\alpha} \leq 0 \\ 0 & \dot{\alpha} > 0 \end{cases} \tag{4.2}$$

where k^i is a proportional gain.

Task two, which holds the knee while the hip controller moves the leg toward α_{tgt}, triggers once l becomes less than a predefined clearance length l_{clr}. Knee flexion ($\dot{\phi}_k \leq 0$) is damped using a pure damping input. Knee extension uses a modulated damping input which allows passive extension to occur when α approaches α_{tgt}, but prevents premature landing if the knee extends faster than the overall leg angle. The resulting control of task two is

$$\tau_k^{ii} = \begin{cases} -k^{ii}\dot{\phi}_k & \dot{\phi}_k \leq 0 \\ -k^{ii}\dot{\phi}_k(\alpha - \alpha_{tgt})(\dot{\phi}_k + \dot{\alpha}) & \dot{\phi}_k > 0 \;\&\; \dot{\phi}_k > -\dot{\alpha} \\ 0 & otherwise \end{cases} \tag{4.3}$$

where k^{ii} is a proportional gain.

Control task three stops swing and extends the leg into α_{tgt} when α passes the threshold $\alpha_{thr} = \alpha_{tgt} + \Delta\alpha_{thr}$. This control task, inspired by nonlinear contact models [35][56], generates a stopping knee-flexion torque

$$\tau_k^{iii} = \begin{cases} -k^{stp}(\alpha_{thr} - \alpha)(1 - \frac{\dot{\alpha}}{\dot{\alpha}_{max}}) & \alpha < \alpha_{thr}, \dot{\alpha} < \dot{\alpha}_{max} \\ 0 & otherwise \end{cases} \tag{4.4}$$

where $\dot{\alpha}_{max}$ is a parameter describing the leg's maximum return velocity at which reaction forces are developed. To cancel this torque's effect on the hip motion, a hip torque $\tau_h^{iii} = -\tau_k^{iii}$ is applied during this control task.

4.3 Transfer to Hardware Platform: Simulation

A model-based design approach is used to transfer the proposed swing-leg controller to robotic hardware. The following sections describe simulation tasks used to validate that the idealized swing-leg controller functions on robotic hardware. First, idealized swing-leg controller foot placement behavior and accuracy is tested when the controller is dynamically scaled to RNL2's size. Next, a high-fidelity simulation of the robot is created to investigate the feasibility of transferring neuromuscular controls to robotic hardware and pre-tune hardware gains via optimization. Actuator and joint friction characterization is necessary to create such a simulation. This simulation is used as a benchmark for hardware experiments.

4.3.1 Validate Dynamic Scaling

It is unclear if the ideal swing-leg controller can be dynamically scaled to RNL2's size and still exhibit the same foot placement behavior and accuracy. To test if the controller can be dynamically scaled, the mechanical properties and control gains of [21]'s model are dynamically scaled to RNL2's size using the scaling factors in section 3.1. Undisturbed foot placement experiments, consisting of 2057 simulation experiments, are performed on the dynamically scaled model and compared to results reported in [21].

Over all trials, the average absolute error between the full-sized and dynamically scaled model is 0.27^o/trial, indicating that the controller can be dynamically scaled while still exhibiting the same performance. The discrepancy between the full-sized and dynamically scaled model results from numerical error stemming from implementation differences between [21]'s original and the dynamically scaled code.

	μ_1 (mNm)	μ_2 (mNm)	μ_3 (mNm)	ν_j (mNms/rad)
Hip extensor	2.4	2.6	7.7	154.7 (Hip)
Hip flexor	3.5	1.0	2.9	154.7 (Hip)
Knee extensor	0.4	0.4	1.1	390.0 (Knee)
Knee flexor	1.9	0.2	0.6	390.0 (Knee)

Table 4.1: RNL2 drivetrain and joint friction coefficients for SEAs and joints active during ideal swing-leg control

4.3.2 Actuator and Joint Friction Characterization

To enable a model-based design approach, a high-fidelity simulation that exhibits representative dynamics of the hardware system is necessary. Several differences exist between the ideal swing-leg control model in [21] and RNL2 hardware. First, while the robot's mass distribution is closer to a human leg's than in traditional humanoid robots, it does not perfectly match human mass distributions. Second, the controller and hardware model in [21] ran in continuous time with a variable step solver. Generated controller behavior when its high- and mid-level control runs discretely at 1 kHz is unclear. Third the robot's SEAs introduce actuation time delays. Fourth, friction exists both within RNL2's SEAs, which further affect the ability to apply desired joint torque, and at the robot's joints, which may impede passive dynamics observed in the idealized simulation.

A high-fidelity simulation of the robot will allow us to investigate the effect of robot mass distribution discrepancies and discrete time controller execution. To create such a simulation, it is necessary to characterize SEA spring coupler stiffnesses, as well as drivetrain and joint frictions. Spring coupler stiffnesses are identified using the actuator characterization testbed described in section 3.3.1. Procedures to characterize joint and drivetrain friction experimentally are described below.

RNL2's joints are assumed to have only viscous friction. With this assumption, it is possible to identify joint friction experimentally using measured joint position over time by treating each

segment as a gravity driven damped pendulum

$$l_{COM}\frac{\partial^2\theta}{\partial t^2} + \nu\frac{\partial\theta}{\partial t} + gsin(\theta) = 0. \tag{4.5}$$

where l_{COM} is the segment's center of mass length from the axis of rotation, θ is the measured joint position, ν is the viscous friction coefficient, and g is the acceleration due to gravity. This equation can be rewritten in regressor form as

$$\left[\frac{\partial^2\theta}{\partial t^2}\,\frac{\partial\theta}{\partial t}\right]\left[l/g\nu/g\right]^T = \left[sin(\theta)\right]. \tag{4.6}$$

For joint friction identification experiments, each segment is suspended from its joint and disturbed. Encoders at the robot's hip and knee measure joint position over time. Friction coefficients are then calculated using equation 4.6. Identified viscous friction parameters are shown in table 4.1.

Due to RNL2's SEA design, it is not possible to actuate each SEA drivetrain stage individually. Since no controlled torque can be applied to individual drivetrain stages, RNL2's SEAs are assumed to only have Coulomb friction. For these characterization experiments, drivetrain stages are disassembled so that each of the actuator's shafts can be accelerated by hand. After applying an acceleration, the shafts are let go, at which point encoders at the stage of each drivetrain measure position over time. A second order polynomial is then applied to data for each spin, enabling a clean estimate of drivetrain velocity and acceleration over time. With these data, it is possible to calculate Coulomb friction coefficients for each drivetrain stage via linear regression using

$$\left[sgn(\dot{\theta})\left|\dot{\theta}\right|\left|\dot{\theta}\right|\right]\left[\mu_i\right]^T = \left[-J\ddot{\theta}\right]. \tag{4.7}$$

where μ_i is the coefficient of Coulomb friction and -J is the drivetrain stage's rotational inertia estimated from CAD. Identified Coulomb friction parameters of each SEA's drivetrain

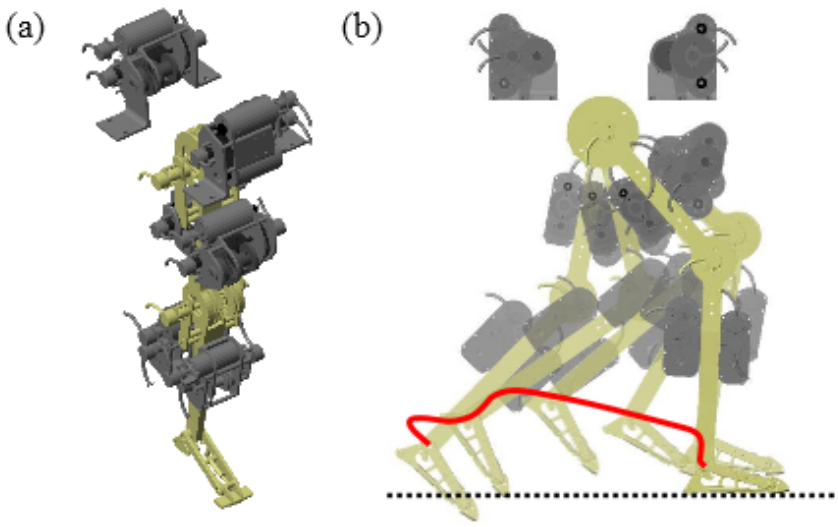

Figure 4.3: RNL2 simulation. a) Screenshot of RNL2 model. b) Undisturbed trajectory experiment motion, α_{tgt}=70^o. Solid: Traced ankle point trajectory. Dotted: Virtual ground.

stages used in ideal swing-leg experiments are shown in table 4.1.

4.3.3 High-Fidelity Robot Simulation

A simulation of RNL2 is developed in Simulink SimMechanics (SimMechanics: Second Generation; Mathworks, Inc.) (Fig. 4.3), which models the robot at the individual component level. It is used to transfer the proposed heuristic swing-leg control approach to hardware, serving as a tool to evaluate if it can be applied to robotic systems, and how specific control components need to be implemented to account for hardware constraints. The simulation's software and control architecture is shown in figure 4.4. The implementation of each control level is outlined in the following sections. To capture behavior imposed by the proposed swing-leg controller's discrete-time execution when controlling robotic hardware, simulated high- and mid-level control loops are constrained to run at 1 kHz.

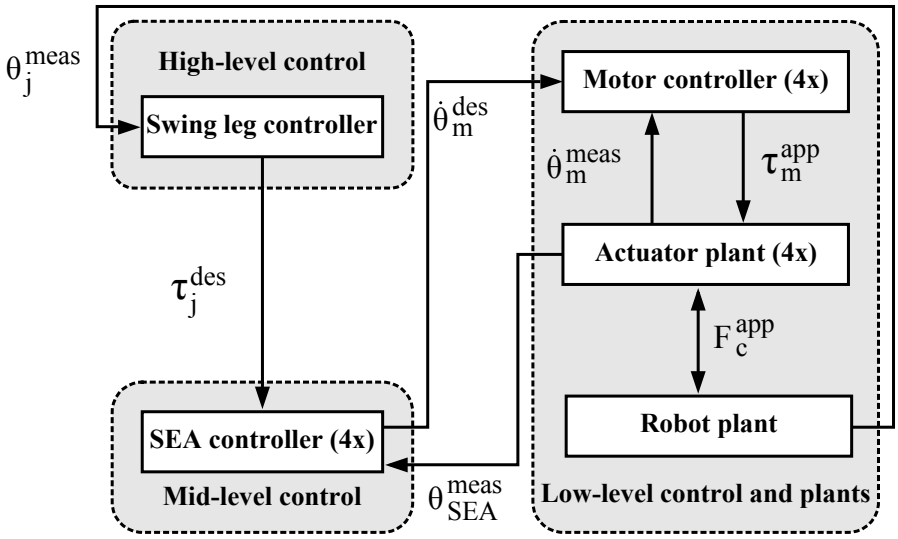

Figure 4.4: RNL2 software and control architecture. θ_j^{meas}: measured joint angles. τ_j^{des}: desired joint torques. $\dot{\theta}_m^{des}$: desired motor velocities. $\dot{\theta}_m^{meas}$: measured motor velocities. τ_m^{app}: applied motor torques. F_c^{app}: force transmitted by cable drive. θ_{SEA}^{meas}: measured spring angles.

4.3.4 High-Level Control

The high-level control block contains the swing-leg control approach described in section 4.2 and generates desired net joint torques τ_j^{des} based on α and $\dot{\alpha}$ calculated from joint measurements of the robot's hip and knee. Since RNL2's cable-driven SEAs can only pull on a joint, generated high-level torques are distributed to the extensor and flexor actuators based on sign, with positive torques representing extension torques. To compensate for instantaneous torque changes commanded by the swing-leg controller, which could result in system instability, commanded torques are low-pass filtered (f_c=25 Hz).

4.3.5 Mid-Level Control

Desired actuator torques are fed into the mid-level control block that regulates SEA torques. This block generates desired motor velocity signals $\dot{\theta}_m^{des}$ using the velocity-based control scheme described in section 3.2.3. Desired motor velocities are sent to the low-level control block that contains motor controller simulations and hardware models.

4.3.6 Low-Level Control and Hardware Plant

The low-level control block contains a simulation of RNL2's motor controllers, as well as plant models of physical hardware. Due to the motor controllers' 10 kHz operating frequency, their dynamics, as well as the rest of the hardware, are modeled as continuous time systems.

Motor electrical dynamics are modeled as

$$V - iR - L\frac{di}{dt} - k_{EMF}\dot{\theta}_m = 0 \tag{4.8}$$

where V, i, R, L, k_{EMF}, and $\dot{\theta}_m$ are the motor voltage, current, resistance, inductance, back-EMF constant, and motor velocity [128]. A $\pm$ 48 V saturation is applied to the model to simulate voltage limits of RNL2's motor controllers. The motor controller is abstracted as a PID loop which generates voltage commands. The PID loop's error signal is the difference between desired and measured motor velocity. Dissipative no-load currents are subtracted for each motor before calculating transmitted torque.

Drivetrain stages are modeled in SimMechanics as separate physical, interacting bodies. Rotational inetias of each drivetrain stage are estimated from CAD models (Solidworks 2012; DSS Corp.). Coulomb friction τ_f^c is applied to each bearing stage using the equation $\tau_f^c = \mu_i sgn(\dot{\theta_m})$, where μ_i is the bearing stage's experimentally identified Coulomb friction coefficient described in the previous section. Viscous friction is applied to the robot's hip and knee joints, using identified viscous friction coefficients described in the previous section. RNL2's modeled SEAs use identified torsional spring stiffnesses listed in section 3.3.1. RNL2's cable drives, which connect the robot's SEA drivetrain outputs to its joints, are modeled as series spring dampers with a stiffness k_c=10,000 N/m and damping v_c=500 Ns/m.

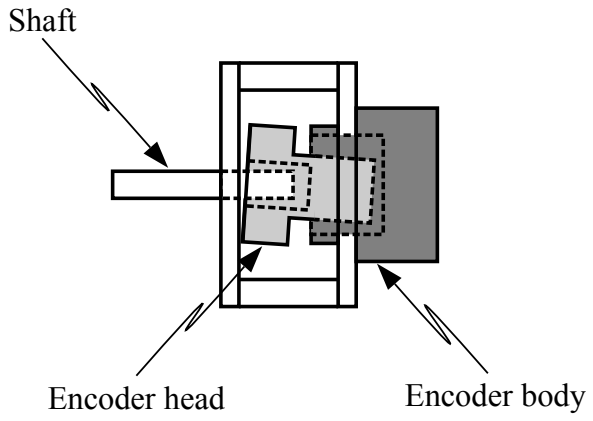

Figure 4.5: Encoder swash illustration. The encoder head's axis of rotation is not concentric with the encoder body, and results in the encoder head rotating about an elliptical instead of circular orbit.

4.3.7 Tuning

All simulation gains are tuned using optimization based on the covariance adaptation evolution strategy (CMA-ES) [43], with a cost function to minimize the sum squared difference between the desired and measured signals at each control level. Gains are tuned hierarchically starting at the lowest level, ensuring that the motor controllers can follow commanded velocity signals. Mid-level SEA gains are then tuned to ensure that the actuators can realize commanded torques. Finally, high-level swing-leg gains are optimized to minimize the difference between the desired and measured α_{tgt}. To mimic conditions used to tune [21]'s controller, all placement gains are tuned for α_{tgt}=70^o. This gain set is used for all experiments.

Hardware gains are also tuned hierarchically. All hardware gains are manually tuned, with high-level gains using the optimized simulations gains as a starting point.

4.4 Hardware Nonidealities: Dynamic Encoder Calibration

As discussed in section 3.3.1, high resolution encoders are required to meet the desired 0.5 Nm torque resolution goal at the robot's knee when using stiffer spring couplers. Due to manufacturing limits and mounting tolerances of the magnetic encoder head, the RM22S encoders

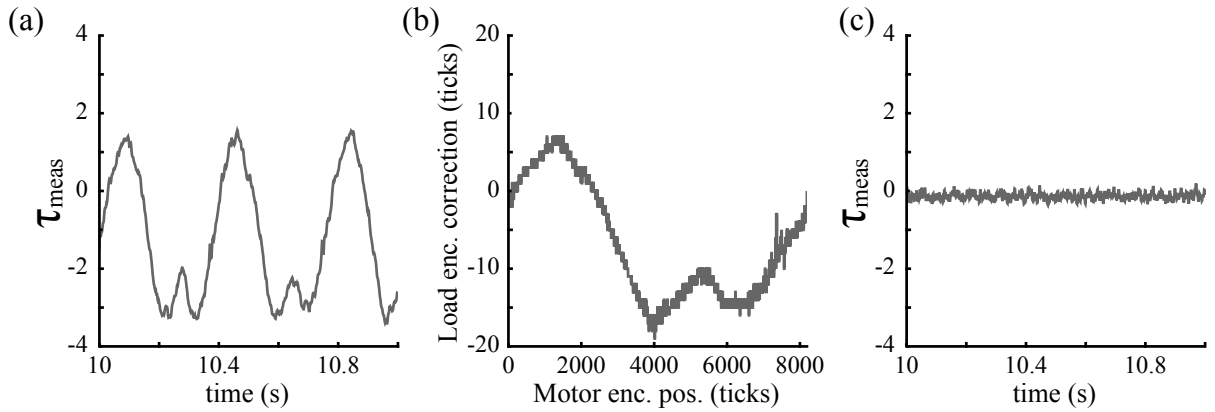

Figure 4.6: Experimentally observed encoder swash. Gastrocnemius torque measurements are used as an example. a) Erroneous torque measurements resulting from encoder swash. Swash is indicated by periodic torque error. b) Lookup table generated via error mapping. c) Corrected torque measurements.

exhibit swash [96]. Swash occurs when the encoder head's axis of rotation is not concentric with the encoder body (Fig. 4.5), and results in the encoder head rotating about an elliptical instead of circular orbit. Due to mounting and manufacturing tolerances, some amount of swash will always be present in the reading.

In several cases, swash can be ignored with minimal impact to torque following performance. If the spring coupler is soft, the encoder resolution is high, and the desired minimal torque resolution is less than the error introduced by the encoder head eccentricities, swash can often be ignored, as significant spring deflections are necessary to realize the minimum desired torque resolution. Similarly, if the encoder resolution is less than the amount of error introduced by swash, the eccentricities will not show up in the encoder reading.

RNL2's spring coupler stiffness and encoder resolution results in each encoder tick corresponding to roughly 0.5 Nm at the knee. Experimentally observed swash between the two encoders results in an eccentricity up to 8 Nm (Fig. 4.6a). This results in poor torque control. Since the RM22S encoders that measure spring deflection are absolute encoders, it is possible to account for encoder eccentricities by calibrating the relative position of two encoders via error mapping.

The applied error mapping technique makes the two encoders follow the same elliptical or-

bit so that the relative position differences reported by the sensors provides an accurate torque estimate. In this approach, one encoder serves as ground truth, and acts as a lookup table of corrections to positions measured by the other encoder. The error mapping process can be broken into two stages: a calibration step and a correction step.

Encoder calibration is performed offline. SEAs are disconnected from RNL2's joints. A 500 rpm, 1/4 Hz square wave is commanded to each SEA for 30 s. Periods of constant velocity should produce no drivetrain torque; therefore, there should be no difference between the positions reported by the two encoders. The two absolute positions reported by the encoders are used to create a lookup table that applies offsets to the position reported by the load side encoder in order to generate a measurement of zero torque when no load is applied. Multiple encoder rotations ensure that an offset is calculated for every encoder tick, despite their asynchronous interface with the host PC. Offsets which were not measured for encoder ticks during the calibration step are estimated via linear interpolation. Example lookup table values are shown in figure 4.6b.

Using the created lookup table, encoder correction is performed in real-time during robot operation. Lookup table values are applied to the load encoder readings based on reported motor position, effectively eliminating false torque readings due to encoder swash (Fig. 4.6c). Artifacts resulting from torque corrections are eliminated with a low-pass filter before the measured torque is used as feedback for the SEA controller.

4.5 Swing-Leg Control Evaluation on RNL2

4.5.1 Experiments

A simulation of the swing-leg controller applied to an ideal double pendulum with human-sized segment mass, length, and inertia [21] is used as a baseline to evaluate behavior generated by the swing-leg controller running on RNL2 in simulation and hardware. In these experiments, RNL2 is suspended from its hip in a rigid mounting cage. While this setup eliminates dynamic

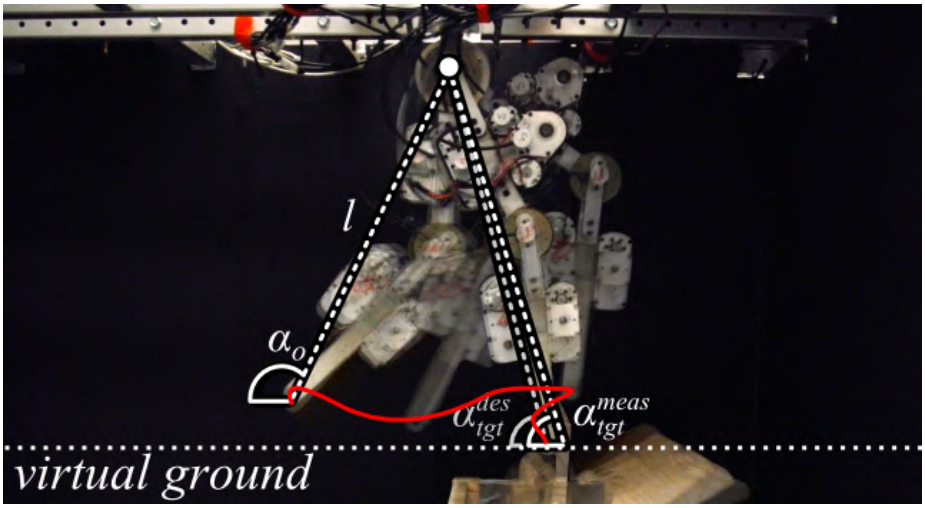

Figure 4.7: Example disturbed swing-leg control experiment. Shown: α_{tgt}=70^o, late disturbance.

effects due to trunk motion, previous simulation work showed that trunk dynamics do not negatively impact controller performance [133]. For control purposes, RNL2 is assumed to have equal, nominal shank and thigh lengths l_t=l_s=27 cm, with l_{clr}=2 cm and $\Delta\alpha_{thr}$=10^o. Equivalent human-sized double pendulum simulation control variables are l_t=l_s=54 cm and l_{clr}=4 cm.

During experiments, the robot starts at a neutral position (Fig. 3.8). A feed-forward torque that is constant across all experiments moves the leg into an initial pose, which also imparts initial joint velocities. After 250 ms, swing-leg control initializes, measuring l, α_0, and the virtual ground location defined by the robot's initial pose (Fig. 4.1). The controller then executes. Motion continues until the foot point again makes contact with the virtual ground location.

Two sets of experiments are used to evaluate the controller in simulation and hardware. Undisturbed motion experiments (Fig. 4.1) test the controller's ability to place feet into desired ground targets for unimpeded swing. Disturbed motion experiments simulate tripping and test the controller's ability to place feet into desired ground targets when the robot encounters an unexpected obstacle in early, mid, and late swing (Figs. 4.7, 4.8). In hardware experiments, the obstacle is a 600 g block on a set of rockers, approximating an impulse disturbance as the robot knocks the obstacle over during swing. In simulation, obstacle collision is modeled using a horizontal force generated by a nonlinear contact model [38], whose parameters are estimated from

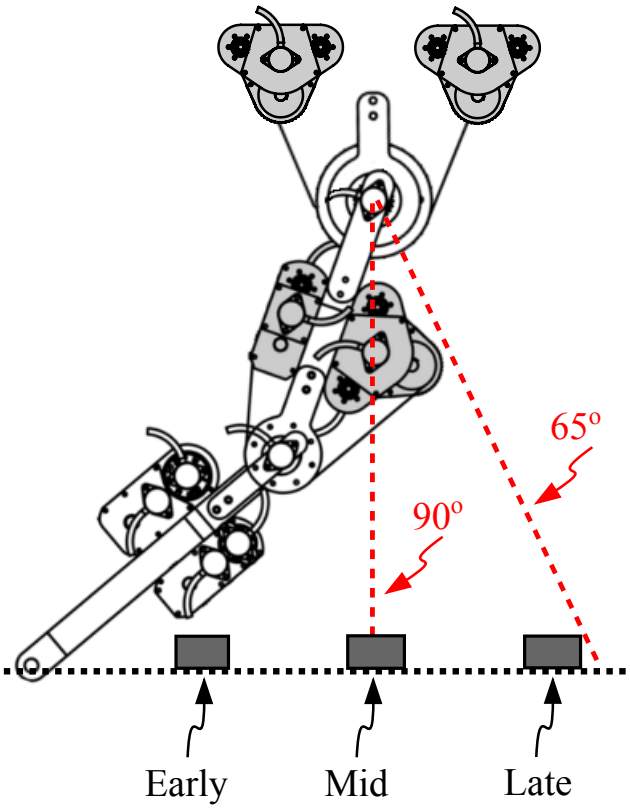

Figure 4.8: Schematic of disturbance experiment setup. Gray: Obstacle locations. Dotted: Virtual ground location as defined by ankle point of robot. Dashed: α_{tgt} range.

the material properties of the leg and obstacle. For disturbed motion experiments, the robot's foot is removed so that disturbances occur at the ankle, better emulating the disturbance condition experienced by the ideal double pendulum simulation. The foot's contribution to total shank mass and inertia is considered negligible. Trip obstacles are positioned as shown in Fig. 4.9. Experiments are conducted at 5^o α_{tgt} increments for undisturbed and all obstacle placement conditions, with 5 hardware trials for each condition. Data from each obstacle position-desired target angle pair are grouped as early, mid, and late motion disturbances depending on whether the occur in the first, middle, or last third of swing. Mean simulation and hardware foot placement errors and swing times for ground target experiments between α_{tgt}=65^o to 85^o, corresponding to various step lengths during walking, are shown in table 4.2 and table 4.3, respectively.

Disturbance	None	Early	Mid	Late
Ideal Sim.	-1.5±0.6	-3.8±0.3	-4.3±0.6	-5.9±0.9
RNL2 Sim.	1.2±0.7	1.2±1.6	1.4±1.4	1.3±1.1
RNL2 Hrdw.	1.2±3.7	-2.7±4.3	-1.5±3.7	-3.5±3.4

Table 4.2: Mean placement error (^{o}) for α_{tgt} range: 65^o to 85^o

Disturbance	None	Early	Mid	Late
Ideal Sim.	394±5	529±3	593±6	516±86
RNL2 Sim.	328±9	366±14	356±25	334±13
RNL2 Hrdw.	481±102	640±103	583±129	546±70

Table 4.3: Mean swing time in ms for α_{tgt} range: 65^o to 85^o

4.5.2 Discussion

In both simulation and hardware, RNL2 places feet with comparable accuracy to the ideal double pendulum for all tested conditions (Tab. 4.2), suggesting that the controller can accurately regulate foot placement of robotic legs. Hardware mean placement error either improves or is within the standard deviation of the ideal double pendulum simulation for all tested conditions.

Foot point trajectories of the double pendulum simulation, RNL2 simulation, and hardware experiments for α_{tgt}=70^o experiments are shown in Fig. 4.9. Disturbed hardware trajectories suggest that the controller makes the robot execute a human-like foot elevation strategy when encountering obstacles in early swing, indicated by retraction of the foot point after it collides with the obstacle [30]. Whether the controller also causes execution of a lowering strategy for late swing obstacles is unclear. Though the step length for late obstacle encounters is shorter than the undisturbed case, characteristic of lowering strategy behavior, the step length for other disturbed swing motions are shorter than the undisturbed case as well.

While the magnitude of normalized step length is the same for all experiments, foot point height during swing, especially early swing, is less pronounced in both the RNL2 simulation and hardware compared to the ideal double pendulum simulation. Comparing commanded joint torques between these systems (Fig. 4.10) reveals that both the characteristic shape and relative magnitude of the commanded torque are different between the ideal double pendulum and RNL2

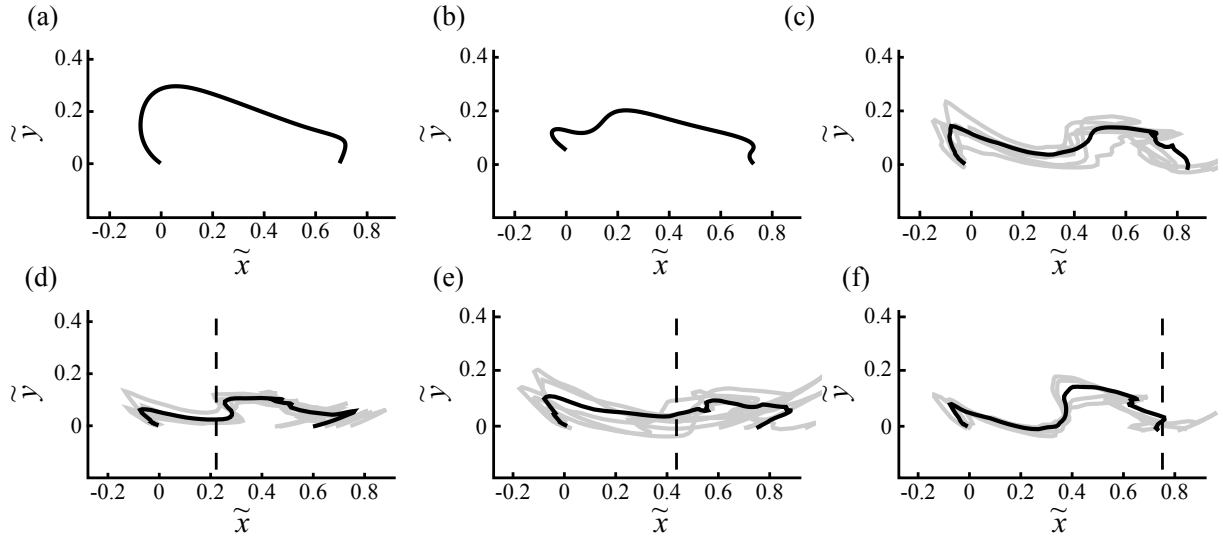

Figure 4.9: Foot point trajectories of α_{tgt}=70^o experiments normalized by respective total leg length ($\tilde{x}, \tilde{y}$). Disturbance type noted in parentheses. Black: Mean trajectories. Gray: Individual trials. Dashed: Obstacle location. a) Ideal double pendulum (none) b) RNL2 sim. (none) c) RNL2 hrdw. (none) d) RNL2 hrdw. (early) e) RNL2 hrdw. (mid) f) RNL2 hrdw. (late)

systems, which plausibly results in the less pronounced ground clearance. This behavior is likely the result of the cost function used to tune the RNL2 simulation, whose gains served as a tuning starting point of the hardware gains. Whereas the swing-leg controller gains for the ideal double pendulum simulation were hand-tuned [21], the cost function used to tune the RNL2 gains did not include an explicit term to consider the overall cosmesis of the motion.

Hardware swing duration exceeds dynamically scaled goals. Based on RNL2's dynamic scaling, swing should be approximately 30% faster than the ideal double pendulum's human-sized motion. This discrepancy can again be attributed to the lack of explicit consideration for factors besides overall placement accuracy when tuning the gains. These results suggest the need for additional cost terms when tuning robot control gains that trade off between placement accuracy, human-like motion, and swing execution time.

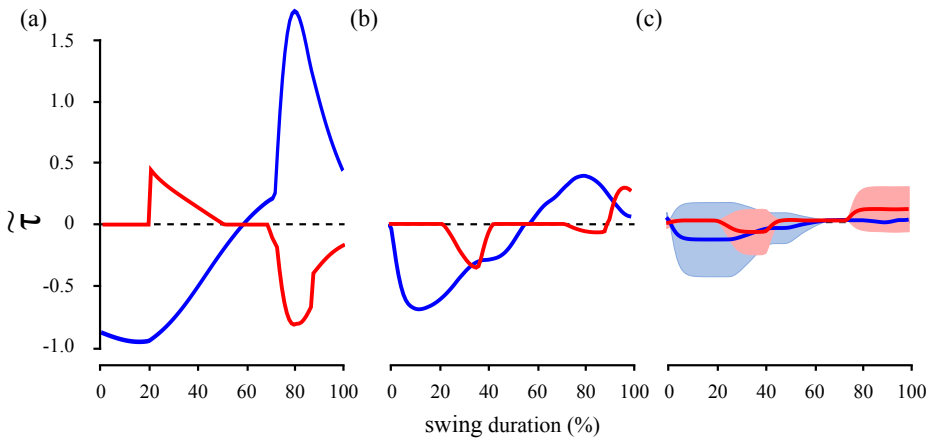

Figure 4.10: Commanded torques for undisturbed α_{tgt}=70^o experiments normalized by product of respective total leg mass, length, and gravity ($\tilde{\tau}$). Blue: Hip. Red: Knee. a) Ideal double pendulum. b) RNL2 sim. c) RNL2 hrdw. Light blue: S.D. of mean hip torque. Pink: S.D. of mean knee torque.

4.6 Summary

This chapter presented work to transfer a swing-leg controller with ideal torque sources at its hip and knee to robotic hardware, and quantified the controller's ability to execute foot placements into desired ground targets, both when the robot's motion was undisturbed, as well as when unknown obstacles were placed in the leg's path during swing. Experimental results show that the controller can be transferred to robotic hardware and achieve comparable foot placement accuracies to the controller running on an ideal, anthropomorphic simulated double pendulum. This performance suggests that the controller could be a potential alternative control method for both humanoid robots and powered prosthetic devices, which require the use of decentralized control schemes. While the swing-leg controller's performance is promising, it is unknown how its performance compares to state-of-the-art locomotion controls. To address this issue, impedance control is implemented on the simulated and hardware RNL2 platforms in the next chapter. The next chapter additionally implements a neuromuscular reinterpretation of the idealized swing-leg controls on robotic hardware in order to investigate potential performance benefits yielded by neuromuscular-inspired foot placement control schemes.

Chapter 5

Neuromuscular Swing-Leg Control & Benchmarking

Reactive swing-leg placement is vital to the stability of animals and legged robots. Work in the previous chapter showed that decentralized, reactive swing-leg control based on the dynamics of an idealized, anthropomorphic double pendulum enables legged robotic systems to robustly place feet into desired ground locations, both when swing is unimpeded, as well as when the robot encounters unknown obstacles. While these experiments suggest that the controller is suitable for controlling powered, segment legs, several extensions to this work exist. First, the idealized swing-leg controller neither considers actuation or control limitations of biological legged systems, nor the presence of a foot segment found in biological legs. Second, it is unknown how the performance of the idealized swing-leg controller compares to state-of-the-art control methods used in powered prosthetic devices.

To account for biological limitations, the idealized swing-leg controller presented in [21] was later reinterpreted in a neuromuscular framework [22], and was applied to an anthropomorphic three-segment leg in simulation. Decentralized hip and knee controls were reformulated using inter-muscle reflexes, which were used to stimulate virtual Hill-type muscles that actuate the leg. Comparisons to human muscle activations, joint torque traces, and ankle point trajectories showed that the neuromuscular swing-leg controller produced similar behavior to that observed in humans.

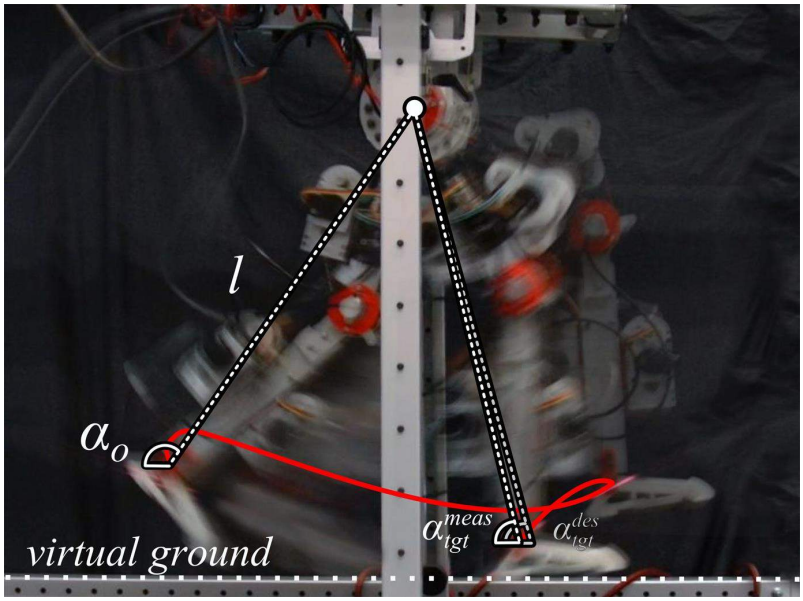

Figure 5.1: Neuromuscular swing-leg control experiments. The controller issues actuator commands based on the output of virtual muscle models to regulate foot placement into a target position α_{tgt} when making contact with a virtual ground (dotted). Solid: Trajectory traced out by ankle point during experiment.

These results suggest that neuromuscular control may generate more human-like behavior than the idealized swing-leg controller when used on robotic hardware. However, due to the increased model and actuation complexity, it is unknown if this transfer is possible. Similarly, it is unknown whether this control method provides any performance advantages compared to state-of-the-art locomotion controls. Work in this chapter addresses these topics.

A summary of the neuromuscular swing-leg controller is presented first, followed by a brief summary of the impedance control scheme used to benchmark the swing-leg controller against state-of-the-art locomotion controls. To realize neuromuscular control on robotic hardware, it is necessary to modify the RNL testbed used in the previous chapter. These modifications are discussed in the following section, along with implementation details necessary for transferring neuromuscular swing-leg control to discrete-time hardware systems. Simulation and hardware experiments that evaluate the performance of the presented swing-leg controllers are presented next, followed by a discussion of advantages of each control method. Experiments show that the neuromuscular controller can be transferred to robotic hardware, and achieve similar placement

accuracy to the idealized swing-leg control and impedance controls tuned for specific locomotion conditions. Results suggest that both the idealized and neuromuscular swing-leg controller possess an implementation advantage over impedance controls, as a single set of gains is applicable to a wide range of desired ground targets. This scalability makes decentralized controls an attractive candidate to regulate locomotion of powered legged systems in real-world applications. Experiments also show that the ankle trajectory for undisturbed swing traced out by the robot under neuromuscular control is more human-like than the trajectories traced out under the proposed idealized control and impedance control.

5.1 Neuromuscular Swing-Leg Control

For completeness, the neuromuscular swing-leg controller originally presented in [22] is summarized below, which reinterprets idealized swing-leg control in a neuromuscular framework. Like its idealized counterpart [21], neuromuscular swing-leg control utilizes distinct hip and knee controls, whose stages are transitioned between using a state machine. Unlike idealized control, which generates actuation commands for ideal torque sources located at the hip and knee joint and relies on explicit knowledge of joint angles, neuromuscular control generates muscle stimulations that cause nine virtual muscles that span between joints to contract and reflexively interact with each other based on inferred joint positions and velocities from internal muscle lengths and contraction speeds. A summary of the controller is presented below.

5.1.1 Muscle Mechanics & Controller Feedback

The neuromuscular controller is designed to actuate a planar, three segment model with anthropomorphic mass and inertia properties [171]. Revolute joints, corresponding to the hip, knee, and ankle, connect these segments. Segments are actuated by nine Hill-type muscles [48] arranged in antagonistic pairs (Fig. 5.2). Of these muscles, the hamstring, rectus femoris, and

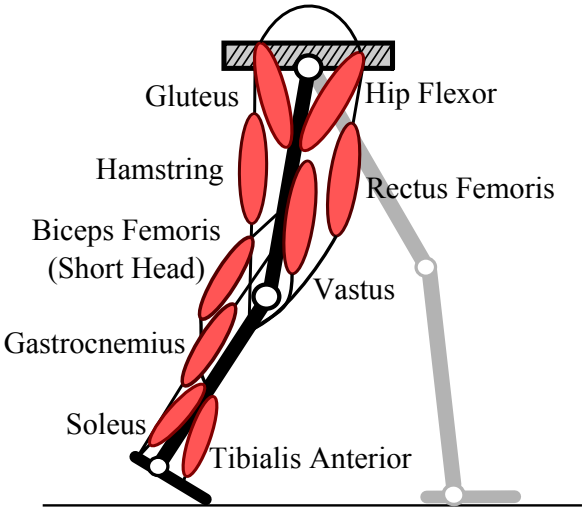

Figure 5.2: Neuromuscular swing-leg muscle diagram. Dashed gray: Trunk segment.

gastrocnemius are bi-articular, spanning the hip-knee, hip-knee, and knee-ankle joints, respectively. Mathematically, the muscles are composed of parallel and series elasticities, as well as a contractile element [35] (Fig. 5.3) and generate contraction forces F^m

$$F^m = F^m_{max} f_l f_v A^m \tag{5.1}$$

where F^m_{max} is the muscles' maximum isometric forces, f_l and f_v are the force-length and force-velocity relationships of the contractile elements, and A^m are the muscle activations. Muscle activations are generated by stimulation signals S^m commanded by muscle reflexes with the equation

$$\dot{A^m} = (S^m - A^m)/\tau_{ecc} \tag{5.2}$$

where τ_{ecc} is the excitation-contraction coupling constant of the muscle and $S^m \in [0, 1]$ to remain consistent with biology that neuron output cannot be negative.

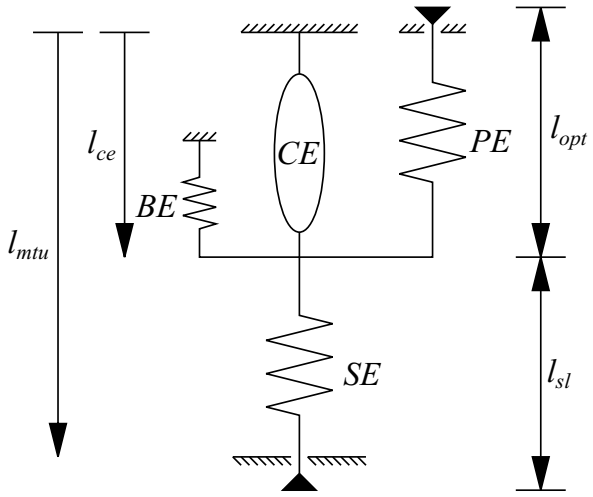

Figure 5.3: Muscle-tendon model. Adapted from [35]. The muscle-tendon unit (MTU) is comprised of a force-generating contractile element (CE) and a series element (SE). If the contractile element stretches beyond its optimum length ($l_{ce} > l_{opt}$, a parallel elasticity (PE) engages, preventing hyper extension. Similarly, a buffer element (BE) engages when the series element goes slack ($l_{mtu} - l_{ce} < l_{sl}$), which prevents the contractile element from collapsing.

Muscle stimulations are generated by local proprioceptive feedbacks of other muscles in the model and take the general form

$$S^m(t) = S_0^m + \sum_n G_n^m P_n^m (t - \Delta t_n^m) \tag{5.3}$$

where n is the index of the muscle contributing to the specific feedback reflex for muscle m, G_n^m is the feedback gain, P_n^m is the proprioceptive signal, and Δt_n^m is the neural transport delay [171]. Specific reflexes for each muscle are described in the next section. Proprioceptive signals are either length L_n^m or velocity V_n^m signals of a specific muscle n, and are modeled as

$$L_n^m = l_{ce}^n - l_{off}^n \tag{5.4}$$

and

$$V_n^m = v_{ce}^n - v_{off}^n \tag{5.5}$$

where l^n_{ce} and v^n_{ce} are the muscle's contractile element length and velocity, respectively, and l^n_{off} and v^n_{off} are offsets. In biology, these offsets are modulated by γ-motoneurons at muscle spindles, which are sensory organs that measure muscle length and velocity [55].

5.1.2 Estimating Leg Angle with Bi-Articular Muscles

The idealized controller described in [21] modulates foot placements into desired ground targets. In this controller, ground targets were expressed as α_{tgt}, the desired leg angle at touchdown. Control inputs for the hip and knee depended on the difference between the current angle α, which could be directly calculated using joint angles, and α_{tgt}. In the neuromuscular controller, this quantity can be estimated using the lengths of the bi-articular muscles that span the hip and knee joints.

Based on the hip and knee joint positions ϕ_h and ϕ_k, the muscle length for the rectus femoris and hamstring can respectively be calculated for any pose as

$$l^m = l^m_0 + r^m_h(\phi_h - \phi^m_{h,r}) - r^m_k(\phi_k - \phi^m_{k,r}) \tag{5.6}$$

and

$$l^m = l^m_0 - r^m_h(\phi_h - \phi^m_{h,r}) - r^m_k(\phi_k - \phi^m_{k,r}) \tag{5.7}$$

where l^m_0 is the muscle rest length reached when the knee and hip are at $\phi^m_{h,r}$ and $\phi^m_{k,r}$ and r^m_h and r^m_k are the muscle moment arms at the hip and knee. Assuming that $r^m_k = r^m_h/2$ and that $\alpha = \phi_h - \phi_k/2$ as in the idealized swing-leg controller, equations 5.6 and 5.7 simplify to

$$l^m = l^m_0 + r^m_h(\alpha - \alpha^m_r) \tag{5.8}$$

and

$$l^m = l_0^m - r_h^m(\alpha - \alpha_r^m) \tag{5.9}$$

where $\alpha_r^m = \phi_{h,r}^m - \phi_{k,r}^m/2$. The muscle length l^m can also be expressed in terms of Hill-type muscle parameters as

$$l^m = l_{ce}^m + l_{se}^m \tag{5.10}$$

where l_{ce}^m is the muscle's contractile element length and l_{se}^m is the muscle's series element length. Due to comparatively low forces required for leg swing, it is assumed that the muscle does not stretch substantially beyond its slack length l_{sl}^m, leading to the approximation that $l_{se}^m = l_{sl}^m$. With this assumption, combining equation 5.10 with equations 5.8 and 5.9, respectively and solving for α yields

$$\alpha = (l_{ce}^m + l_{sl}^m - l_0^m)/r_h^m + \alpha_r^m \tag{5.11}$$

and

$$\alpha = -(l_{ce}^m + l_{sl}^m - l_0^m)/r_h^m + \alpha_r^m. \tag{5.12}$$

The difference between α and α_{tgt} can then be written as

$$\alpha - \alpha_{tgt} = (l_{ce}^m - l_{off}^m)/r_h^m \tag{5.13}$$

where $l_{off}^m = r_h^m(\alpha_{tgt} - \alpha_r^m) - l_{sl}^m + l_0^m$ for rectus femoris and $l_{off}^m = -r_h^m(\alpha_{tgt} - \alpha_r^m) - l_{sl}^m + l_0^m$ for hamstring.

Similarly, it is possible to estimate leg angular velocity from the muscles' spindle velocity feedback as

$$\dot{\alpha} = (v_{ce}^m - v_{off}^m)/r_h^m \tag{5.14}$$

and

$$\dot{\alpha} = -(v_{ce}^m - v_{off}^m)/r_h^m \tag{5.15}$$

for rectus femoris and hamstring, respectively, where $v_{off}^m = 0$.

5.1.3 Neuromuscular Control - Hip

Hip joint control is responsible for driving the leg towards a desired target angle. In idealized swing-control, this was realized as a PD controller that operated on the difference between the current leg angle α and the target angle α_{tgt}. In the neuromuscular reinterpretation, this control is realized by stimulating the gluteus hip-extensor muscle $S^{GLU}(t)$ and the hip flexor muscle $S^{HFL}(t)$ with bi-articular length feedbacks of their agonists as described in section 5.1.2. Specifically, these stimulations take the form

$$S^{HFL}(t) = S_0^{HFL} + G_{RF}^{HFL} L_{RF}^{HFL}(t - \Delta t_{RF}^{HFL}) \tag{5.16}$$

$$S^{GLU}(t) = S_0^{GLU} + G_{HAM}^{GLU} L_{HAM}^{GLU}(t - \Delta t_{HAM}^{GLU}) \tag{5.17}$$

for the hip flexor and gluteus muscle, respectively. It is unnecessary to implement an explicit damping term, as was necessary in idealized control, since muscle behavior is automatically damped due to its force-velocity relationship f_v.

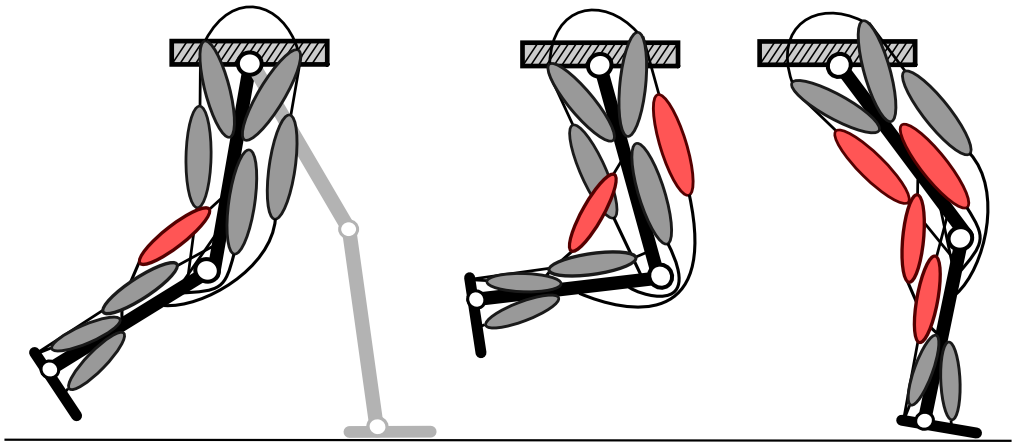

Figure 5.4: Active muscles during knee control. From left to right: Stage *i*, Stage *ii*, and Stage *iii*.

5.1.4 Neuromuscular Control - Knee

The three-stage state machine used in idealized control to regulate knee motion is implemented in the neuromuscular control using a combination of length and velocity feedbacks stemming from hip and knee muscles.

The first control task (Fig. 5.4), active knee flexion, is realized using the mono-articular knee flexor biceps femoris short-head (BFsH), which receives velocity feedback from its bi-articular antagonist rectus femoris

$$S^{BfsH,i}(t) = G_{RF}^{BFsH} V_{RF}^{BFsH}(t - \Delta t_{RF}^{BFsH}) \tag{5.18}$$

with $v_{off}^{RF} = 0$.

The second control task, which corresponds to mid-swing, is responsible for modulating leg length while the foot advances toward target so that the toe does not scuff the ground (Fig. 5.4). The signal for transitioning between the first and second control task stems from the BFsH muscles' length feedback L^{BFsH}, and engages reflexes in the second control task once its contractile element length l_{CE} becomes greater than its offset l_{off}^{BFsH}. Once active, leg extension is achieved during this task using the rectus femoris muscle, stimulated by the signal

$$S^{RF,ii}(t) = G^{RF}_{VAS} V^{RF}_{VAS}(t - \Delta t^{RF}_{VAS}) \tag{5.19}$$

which receives velocity feedback from the vastus muscle that indicates knee extension velocity. Leg flexion is achieved during this task using the BFsH, stimulated by the signal

$$S^{BFsH,ii}(t) = G^{BFsH}_{BFsH} V^{BFsH}_{BFsH}(t - \Delta t^{BFsH}_{BFsH}) M. \tag{5.20}$$

M acts as a selector to implement the specific conditions of the second control task in the idealized swing-leg controller, and is given by

$$M = L^{BFsH}_{RF}(t - \Delta t^{BFsH}_{RF})[V^{BFsH}_{BFsH}(t - \Delta t^{BFsH}_{BFsH}) + V^{BFsH}_{RF}(t - \Delta t^{BFsH}_{RF})]. \tag{5.21}$$

Specifically, $L^{BFsH}_{RF}(t - \Delta t^{BFsH}_{RF})$ implements the $(\alpha - \alpha_{tgt})$ term of equation 4.3, and $[V^{BFsH}_{BFsH}(t - \Delta t^{BFsH}_{BFsH}) + V^{BFsH}_{RF}(t - \Delta t^{BFsH}_{RF})]$ implements the $(\dot{\phi}_k + \dot{\alpha})$ and $(\dot{\phi}_k > \dot{\alpha})$ terms of equation 4.3.

The third control task, responsible for braking the leg, extending the knee, and preventing knee hyper-extension relies on the hamstring, BfSH, and gastrocnemius muscles (Fig. 5.4). The hamstring is stimulated using the equation

$$S^{HAM,iii}(t) = G^{HAM}_{HAM} L^{HAM}_{HAM}(t - \Delta t^{HAM}_{HAM}), \tag{5.22}$$

which is active only once the leg is sufficiently close to the desired target, surpassing as the threshold angle α_{thr}. This condition is enforced by setting the hamstring offset length to $l^{HAM}_{off} = -r^{HAM}_h(\alpha_{thr} - \alpha^{HAM}_r) - l^{HAM}_{sl} + l^{HAM}_0$. Once hamstring is active and its own activation surpasses a preset threshold S_{thr}, hamstring recruits the other knee flexors, BFsH and gastrocnemius, to assist in braking the knee and prevent hyper-extension with the stimulation signals

$$S^{BFsH,iii}(t) = G^{BFsH}_{HAM}[S^{HAM,iii}(t) - S_{thr}], \tag{5.23}$$

and

$$S^{GAS,iii}(t) = G^{GAS}_{HAM}[S^{HAM,iii}(t) - S_{thr}]. \tag{5.24}$$

Once leg angular velocity slows to zero ($\dot{\alpha} = 0$), which is monitored using the hamstring's velocity feedback V^{HAM}, the vastus knee extensor is stimulated to extend the knee until ground contact with the signal

$$S^{VAS,iii}(t) = G^{VAS}_{VAS} L^{VAS}_{VAS}(t - \Delta t^{VAS}_{VAS}). \tag{5.25}$$

5.1.5 Neuromuscular Control - Ankle

Unlike the idealized swing-leg controller, neuromuscular swing-leg control contains an ankle joint. This joint is actuated by the mono-articular ankle flexor tibialis anterior, mono-articular ankle extensor soleus, and bi-articular ankle extensor gastrocnemius. Ankle extensors are not actively stimulated for ankle control during swing; gastrocnemius torques to the ankle result from stimulations generated by the knee control in the previous section, and the soleus is stimulated to maintain a constant pre-stimulation value S^{SOL}_0. To counteract these torques, the tibialis is stimulated with its own length feedback

$$S^{TA}(t) = S^{TA}_0 + G^{TA}_{TA} L^{TA}_{TA}(t - \Delta t^{TA}_{TA}) \tag{5.26}$$

with its length off-set selected to maintain a constant ankle angle throughout swing.

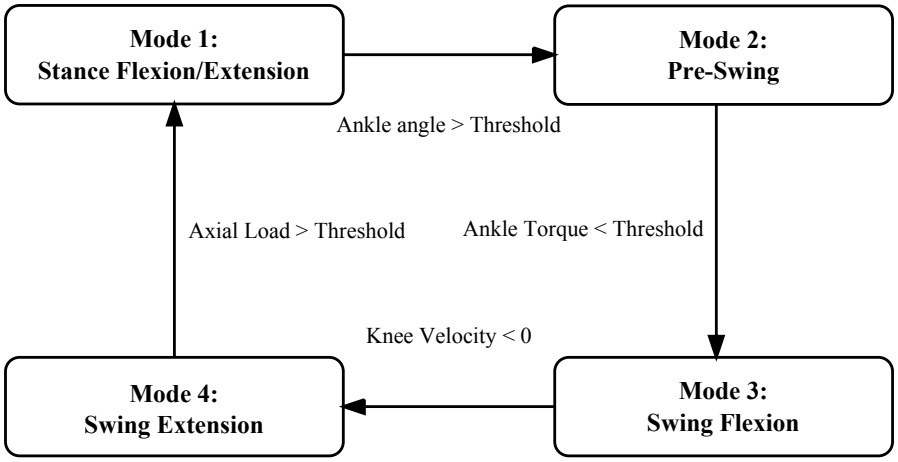

Figure 5.5: Impedance control state machine. Adapted from [138].

5.2 Impedance Control

Whereas experiments in the previous chapter showed that the idealized swing-leg controller is able to accurately regulate foot placement, it is unclear how its performance, as well as the performance of neuromuscular swing-leg control, compares to state-of-the-art locomotion controls. Impedance control is the most widely used control method for legged prosthetic devices to date [8][61][75][76][139][140][143][159][141][142]. Originally formulated as a control strategy for robotic manipulators [51], impedance control has been widely adopted for a variety of robot applications that require a robot to behave compliantly when interacting with its environment [59][78][95].

In human-in-the-loop prosthesis applications, impedance control is used to realize both swing- and stance- leg behavior [138]. The swing-leg portion of this controller is transferred to the RNL test platforms and is used to benchmark idealized and neuromuscular swing-leg performance against state-of-the-art locomotion controls. Its implementation is briefly summarized below.

The impedance controller presented in [138] was originally implemented on a powered knee and ankle prosthesis and divided gait into a series impedance functions which were transitioned between using a state machine (Fig. 5.5). Joint torques τ are commanded to prosthesis actuators based on the position and velocity of each joint using the equation

$$\tau = k_1(\theta - \theta_{des}) - k_2\theta^3 - b\dot{\theta} \tag{5.27}$$

where θ and $\dot{\theta}$ are measured joint position and velocity, respectively. Controller parameters k_1, k_2, and b act as virtual spring-dampers that drive the prosthesis shank and foot segments toward desired joint set-point locations θ_{des}. In [138], parameters were coarsely tuned to match the torque-angle relationship observed in normal, able-bodied walking using linear regression, and then optimized on test subjects *in-vivo* for cosmesis and user comfort.

For benchmark experiments, this controller is transferred to the RNL2 robot presented in section 3.3. While equation 5.27 is used in [138] to generate knee and ankle torques, it is used in swing-leg benchmark experiments to generate hip and knee torques. To verify that this controller can generate human-like motion when controlling other joints, the impedance control scheme is applied to a simulation of RNL2, and its parameters and joint set-points are optimized to match a dynamically scaled, experimentally recorded swing-leg trajectory of able-bodied normal walking [171]. The resulting swing-leg trajectory closely matches the experimentally recorded trajectory (RMSE=0.07 rad) validating that this control scheme can be used to regulate hip motion and serve as a benchmark of the decentralized controllers presented in this thesis.

5.3 Robot for Neuromuscular Control - RNL3

RNL3 is a legged robot designed to directly implement neuromuscular swing-leg control on robotic hardware (Fig. 5.6). A key component of this implementation is that the robot has both antagonistic and bi-articular actuators. Specifically, it is necessary for the robot to have a bi-articular hamstring that spans both the hip and knee joints, which acts to brake the leg as it approaches a desired target angle. While the RNL2 testbed presented in section 3.3 realized antagonistic actuation with cable-driven SEAs, its joint design, which used a combination of cables, sprockets, and chains, was limited to mono-articular SEAs, which could only act across

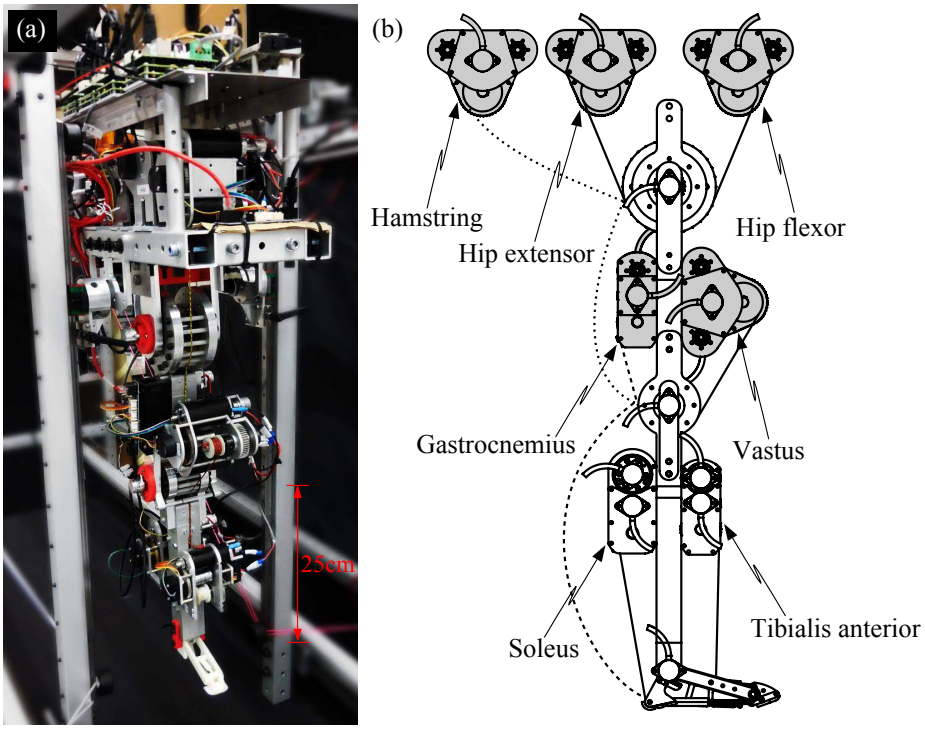

Figure 5.6: The Robotic Neuromuscular Leg 3 (RNL3) test platform. a) Hardware implementation. b) Actuation schematic. Solid: Mono-articular actuator cables. Dashed: Bi-articular actuator cables. Gray: Actuators used to evaluate neuromuscular swing-leg control.

one joint.

To address these shortcomings, RNL3 contains a new joint design composed of small rollers (Fig. 5.7). Cables from the SEAs wrap around these rollers and terminate at the robot's thigh and shank segments. This design both enables a single cable to span multiple joints, and eliminates the massive sprockets present in RNL2, which shifted that robot's center of mass segments close to its proximal joints. Unlike RNL2, which used steel cable and chain in series to transmit torque between each SEA output and the robot, RNL3 uses high-strength, low creep polyester fiber (Mini-V Vectran: ϕ=4 mm, F_{max}=6500 N; Port Supply), which is able to wrap around small-bend radii without kinking.

The design and drivetrain layout of RNL3's SEAs remain the same as RNL2. Additionally, RNL3's hamstring actuator, which was not present in the RNL2 testbed, follows the same layout as the hip extensor, hip flexor, and vastus SEAs. Instead of absolute encoders, RNL3's SEAs uses

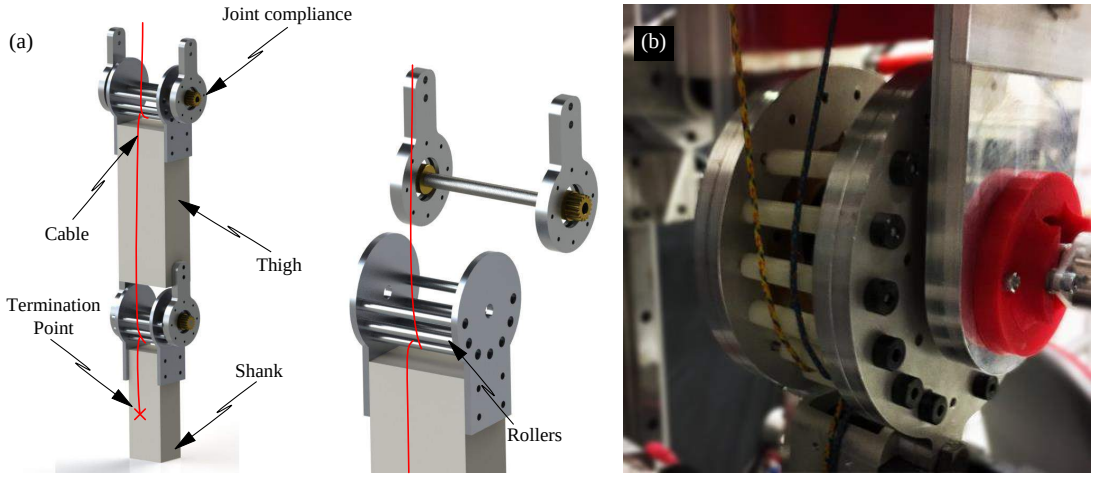

Figure 5.7: RNL3 joint design. a) Schematic and exploded view. b) Hardware implementation.

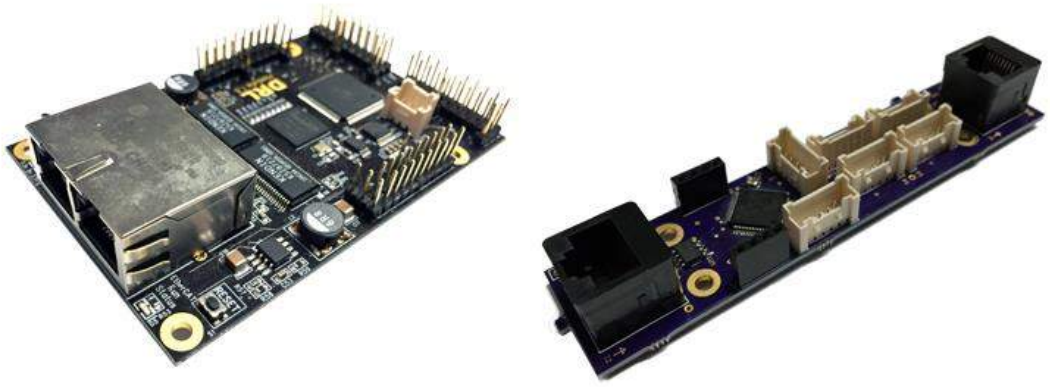

Figure 5.8: RNL3 sensor interface PCBs. Left: Medulla EtherCAT slave. Right: Daisy-chainable sensor interface boards.

two incremental encoders to measure spring deflection (E2-5000-197-IE-D-D-B: 5000 CPR; US Digital), which doubles the SEA torque resolution of each motor compared to RNL2.

Like RNL2, the host PC interfaces with the robot using xPC Target (Mathworks, Inc.) to guarantee synchronous 1 kHz control. Motor controllers used to control RNL3's hip actuators and knee extensor (DZEANTU-040B080B) have twice the current limit than those used on RNL2, allowing them to generate higher peak torques. Motor controllers and all sensors communicate with the host PC via EtherCAT (Beckhoff Automation), interfacing with the host computer using a Medulla EtherCAT slave [73] running custom firmware, which connects to daisy-chainable

	μ_1 (mNm)	μ_2 (mNm)	μ_3 (mNm)	ν_1 (mNms/rad)	ν_2 (mNms/rad)	ν_3 (mNms/rad)
Hip extensor	25.1	<0.1	<0.3	<0.1	<0.1	<0.1
Hip flexor	30.3	<0.1	<0.3	0.2	<0.1	<0.1
Hamstring	19.5	<0.1	<0.3	<0.1	<0.1	<0.1
Vastus	51.1	<0.1	<0.3	0.7	<0.1	<0.1
Gastrocnemius	5.8	<0.1	<0.3	0.2	<0.1	<0.1
Soleus	17.8	<0.1	<0.3	0.3	<0.1	<0.1
Tibialis Anterior	3.9	<0.1	<0.3	<0.3	<0.1	<0.1

Table 5.1: RNL3 drivetrain SEA friction coefficients

interface PCBs for robot sensors (Fig. 5.8).

5.4 Transfer to Hardware Platform: Simulation

The model-based design approach used to transfer idealized swing-leg controls to hardware is adopted to transfer neuromuscular swing-leg controls to RNL3.

Friction for drivetrain stages of each RNL3 SEA are calculated using characterization experiments described in section 4.3.2. Coulomb μ and viscous ν friction coefficients are calculated via linear regression using the equation

$$\mu + \nu\dot{\theta} = -J\ddot{\theta} \tag{5.28}$$

where J is the drivetrain stage's rotational inertia estimated from CAD, $\dot{\theta}$ is the stage's rotational velocity, and $\ddot{\theta}$ is the stage's rotational acceleration. Identified friction parameters are shown in table 5.1.

As for RNL2, a simulation of RNL3 is developed in Simulink SimMechanics (SimMechanics: Second Generation; Mathworks, Inc.) (Fig. 5.9), which models the robot at the individual component level. This simulation is again used to transfer neuromuscular swing-leg controls to hardware, serving as a tool to evaluate if the controller can be applied to robotic systems, and how specific control components need to be implemented to account for hardware constraints. The simulation's software and control architecture is shown in figure 5.10.

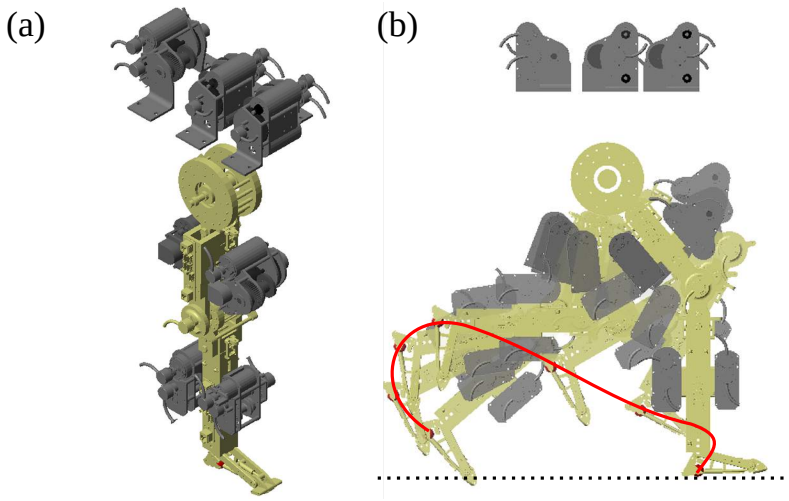

Figure 5.9: RNL3 simulation. a) Screenshot of RNL3 model. b) Undisturbed trajectory experiment motion, α_{tgt}=70^o. Solid: Traced ankle point trajectory. Dotted: Virtual ground.

Two major differences exist between the RNL3 and RNL2 simulations. First, the high-level controller no longer outputs desired net joint torques and distributes them to corresponding SEAs based on sign. Instead, the high-level controller directly converts commanded forces from the virtual muscles in the neuromuscular swing-leg controller to desired torques of corresponding SEAs. As a result, multiple actuators simultaneously act at each joint to generate desired swing-leg behavior. Muscle forces are converted to desired SEA torques by multiplying generated muscle forces by the moment arm of the joint that a SEA acts across. For bi-articular actuators, the desired actuator torque is calculated by multiplying the muscle force with the joint radius closest to the actuator, i.e. while the hamstring acts both across the hip and the knee, desired hamstring torque is calculated as the product between virtual hamstring muscle force and the robot's hip joint radius. The neuromuscular swing-leg controller uses nine muscles, while RNL3 only has seven SEAs. The robot does not have SEAs to directly implement the bi-articular hip flexor-knee extensor rectus femoris, nor the mono-articular knee flexor biceps femoris short head. To overcome this limitation, rectus femoris forces are multiplied by the hip radius and added to commanded hip flexor torques for hip actuation, and are multiplied by the knee radius and added to commanded vastus torques for knee actuation. Similarly, biceps femoris short head torques are

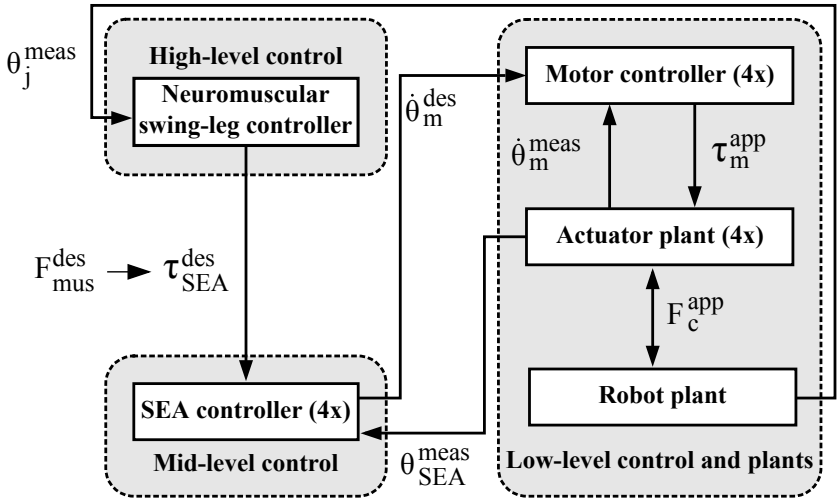

Figure 5.10: RNL3 software and control architecture. θ_j^{meas}: measured joint angles. F_{mus}^{des}: desired muscle forces. τ_{SEA}^{des}: desired SEA torques. $\dot{\theta}_m^{des}$: desired motor velocities. $\dot{\theta}_m^{meas}$: measured motor velocities. τ_m^{app}: applied motor torques. F_c^{app}: force transmitted by cable drive. θ_{SEA}^{meas}: measured spring angles.

multiplied by the knee radius and added to commanded gastrocnemius torques for knee actuation. Since the gastrocnemius is a bi-articular knee-ankle actuator, this additional torque would need to be compensated for at the ankle using the tibialis anterior actuator for a fully actuated three segment robot.

The second key difference is that unlike RNL2, whose entire control system ran at 1kHz, subsystems of RNL3's control system run at multiple execution speeds. High-level neuromuscular control runs at 5 kHz. This change was necessary to stabilize virtual muscle models that generate force commands, as coarse integration steps result in numerical instability when inverting the force-velocity relationship of the Hill-type model, which is used to calculate the muscles' contractile element length l_{ce} [156]. RNL3's SEA controller and data transmission interfaces run at 1 kHz. Due to the motor controller's 10 kHz operating frequency, their dynamics, as well as the rest of the hardware, are modeled as continuous time systems.

All simulation gains are tuned using optimization based on the covariance matrix adaptation evolution strategy (CMA-ES) [43], with a cost function to minimize the sum squared difference between the desired and measured signals at each control level, as well as an explicit cost term

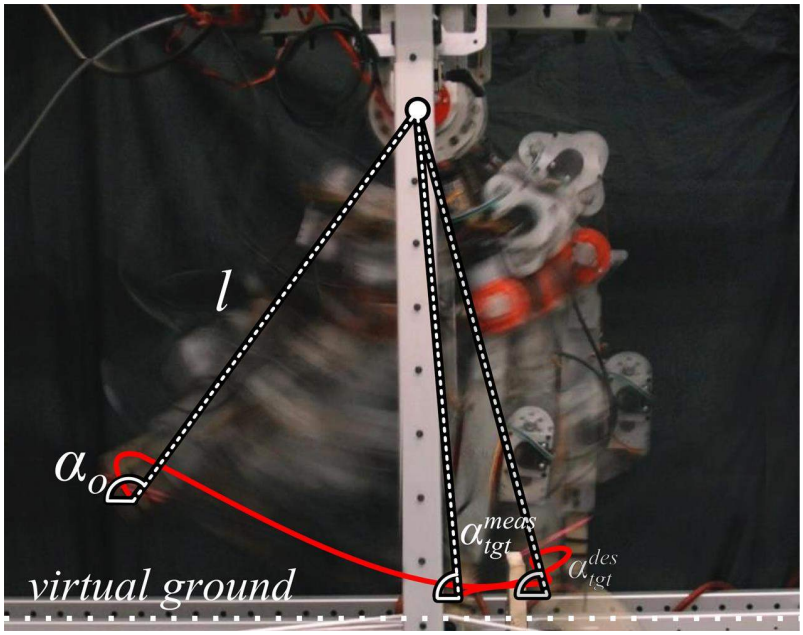

Figure 5.11: Example disturbed neuromuscular swing-leg control experiment. Shown: α_{tgt}=70^o, late disturbance.

to penalize the optimization if the final leg extension phase does not trigger during swing. This latter term was necessary, as the optimization was unable to find gains to generate swing-leg behavior that did not scuff the ground without an additional cost penalty.

Gains are again tuned hierarchically starting at the lowest level, ensuring that the motor controller can follow commanded velocity signals. Mid-level SEA gains are then tuned to ensure that the actuators are able to realize commanded torques. Finally, high-level neuromuscular gains are optimized to minimize the difference between the desired and measured α_{tgt}.

This optimization mimics the tuning procedure of hardware gains used in the below experiments. In hardware, all gains are manually tuned, with high-level gains using the optimized simulation gains as a starting point.

5.5 Swing-Leg Control Evaluation

5.5.1 Hardware Experiments

Hardware experiments to evaluate the performance of neuromuscular swing-leg controls mimic those used to evaluate idealized swing-leg controls in section 4.5.1. For these experiments, RNL3 is suspended from its hip in a rigid mounting cage. During experiments, the robot starts at a neutral position (Fig. 5.6). A feed-forward torque that is constant across all experiments moves the leg into an initial pose of α_0=118^o, matching the initial pose used to tune control gains in simulation. This motion also imparts initial joint velocities. At the initial pose, the virtual ground location initializes, defined as a horizontal plane located at the robot's toe. Segment lengths used to calculate this ground location are taken from CAD. Motion continues until the robot's heel makes contact with the virtual ground plane. Since the primary role of the soleus and tibialis anterior actuators is to keep the foot at a constant orientation relative to the shank during swing, and a foot design suitable for future stance experiments was not finalized at the time of testing, the ankle segment was locked during these experiments.

To enable direct comparison to swing-leg experiments performed in the previous chapter, two sets of experiments are used to evaluate the controller. Undisturbed motion experiments (Fig. 5.1) test the controller's ability to place feet into desired ground targets when the swing-leg is unperturbed. Disturbed motion experiments simulate tripping and test the controller's ability to regulate foot placement into desired ground targets when the robot encounters an unexpected obstacle in early-, mid-, and late-swing (Fig. 5.11). The same 600 g obstacle used in idealized swing experiments is used to evaluate neuromuscular control. The obstacle sits on rockers and approximates an impulse disturbance as the robot knocks the obstacle down during swing. For disturbed experiments, the robot's foot is removed so that disturbances occur at the ankle, though foot geometry is still considered when calculating the height of the virtual ground. Five trials are conducted for each locomotion condition with a desired α_{tgt} range between 65^o to 90^o.

To compare both idealized and neuromuscular controls to state-of-the-art locomotion controls, the impedance controller described in section 5.2 is implemented on the RNL2 robot presented in section 3.3. Initial pose, virtual ground initialization, and swing-leg termination are the same as presented in section 4.5.1. Impedance gains are hand-tuned for undisturbed swing, with α_{tgt}=70^o, which roughly corresponds to normal walking. The robot under impedance control is exposed to the same experimental conditions as the robot under idealized and neuromuscular control, and attempts to place feet into desired ground targets both when swing-leg motion is undisturbed, as well as when the robot encounters unknown obstacles in early-, mid-, and late-swing.

5.5.2 Simulation Experiments

Simulation experiments further explore the performance comparison between decentralized swing-leg controls and impedance controls. Since impedance control gains do not generalize to different locomotion conditions, it is necessary to retune the controller for different desired ground targets. With multiple gain sets, it is then possible to interpolate between gain values to enable to controller to execute foot placements over the α_{tgt} range. This approach essentially creates a motion library. Experiments presented in this section aim to determine how large the impedance controller's motion library needs to be in order to match the performance of both idealized and neuromuscular swing-leg controls.

Due to the number of gains and desired ground targets in the testing range, it is impractical to tune the impedance controller for all locomotion conditions on hardware. Instead, the high-fidelity simulations of the RNL2 and RNL3 systems presented in the previous sections are used in conjunction with optimization to generate multiple gain sets and further explore the performance between control methods.

CMA-ES is used to tune impedance gain sets for ground targets between α_{tgt} values of 60^o and 90^o at 5^o increments. During optimization, the impedance controller is exposed to undis-

Disturbance	None	Early	Mid	Late
Idealized	-1.5±0.6	-3.8±0.3	-4.3±0.6	-5.9±0.9
Neuromuscular	-5.0±2.3	-3.1±3.4	-2.5±4.0	-5.1±2.2
Impedance*	5.2±2.4	0.37±1.9	7.2±3.0	2.0±1.6

Table 5.2: Mean hardware placement error (^{o}) for α_{tgt} range: 65^o to 90^o. * α_{tgt}=70^o only. RNL2 used for idealized swing-leg control and impedance control experiments. RNL3 used for neuromuscular swing-leg control experiments.

Disturbance	None	Early	Mid	Late
Idealized	519±56	702±56	649±202	547±50
Neuromuscular	451±4	495±20	470±7	456±4
Impedance	578±32	690±28	649±90	672±17

Table 5.3: Mean hardware swing times in ms for α_{tgt}=70^o.

turbed and all disturbance cases used to tune hardware gains. Generated gain sets are used as interpolation nodes for an impedance control motion library. Simulated foot placement experiments are then performed, which test the controller's ability to regulate foot placements into ground targets between α_{tgt} values of 60^o and 90^o at 1^o increments, with motion libraries that have the following nodes: 3 nodes (α_{tgt}=60^o, 70^o, 90^o), 4 nodes (α_{tgt}=60^o, 70^o, 80^o, 90^o), and 7 nodes (α_{tgt}=60^o, 65^o, 70^o, 75^o, 80^o, 85^o, 90^o). Motion library nodes are interpolated for intermediate ground targets using spline interpolation.

To enable a direct comparison between the impedance controller and decentralized swing-leg controls for different motion library sizes, idealized and neuromuscular control gains are reoptimized at each motion library size. This optimization results in a single gain set for each controller at each motion library size. Simulated foot placement experiments using these controllers are then repeated for the same ground target range and resolution as simulated impedance control experiments.

5.5.3 Discussion

The conducted swing-leg experiments aim to address two hypotheses. The first hypothesis posits that the proposed swing-leg controllers enable more robust foot placements into ground

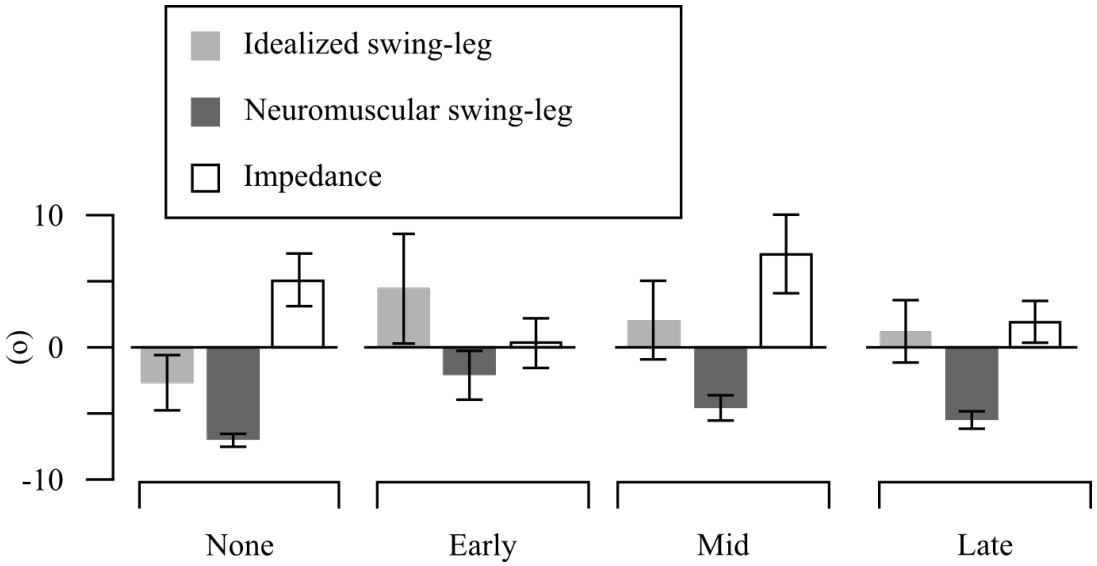

Figure 5.12: Hardware foot placement results for α_{tgt}=70^o locomotion conditions. RNL2 used for idealized swing-leg control and impedance control experiments. RNL3 used for neuromuscular swing-leg control experiments.

targets than state-of-the-art impedance controls. The second hypothesis posits that neuromuscular swing-leg control enables more human-like motion than both idealized swing-leg control and impedance control.

Hardware experiments show that neuromuscular swing-leg control accurately regulates foot placement into desired ground targets, both when motion is undisturbed, as well as when the robot encounters unknown obstacles (Tab. 5.2). Comparing the foot placement accuracies achieved by each controller at α_{tgt}=70^o, the locomotion condition that each controller was tuned for, reveals that while the proposed controls achieve accurate foot placements into desired ground targets, placement accuracy achieved by the proposed controllers for this condition is comparable to the accuracy achieved by impedance controls (Fig. 5.12). However, the proposed controls enable foot placement into a wide target range with a single gain set.

Due to the formulation of impedance control, it is not possible to use the same set of control gains to place feet into multiple ground targets. This is unlike both the idealized and neuromuscular swing-leg control, which can achieve foot placements into a range of ground targets with a single set of gains. Foot placement accuracies achieved by the proposed controllers over

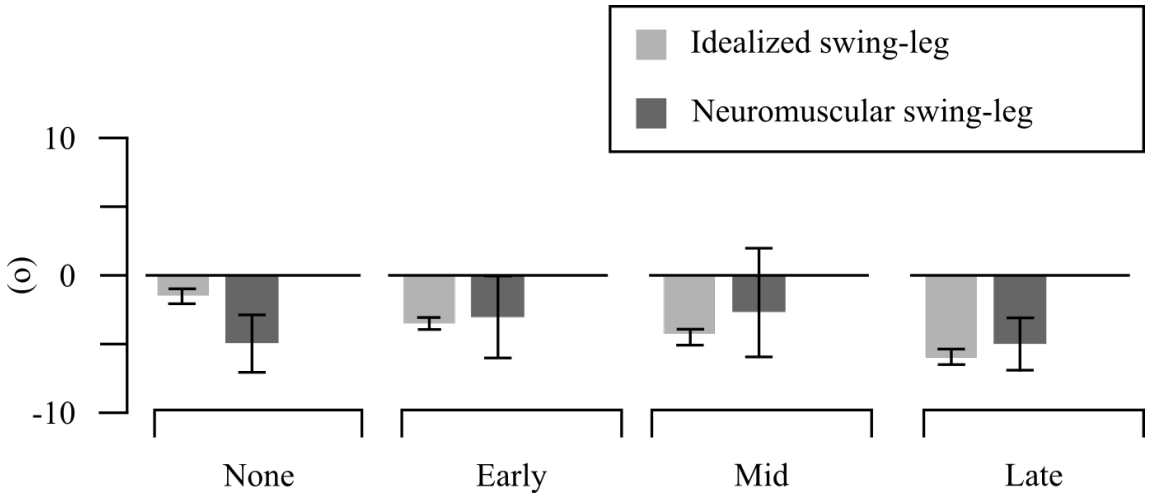

Figure 5.13: Hardware foot placement results for α_{tgt} range: 65^o to 90^o locomotion conditions. RNL2 used for idealized swing-leg control and impedance control experiments. RNL3 used for neuromuscular swing-leg control experiments.

an α_{tgt} range between 65^o to 90^o are comparable to accuracies achieved by the controls for the locomotion condition they were tuned for (Fig. 5.13). Therefore, the proposed controls enable more robust foot placements into ground targets than state-of-the-art impedance controls over a wide target range with a single set of controller gains.

An additional advantage of the proposed controls is that the proposed controls require the tuning of fewer gains than impedance controls. Idealized control requires 6 gains to be tuned, neuromuscular control requires 12 gains to be tuned, and impedance control requires 16 gains per locomotion condition to be tuned. This suggests that both proposed control methods possess an implementation advantage compared to impedance control, making them more suitable for practical application on legged robots and in clinical settings.

Hardware ankle point trajectories of undisturbed swing generated by neuromuscular control more closely match human trajectories than ankle point trajectories generated by idealized swing-leg and impedance control, both in terms of cosmesis, as well as standard deviation of the landing angle at touchdown between consecutive swing-leg motions (Fig. 5.14). To compare human ankle trajectories to those generated by hardware, motion capture (MX40: Vicon) was used to record joint trajectories of a single subject walking at self-selected walking speed

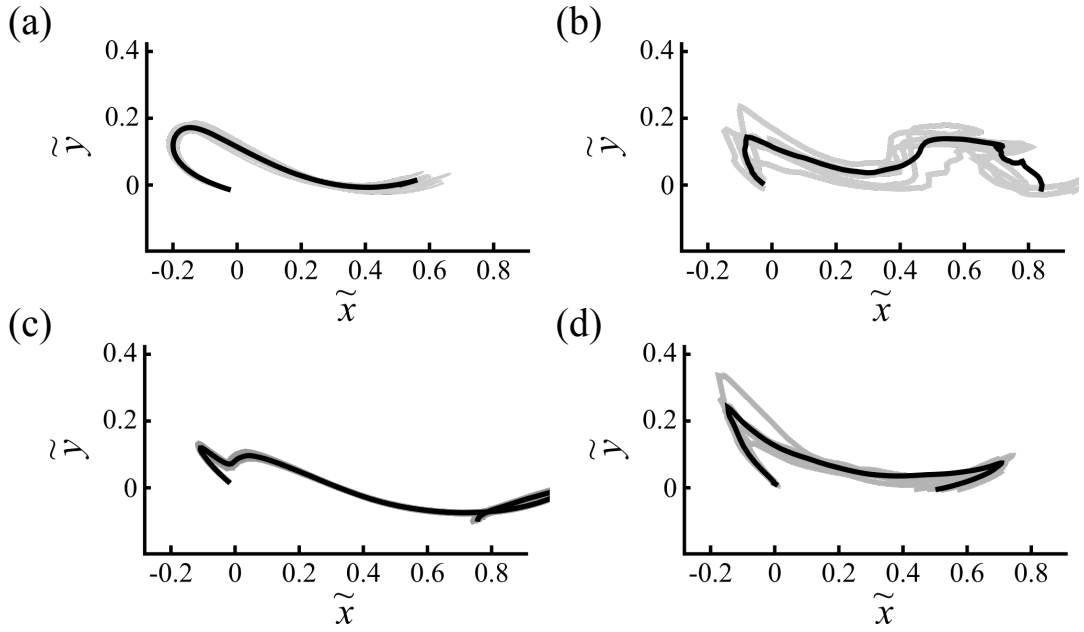

Figure 5.14: Ankle point trajectories for α_{tgt}=70^o undisturbed locomotion condition, normalized by leg length. a) Human. b) Idealized. c) Neuromuscular. d) Impedance. Black: Mean trajectory. Gray: Individual trajectories. RNL2 used for idealized swing-leg control and impedance control experiments. RNL3 used for neuromuscular swing-leg control experiments.

and stride length on a split-belt treadmill (Belt speed = 1.3 m/s). To compare the cosmetic similarity between human and hardware ankle point trajectories, two-dimensional correlation between the human trajectory and each hardware trajectory was performed after normalizing each for leg length. Neuromuscular control exhibited the highest correlation to the human trajectory (R=0.89), followed by impedance control (R=0.81), and idealized control (R=0.71). At the self-selected walking speed and stride length, the human subject landed at a target angle of α_{tgt}=73.0$\pm$0.8^o (n=28). This α_{tgt} at touchdown roughly corresponds to the tuned-for undisturbed locomotion condition of α_{tgt}=70^o in hardware. For these locomotion conditions, foot placement under neuromuscular control exhibited a standard deviation of $\pm$0.5^o, approximately the same variability of human trials, whereas the standard deviation of both idealized swing-leg control and impedance control was larger, at $\pm$2.3^o and $\pm$2.4^o, respectively. These results suggest that neuromuscular control enables more human-like motion than both idealized swing-leg control and impedance control.

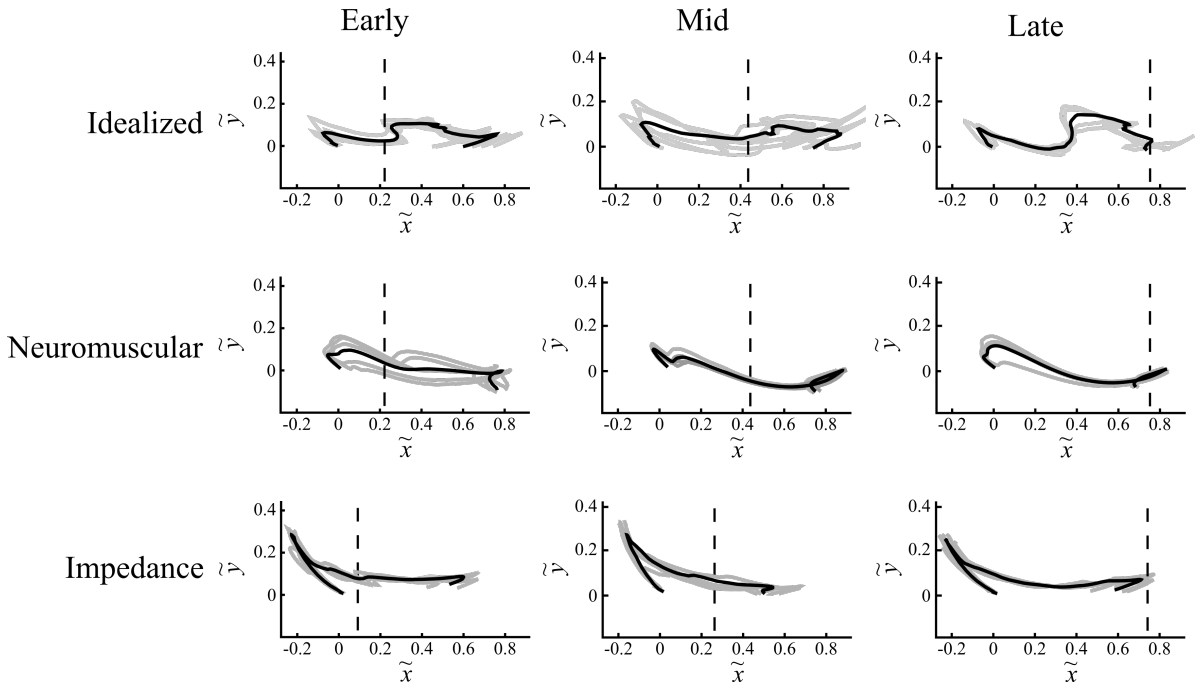

Figure 5.15: Ankle point trajectories for α_{tgt}=70^o disturbed locomotion conditions, normalized by leg length. Dashed: Obstacle location for locomotion condition. Black: Mean trajectory. Gray: Individual trajectories. RNL2 used for idealized swing-leg control and impedance control experiments. RNL3 used for neuromuscular swing-leg control experiments.

Neuromuscular control also generates the most human-like swing duration of the tested controllers (Tab. 5.3). The swing duration under both idealized and neuromuscular control is shorter than the swing duration under impedance control, likely due to impedance control's explicit knee flexion at the beginning of swing. While still not matching the dynamically scaled duration of human swing (Tab. 4.3), the mean undisturbed neuromuscular swing-leg duration is 15% faster than the swing duration under idealized control, and 28% faster than the swing duration under impedance control.

Both idealized and neuromuscular control also elicit a human-like response to sudden swing-leg disturbances (Fig. 5.15). Early-swing disturbances caused the robots to raise their feet, consistent with foot elevation strategies observed in humans, and late-swing disturbances caused the robots to lower their feet, consistent with foot lowering strategies observed in humans [30]. Looking at the the average trajectories for each disturbance case shows that the elevation strategy

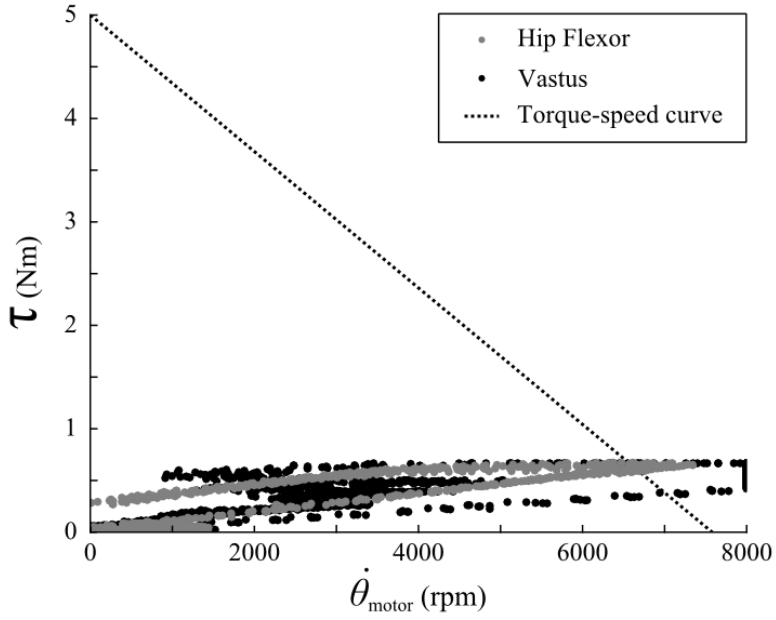

Figure 5.16: Torque-speed commands generated by the neuromuscular controller for undisturbed locomotion trials α_{tgt}=70^o. Gray: Hip flexor. Black: Vastus. Dashed: Torque speed curve of dual RE40 actuator configuration.

executed under idealized control was more pronounced than that seen under neuromuscular control. Ankle trajectories under impedance control remain roughly consistent during early-, mid-, and late-swing disturbance cases, suggesting that impedance control does not generate similar behavior and does not dynamically react to obstacles.

While the trajectory traced out by the ankle point under neuromuscular control is more human-like, high-speed video of the robot shows that the neuromuscular controller on RNL3 is unable to generate sufficient foot clearance to keep the toe from scuffing the ground during swing (Fig. 5.1). While idealized and impedance control do not explicitly account for foot geometry, these controllers were unable to generate enough clearance to preven toe scuffing had a foot been present. Scuffing behavior results from motor saturation during the experiments, as commanded torques generated by the mid-level SEA controller to realize desired actuator torques is outside of the motors' torque-speed curve. In neuromuscular control, this behavior occurs most notably in the hip flexor, which is primarily responsible for driving the hip point towards the desired α_{tgt}

and generating sufficient passive knee flexion in early swing in order to achieve ground clearance in mid-swing (Fig. 5.16). However, saturation is also seen in the vastus actuator, responsible for extending the shank into the desired ground target in late swing. This issue could be addressed in one of two ways. First, higher voltage motor controllers, capable of achieving the commanded torque-speed values required by the neuromuscular swing-leg controller could be used to drive RNL3's actuators. Alternatively, commanded torques to the hip flexor actuator could be reduced by adding an additional bi-articular actuator that directly realizes torques commanded by the virtual rectus femoris muscle. Like the hamstring, this actuator could be located in the robot's trunk, thereby not increasing its overall leg mass. Torques commanded by the neuromuscular controller could also be indirectly reduced by decreasing the robot's mass. Through finite element analysis on actuator plates and robot segments, material not necessary to maintain the robot's structural integrity could be removed, which would decrease the robot's weight. As a result, lower control gains could be used to generate the same motion at lower torques, and provide additional actuation bandwidth to increase toe clearance during swing.

Simulation experiments show that impedance control achieves comparable performance to idealized and neuromuscular control for disturbed conditions with 3 interpolation nodes, and roughly equivalent mean performance for all conditions with 7 optimization nodes. Increased foot placement accuracy for disturbed swing likely results from the cost function used to optimize the impedance controller, which gave each experimental condition equal weight. For disturbed trials, the obstacle acted to brake forward motion in mid- and late-swing. This caused the impedance control to converge on aggressive swing extension gains, which forcefully drive the foot point towards a ground target. When the motion was undisturbed, however, these gains cause the leg to extend beyond the target, and make ground contact when the leg is in front of the swing target with a large placement error.

Optimization results also suggest that the neuromuscular controller is more difficult to tune than its counterparts. Unlike the idealized swing-leg and impedance controllers, whose placement accuracy either remains constant or decreases with additional optimization conditions, neu-

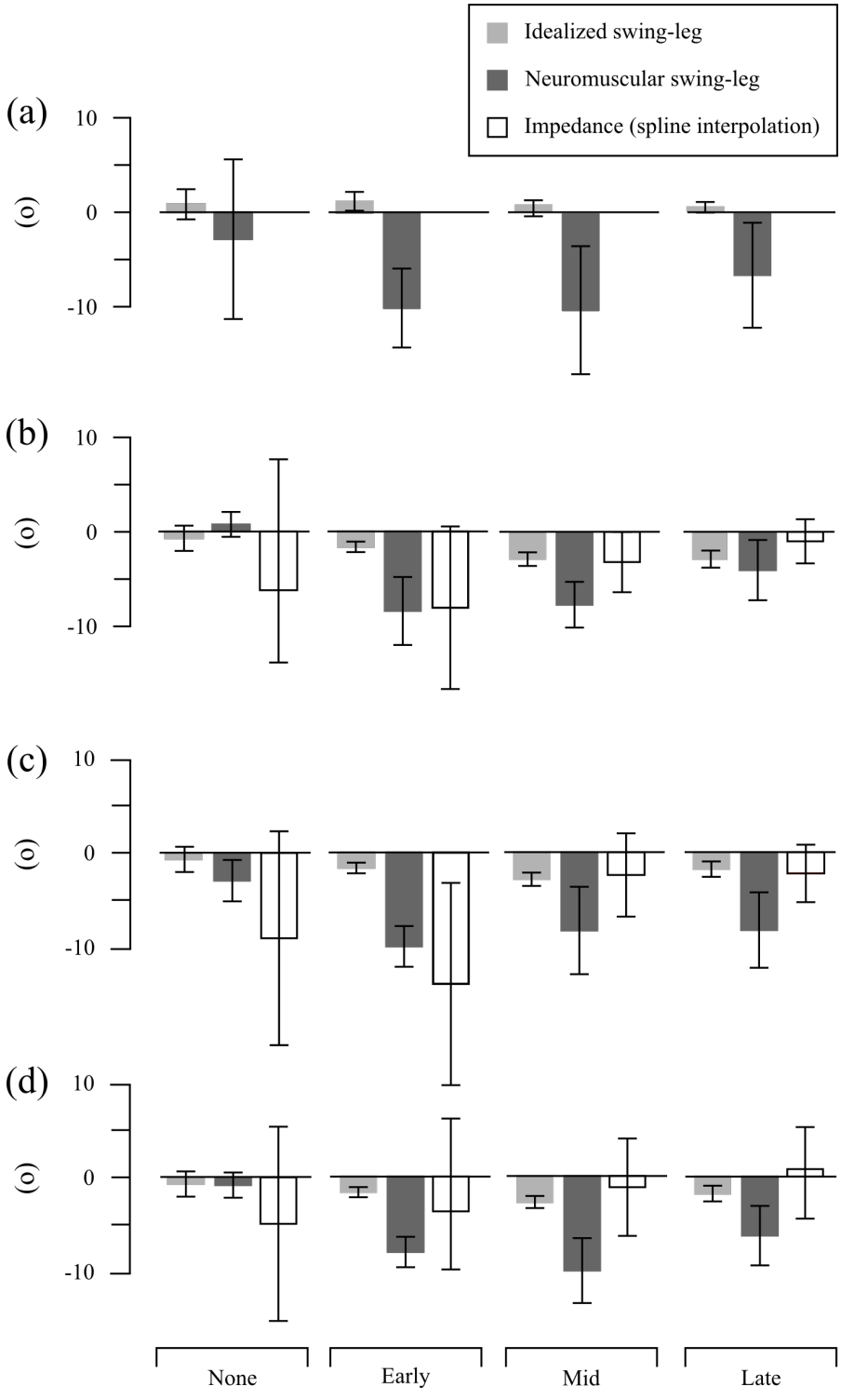

Figure 5.17: Mean placement error for hardware simulation results. α_{tgt} range: 60^o to 90^o. a) 1 node. b) 3 nodes. c) 4 nodes. d) 7 nodes. Idealized swing-leg control and impedance control simulation performed with RNL2 model. Neuromuscular swing-leg control performed with RNL3 model.

romuscular controller placement accuracy appears to hit a local optimum for 3 optimization nodes, since placement error increases with 4 optimization nodes. One explanation for this behavior is that neuromuscular control gains are tightly coupled, as activations of one muscle affect the activation of other muscles in the leg, whereas both idealized and impedance control gains are largely decoupled.

While foot placement accuracy suggests that idealized swing-leg control is the best performing swing-leg controller, neuromuscular control may be advantageous to use in modular prosthetic devices and in devices that regulate motion during the entire gait cycle. Unlike idealized control, which attempts to regulate net joint torque, the neuromuscular controller issues torque commands to individual actuators, which act in aggregate around a joint to achieve a desired net torque. As such, neuromuscular swing-leg control could be used in powered devices that assist the human user, instead of regulating the motion for them entirely. Additionally, neuromuscular swing-leg controls directly integrate with neuromuscular stance control, which has been shown to qualitatively reproduce human leg behavior during walking on a transfemoral prosthesis [148].

5.6 Summary

This chapter presented work to transfer a neuromuscular swing-leg controller to robotic hardware, and evaluate its ability to place feet in desire ground targets. Neuromuscular swing-leg performance was compared to the idealized swing-leg controller presented in the previous chapter, as well as impedance control, the current state-of-the-art control approach for powered robotic legs. Experiments show that neuromuscular control is able to robustly place feet into desired ground targets, and that its performance compares favorably to impedance control, as it is able to place feet into a variety of target locations with a single set of gains. Though neuromuscular control does not enable the robot to place its feet into ground targets with as high accuracy as idealized swing-leg control, the resulting motion is more human-like.

Chapter 6

Nonlinear Springs for Improved SEA Performance

Material in this chapter is based on:

A. Schepelmann, K.A. Geberth, and H. Geyer
Compact nonlinear springs with user defined torque-deflection profiles for series elastic actuators.
Proceedings of the IEEE International Conference on Robotics and Automation: 385-392, 2014. [118]

J. Austin, A. Schepelmann, and H. Geyer
Control and evaluation of series elastic actuators with nonlinear rubber springs.
Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems: 6563-6568, 2015. [5]

The SEAs used to actuate the RNL testbed platforms discussed in the previous chapters use linear metal springs as the torque transmitting elements in their drivetrains. Using linear metal springs is common practice for SEAs [97][105][151][70]. As commercial, off-the-shelf products, metal springs can be easily bought at low prices. However, using series linear springs in the drivetrain requires compromises between torque resolution and actuation bandwidth [161][172]. Torque resolution describes the smallest change in torque that can be measured. It is a function of spring stiffness, with softer springs providing higher torque resolution. Bandwidth describes the speed at which an actuator can apply torque to a load [115]. It can be characterized by rise time, the time an actuator takes to apply its maximum torque to a load when starting from rest, which, for SEAs, also depends on the series spring stiffness. Softer springs cause longer rise times, resulting in lower actuation bandwidth.

In contrast to linear springs, nonlinear torque transmitting elements with variable stiffness enable both high torque resolution and high bandwidth. For instance, variable stiffness actuators enable a range of torque resolutions and actuation bandwidths with a single actuator, by actively tuning a passive mechanical element with a secondary motor [57][149][157][60]. However, such actuators are mechanically complicated, large, and heavy, hampering their application in small, lightweight robots, as well as for retrofit in existing robots.

SEAs with purely passive nonlinear springs (NLSs) are a subset of variable stiffness actuators that omit active tuning of the mechanical element at the cost of embedding a single nonlinear torque-deflection profile. The profile can be designed to meet desired torque resolution and rise time goals. Different realizations of this idea have been pursued, many of which deflect linear metal springs with cams of changing radius. Early designs favored mechanically complex devices for mathematical simplicity, where the cam radius and spring force vectors are kept prependicular throughout the NLS deflection [84][85]. More recently, NLS designs have been simplified mechanically at the cost of mathematical complexity. In [120], the authors identified a closed-form solution to cam profiles for which a linear spring wraps around the cam. While the assumption of a linear spring was necessary to obtain the solution, it does not fully explore the potential for miniaturizing passive nonlinear springs, as the cam profile was again designed around commercially available spring form factors and stiffnesses [121].

Rubber is an alternative to commercial linear metal springs as the elastic element in NLS designs. Although rubber has some disadvantages when compared to metal springs, such as increased hysteresis and van der Wall force-dependent stiffness characteristics [33][6], it tolerates large stretches before plastic deformation and can be molded with custom form factors. Due to these advantages, rubber is already being used in SEAs when actuator size matters [116]. In contrast to metal springs, rubber springs can be nonlinear, especially at large stretches [68]. Therefore, the closed form solution for NLS designs developed in [120] cannot be applied.

To overcome this limitation, work in this chapter proposes and demonstrates an optimization-based synthesis method for compact NLSs that generate desired nonlinear torque profiles. The

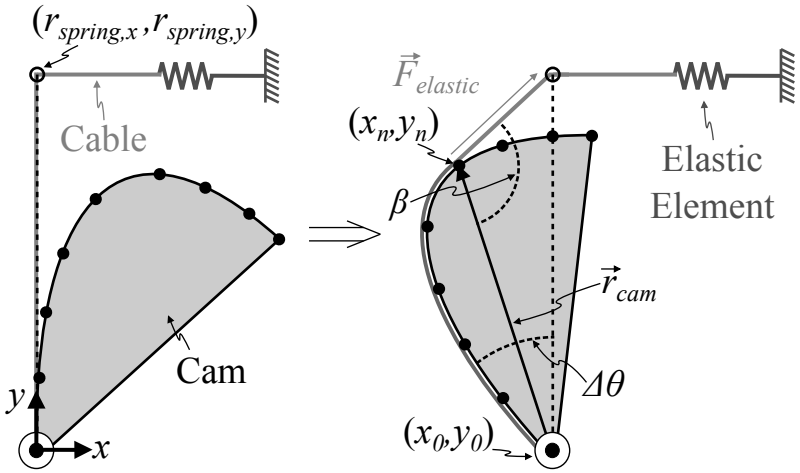

Figure 6.1: NLS concept. L: Undeflected spring. R: Spring deflected $\Delta\theta$. A cam engages a cable attached to an elastic element as it rotates about (x_0, y_0), stretching the elastic element. The cross product of the radius from (x_0, y_0) and the last contact point between the cable and cam (x_n, y_n) with the force generated by the elastic element's stretch creates a desired torque $\boldsymbol{\tau_{des}}$.

following sections describe the optimzation procedure to generate NLS designs, and then, using a particular NLS design as an example, outline the manufacturing process of an NLS prototype. Next, the resulting prototype is characterized via experiments. The experiments show that custom torque profiles can be realized with a small form factor, but also point to drawbacks that stem from using rubber material as the elastic element, namely rubber hysteresis. To account for hysteretic effects introduced by the rubber, a state osbserver is created that models for the rubber's viscoelastic behavior and produces a more accurate estimate of torque transmitted through the NLS. NLS performance in conjunction with the observer is evaluated in an experimental testbed and compared to the performance of traditional linear metal springs to determine if benefits of both soft and stiff elastic elements can be captured in a single mechanism.

6.1 Nonlinear Spring Optimization

The proposed NLS design is a two part assembly, consisting of an elastic element and a rotary cam whose profile is optimized to stretch the elastic element over a variable radius (Fig. 6.1). The cam is defined by a set of (x, y) pairs.

The optimization uses evenly spaced torque-deflection pairs of desired torques $\boldsymbol{\tau_{des}}$ for spring deflections $\Delta\theta$

$$\begin{bmatrix} \boldsymbol{\tau_{des}} \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} 0 \cdots \boldsymbol{\tau_{des,max}} \\ 0 \cdots \Delta\theta_{max} \end{bmatrix} \tag{6.1}$$

where $\boldsymbol{\tau_{des,max}}$ and $\Delta\theta_{max}$ are the maximum desired torque and spring deflection, respectively, and the elastic element's force-displacement relationship

$$\boldsymbol{F_{elastic}} = f(\Delta s) \tag{6.2}$$

where Δs is the change in the elastic element's length due to cam engagement. Δs is the sum of the Euclidian distance between points on the cam with the distance between the cam's last contact point and attachment point to the NLS housing, minus the elastic element's rest length. Δs occurs as convex points on the cam (x_i, y_i) engage the elastic element throughout the cam rotation, and is given by

$$\begin{aligned} \Delta s = &\sum_{i=1}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} + \\ &\sqrt{(r_{spring,x} - x_n)^2 + (r_{spring,y} - y_n)^2} - \\ &\sqrt{(r_{spring,x} - x_0)^2 + (r_{spring,y} - y_0)^2} \end{aligned} \tag{6.3}$$

where (x_n, y_n) is the last, tangent contact point between the cam and elastic element and $(r_{spring,x}, r_{spring,y})$ is the elastic element's contact point with the NLS housing. This latter parameter is based on the desired NLS diameter.

To optimize the cam profile, the covariance matrix adaptation evolution strategy (CMA-ES) algorithm [43] is used. The optimization modifies (x, y) pairs of the cam to minimize the differ-

Input: Cam profile
Calculate convex hull of cam profile;
for $\Delta\theta_j = 0$ *to* $\Delta\theta_{max}$ **do**
    Rotate profile to $\Delta\theta_j$;
    Find (x_n, y_n) between convex hull and elastic element;
    Calculate Δs;
    Calculate $F_{elastic}(\Delta s)$;
    Calculate $\boldsymbol{\tau_{NLS}}(\Delta\theta)_j$;
end
$J(\boldsymbol{\tau_{NLS}}(\Delta\theta))$=$\sum_{j=1}^{m} (\boldsymbol{\tau_{des}}(\Delta\theta)_j - \boldsymbol{\tau_{NLS}}(\Delta\theta)_j)^2$;

Algorithm 1: Cam optimization criterion.

ence between $\boldsymbol{\tau_{des}}$ and the cam profile's current torque $\boldsymbol{\tau_{NLS}}$ using the cost function

$$J(\boldsymbol{\tau_{NLS}}(\Delta\theta)) = \sum_{j=1}^{m} (\boldsymbol{\tau_{des}}(\Delta\theta)_j - \boldsymbol{\tau_{NLS}}(\Delta\theta)_j)^2 \tag{6.4}$$

where

$$\boldsymbol{\tau_{NLS}}(\Delta\theta)_j = \boldsymbol{r_{cam}}(\Delta\theta)_j \times \boldsymbol{F_{elastic}}(\Delta\theta)_j \tag{6.5}$$

and $\boldsymbol{r_{cam}}(\Delta\theta)_j$ is the vector from the cam's center of rotation (x_0, y_0) to (x_n, y_n) at the rotation $\Delta\theta_j$. The direction of $\boldsymbol{F_{elastic}}$ is defined by (x_n, y_n) and $(r_{spring,x}, r_{spring,y})$. During each iteration, the optimization criterion (Alg. 1) first calculates the convex hull of the current cam profile. The cam profile then incrementally rotates through the deflection range. At each rotation, (x_n, y_n) is found using line intersection; starting at (x_0, y_0), the criterion checks for line intersections between $\boldsymbol{F_{elastic}}$ and subsequent points on the cam profile. (x_n, y_n) is found when only one intersection exists. Next, the elastic element's resulting length due to cam rotation is calculated as the difference between the elastic element's rest length and the sum of the Euclidian distance between sequential points from (x_0, y_0) to (x_n, y_n) and (x_n, y_n) to $(r_{spring,x}, r_{spring,y})$. From this, $\boldsymbol{F_{elastic}}$ is evaluated, followed by $\boldsymbol{\tau_{NLS}}(\Delta\theta)_j$. After rotating through the deflection range, $J(\boldsymbol{\tau_{NLS}}(\Delta\theta))$ is evaluated, and the optimization generates a new cam profile, finally resulting in a set of (x, y) pairs that define the cam profile.

6.2 NLS Manufacturing Process

This section describes the NLS manufacturing process. The desired torque-deflection profile is determined via optimization based on design requirements. Next a rubber is selected as the elastic element, and its force-deflection relationship is characterized. A NLS prototype is then manufactured. Key components of the prototype are sized to meet geometric requirements, but are evaluated on a benchtop setup for ease of testing.

6.2.1 Torque Profile Optimization

The created NLS is targeted for use in the SEAs of the RNL test platforms. The NLSs for these SEAs must generate an exponentially stiffening torque with a zero torque at zero deflection, and a maximum torque τ_{max}=5 Nm at $\Delta\theta = \Delta\theta_{max}$. As a function of these constraints, the NLS torque profile is

$$\tau_{des}(\Delta\theta, \tau_{max}, \Delta\theta_{max}) = \frac{\tau_{max}}{e^{\Delta\theta_{max}} - 1}(e^{\Delta\theta} - 1). \tag{6.6}$$

The springs must exhibit a rise time of t_{rise}=16 ms, and have a spring diameter less than 5 cm. $\Delta\theta_{max}$ is defined by the rise time. To meet t_{rise}, $\Delta\theta_{max}$ is optimized by numerically solving the motor dynamics equation

$$J_m\ddot{\theta}_m = \tau_{motor} - \tau_{des}(\Delta\theta, \tau_{max}, \Delta\theta_{max}) \tag{6.7}$$

using constrained nonlinear optimization (MATLAB *fmincon*, active-set algorithm), where J_m is the reflected SEA inertia at the spring, θ_m is the motor angle, τ_{motor} is the applied motor torque, and the actuator is assumed to start with no spring deflection from rest $\Delta\theta(0) = \dot{\theta}_m(0)$=0. In this case, $\Delta\theta$ is given by θ_m, as the output shaft is assumed to be clamped. J_m is estimated as the total reflected inertia of the motors, gears, and cam; inertia from encoders, shafts, and fasteners are ignored. During each iteration, the optimization numerically solves equation 6.7 via a fourth

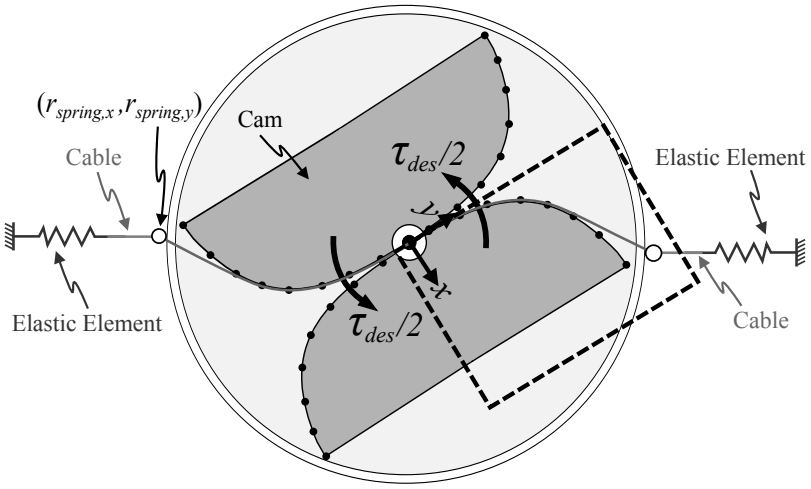

Figure 6.2: NLS prototype schematic. The NLS is realized with two pieces: a cam and an outer housing, to which the cable and elastic element are attached. The elastic element stretches when the cable engages the cam as the two pieces rotate relative to each other. One elastic element is attached to each cable end; each cam surface generates $\tau_{des}/2$. Dark gray: Cam found via optimization to generate $\tau_{des}/2$. The cam is mirrored about $y = -x$ to generate the full τ_{des} during spring deflection in one direction. These two cams are mirrored about x=0 to create a NLS with a symmetric torque-deflection profile about $\Delta\theta$=0.

order Runge-Kutta method, modifying $\Delta\theta_{max}$ to minimize the cost function

$$J(t^*_{rise}) = (t_{rise} - t^*_{rise})^2 \tag{6.8}$$

where t^*_{rise} is the rise time for the current value of $\Delta\theta_{max}$. With an optimized $\Delta\theta_{max}$, equation 6.6 now describes a spring profile that encodes the desired spring shape and rise time. Optimization parameters and results are listed in table 6.1.

6.2.2 Rubber Selection

The NLS design is realized with parallel elastic elements, where each element and cam generate $\tau_{des}/2$ (Fig. 6.2). Rubber is used as the elastic element in the NLS due to its loading properties and customizable form factor. Rubber dimensions must be selected to withstand the uniaxial tension applied by the cam throughout the NLS deflection. The maximum uniaxial tension F_{max} a material can bear without breaking is defined by its cross-sectional area A_o and

$\boldsymbol{J_m}$ (kgm^2)	$\boldsymbol{t^*_{rise}}$ (ms)	$\boldsymbol{\Delta\theta_{max}}$ (^{o})
8.83×10^{-5}	16	53

Table 6.1: NLS optimization parameters & results

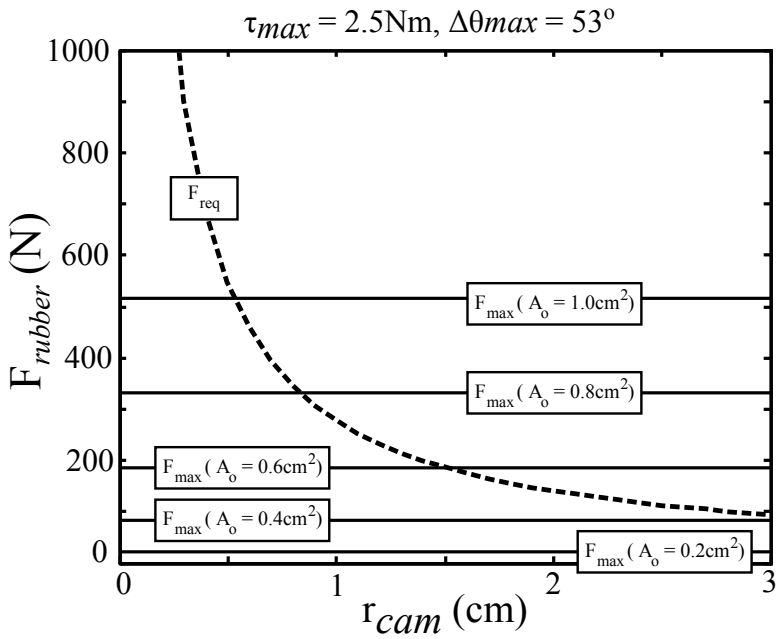

Figure 6.3: PMC-770 minimum required tensile force vs. cam radius and tensile test results. L: Minimum required tensile force vs. cam radius to achieve τ_{max}=2.5 Nm for $\Delta\theta_{max}$=53^o using PMC-770. Dashed: Minimum required tensile force. Solid isocontours: Maximum tensile force for a given A_o.

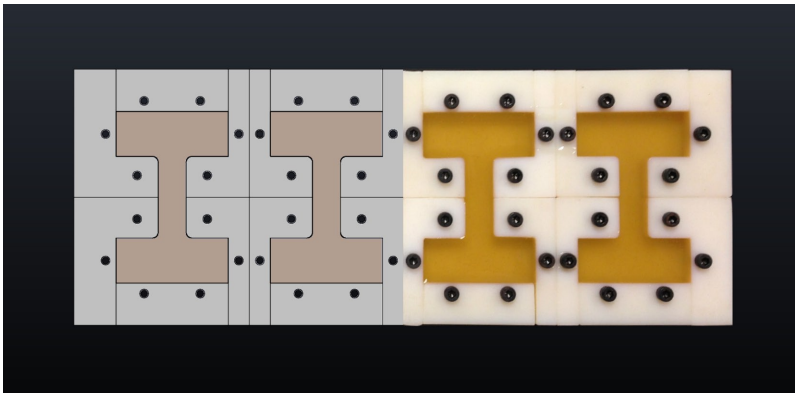

Figure 6.4: Puzzle mold. L: Concept R: Rapid prototyped mold with rubber.

tensile strength σ_f [34]:

$$F_{max} = \sigma_f A_o. \tag{6.9}$$

Given $(r_{spring,x}, r_{spring,y})$, τ_{max}, and $\Delta\theta_{max}$, it is possible to calculate the required uniaxial tension F_{req} at a maximum cam radius $r_{cam,max}$ to realize τ_{max} as

$$F_{req} = \frac{\tau_{max}}{r_{cam,max} sin(\beta)} \tag{6.10}$$

where β is the angle between $\boldsymbol{r_{cam,max}}$ and $\boldsymbol{F_{elastic}}$ as shown in figure 6.1. With this information, it is possible to select an appropriate σ_f to realize a desired NLS size. Urethane rubber (PMC-770: Smooth-On Inc.; σ_f=5.17 MPa) is chosen as the elastic element. Figure 6.3 plots the relationship between F_{req}, $r_{cam,max}$, and PMC-770's F_{max} at various A_o. An $r_{cam,max}$ of 1.7 cm is required for the NLS to fit into the RNL platforms' existing SEAs. To account for small imperfections in the manufactured elastic elements, we use a safety factor of n=3 when choosing A_o. Based on this information and the NLS size requirement, the NLS uses a rubber elastic element with A_o=1 cm^2.

6.2.3 Rubber Manufacturing & Characterization

PMC-770 is a two part urethane rubber mixture. The parts are mixed with a planetary centrifugal mixer (AR-100, THINKY USA, Inc.) and cast into rapid prototyped acrylic molds (VeroWhitePlus, Objet Ltd.). Rubber surface defects create stress points during elongation, which can lead to ripping and premature rubber failure. To mitigate damage to cured rubbers during mold extraction, multi-piece "puzzle molds" are designed, which reduce contact area between the rubber and each mold piece compared to a solid-body mold, and make extraction easier (Fig. 6.4). Rubber samples are cured at room temperature for 24 hours prior to demolding.

The force-deflection relationship of the rubber shows both static and velocity-dependent effects. The dominating static effect, the Mullin's Effect, results in rubber softening after a freshly

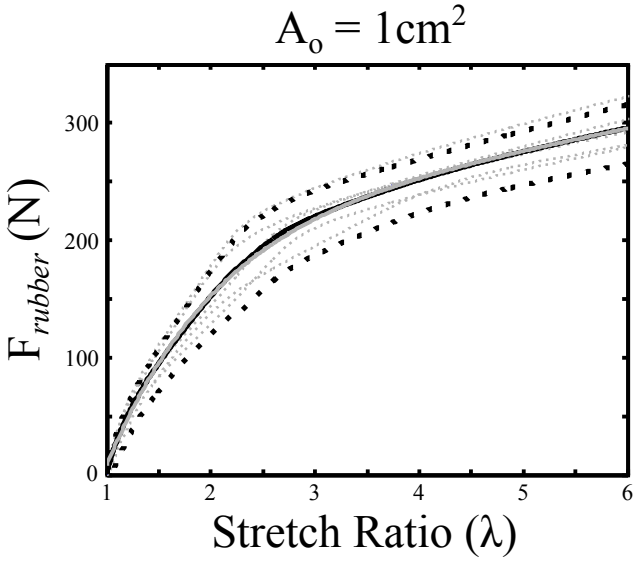

Figure 6.5: R: PMC-770 tensile test results, A_o=1 cm^2. Dotted gray: Individual samples (n=5). Solid black: Average of individual samples. Dotted black: Standard deviation of average. Solid gray: Exponential fit of average: $F_{rubber}(\lambda) = 212.1^{0.975\lambda} - 552.5^{-0.938\lambda}$. R^2=0.9996.

cast sample is stretched to a certain length [6]. This phenomenon occurs as some finite-length polymer chains rupture during initial rubber extension and can no longer resist stretch during subsequent extensions. To account for this effect, samples used to characterize the force-deflection profile of the rubber are pre-stretched past the maximum elongation that will be observed in the NLS prior to tensile testing. Velocity-dependent effects include strain-rate stiffening and hysteresis [33][6]. Both effects are the result of temporary van der Waals bonds forming between polymer chains as they move relative to each other during rubber extension and contraction. Ideally, the rubber's force-deflection profile should be characterized under nominal operating conditions. Due to velocity limits of the load cell available for tensile testing, we characterize the effect of velocity-dependent factors in subsequent NLS experiments.

Tensile tests are performed with an Instron 4400R Load Frame with a 100 lbf analog load cell at an extension rate of 0.5 in (1.27 cm) per minute. 5 samples are stretched until failure. The relationship between force generated in the rubber F_{rubber} and nondimensionalized rubber length

("stretch") λ is given by

$$\lambda = (L_o + \Delta s)/L_o. \tag{6.11}$$

where L_o is the unstretched rubber's rest length. An exponential function of the form $F_{rubber}(\lambda) = a^{b\lambda} + c^{d\lambda}$ is fit to the average of the tensile data using MATLAB's curve-fitting toolbox (MATLAB *cftool*) to obtain a nominal force-stretch relationship for the rubber. The results are shown in figure 6.5. The fit has an R squared value of $R^2 = 0.9996$.

6.2.4 NLS Prototype

The cam shape of the NLS design shown in figure 6.2 to realize the desired torque profile in equation 6.6 using the rubber's empirically characterized force-stretch relationship is found with the optimization procedure described in section 6.1. For the optimization, 11 evenly spaced torque-deflection pairs between $\Delta\theta$=0 and $\Delta\theta_{max}$=53^o are used, with an elastic element rest length L_o=1.25 cm. The resulting cam, as well as the simulated torque-deflection profile realized by the cam, are shown in figures 6.6 and 6.7, respectively. The cam encodes the desired torque profile, with a sum squared error of 0.07 Nm and an average error of 0.07$\pm$0.05 Nm, equivalent to a 1.4% of the maximum target torque.

To test the cam profile experimentally, a NLS prototype is printed out of acrylic. Rubber elastic elements are cast into rapid prototyped molds and glued into retention clamps using urethane adhesive (URE-BOND II; Smooth-On Inc.). Polyethylene cable (Solid Spectra: ϕ=0.5 mm, F_{max}=580 N; BHP Tackle) is used as the cable to interface the cam and rubber elastic element. CAD renderings of NLS components are shown in figure 6.8.

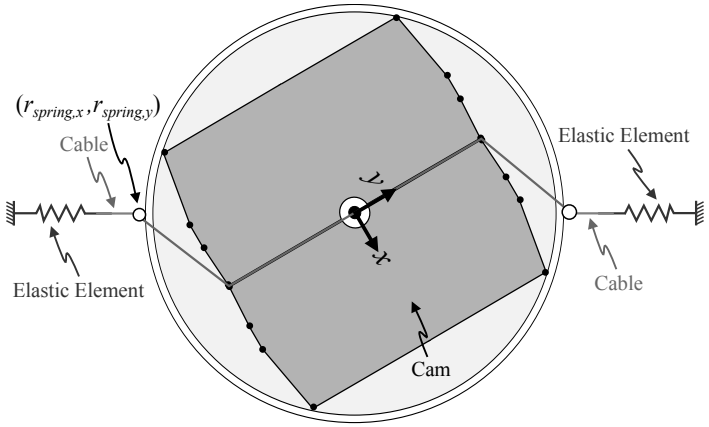

Figure 6.6: NLS prototype schematic with optimized cam to realize desired torque profile when using PMC-770 as the elastic element.

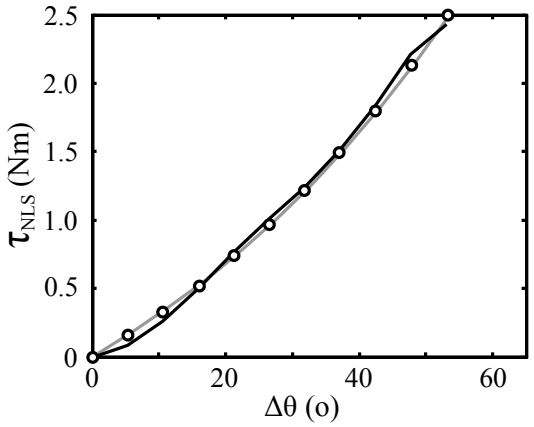

Figure 6.7: Desired vs. optimized cam encoded torque-deflection profile. Gray: Exponential torque profile described by equation 6.6 and parameters in table 6.1. Dots: Discrete torque-deflection pairs used to optimize cam profile. Black: Simulated cam-encoded torque profile.

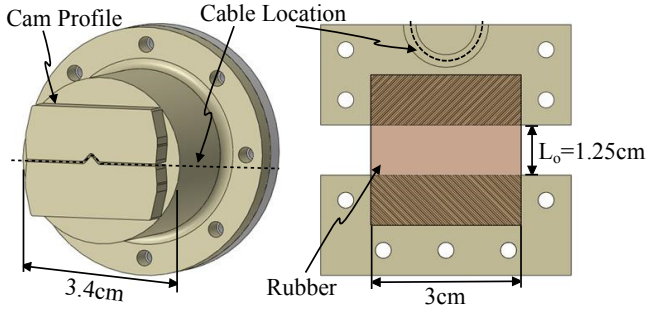

Figure 6.8: NLS components. L: CAD rendering of prototyped cam profile. A divot is placed in the middle of the cam to ensure the cable does not slacken. R: Cross-section of retention clamp. Rubber thickness=3.4 mm. Compression teeth help to secure the rubber during stretching.

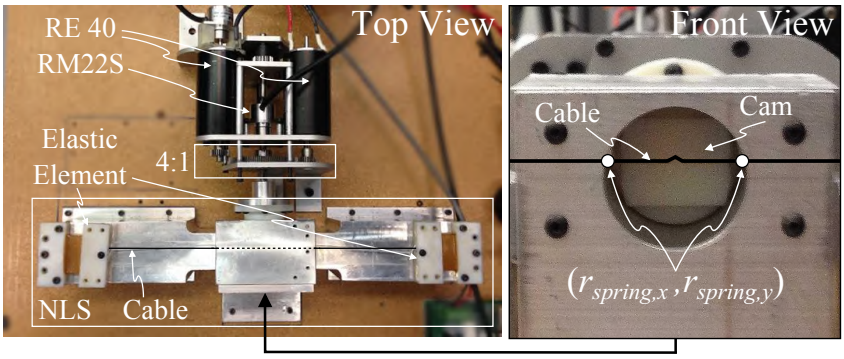

Figure 6.9: NLS benchtop setup. L: Top view of benchtop setup. R: Front view close-up of cam.

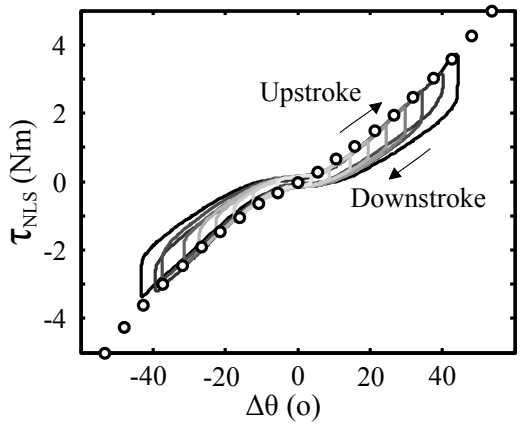

Figure 6.10: Measured NLS profile. Dots: Desired torque profile. Solid isocontours: Average torque profile for 15 deflections.

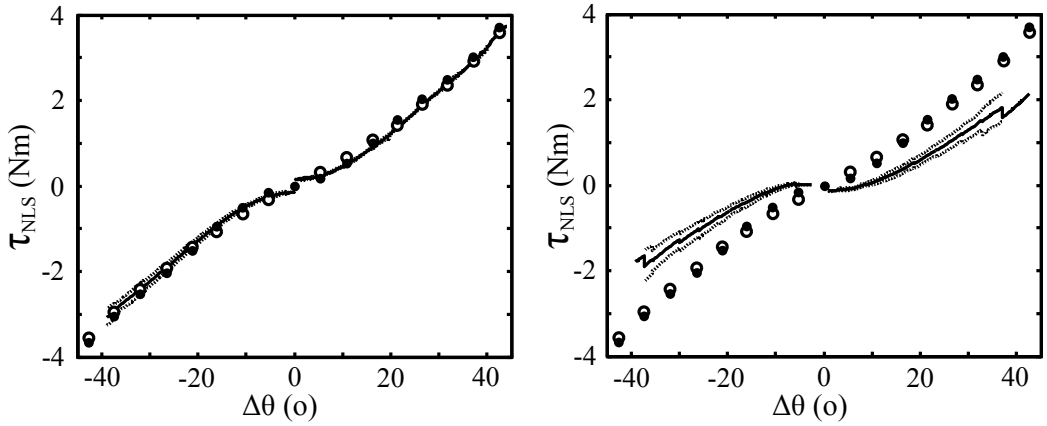

Figure 6.11: Measured NLS profile. L: Upstroke. R: Downstroke. Solid: Average for all deflections (n=150). Dotted: Standard deviation of average. White dots: Desired torque profile. Black dots: Cam-encoded torque profile.

6.3 NLS Characterization

The NLS prototype is evaluated using the benchtop setup shown in figure 6.9. Actuator position is controlled using Mathwork's xPCTarget software and an EtherCAT motor controller (DZEANTU: peak current=20 A; Advanced Motion Controls), which measures applied motor current, from which spring torque is calculated. Spring position is measured with an absolute rotary encoder (RM22S: 13 bit; Renishaw PLC), and fed asynchronously to the control PC using a microcontroller (ATmega328-PU; Atmel Corporation). The NLS is characterized through three experiments. The first experiment validates that the cam profile realizes the desired torque profile. The second experiment tests the NLS' ability to match the desired rise time. Finally, the third experiment investigates velocity-dependent stiffness effects of the rubber elastic element.

6.3.1 Torque Profile Validation

To validate the desired torque profile, the spring is deflected using sinusoidal position commands at a frequency f=0.1 Hz. The low frequency mitigates potential velocity-dependent stiffness effects. 10 sinusoidal position amplitudes are tested with 15 NLS deflections each, where the sinusoid with the largest position amplitude corresponds to the peak current rating of our motor controller. The observed NLS torque-deflection profiles are shown in figure 6.10. Hysteresis is present between cam engagement of the rubber ("upstroke") and cam disengagement of the rubber ("downstroke"). We quantify NLS performance by analyzing the observed torque profile between NLS upstroke and downstroke separately. Figure 6.11 plots the average of observed up- and downstroke profiles for all deflections (n=150).

During upstroke, the observed torque profile matches both the desired and cam-encoded torque profile. The average root-mean-squared error between the desired and measured torque profile is 0.10 Nm, a relative error of 10%, equivalent to 2% of maximum target torque. Ideally, no error should exist between the cam-encoded and measured profile. The average root-mean-squared error between the cam-encoded and measured torque profile is 0.11 Nm, a relative error

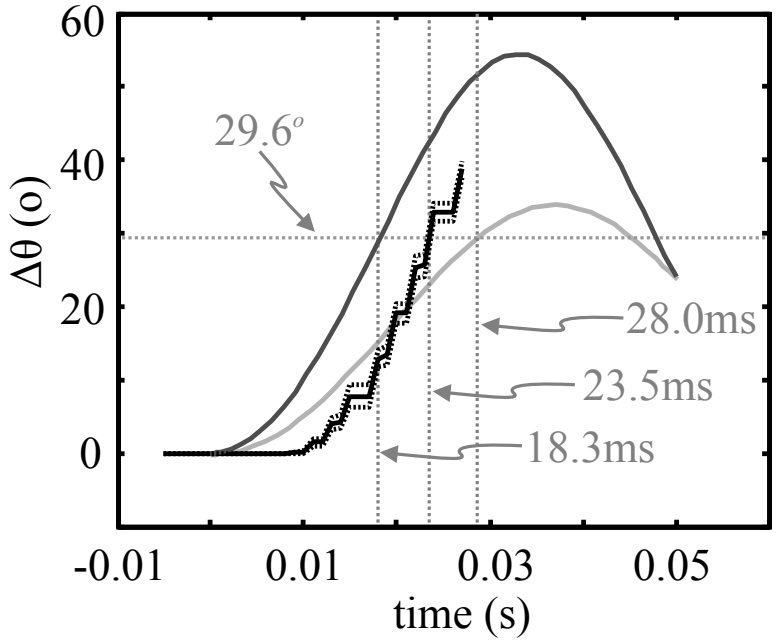

Figure 6.12: NLS step response. Solid black: Average step response (n=10). Dotted black: Standard deviation of average. Dark gray: Step response of ideal model. Light gray: Step response of augmented model.

of 12%, equivalent to 2.2% of maximum target torque. Error results from rubber variability between samples. As shown in figure 6.3, intersample variation in the force-stretch relationship between rubber samples can be as much as 15%.

During downstroke, hysteretic effects of the rubber result in larger discrepancies between the desired and measured torque profile. Downstroke hysteresis is quantified as the average relative error between desired and measured torque. The average relative error between the desired and measured torque is 61%. The average relative error between the cam-encoded and measured torque is 63%. Compared to other rubbers, the NLS' urethane rubber has large hysteresis [116]. To decrease NLS hysteresis, other rubbers could be used as NLS elastic elements.

6.3.2 NLS Step Response

The NLS' ability to match the desired actuator rise time is tested next. For monotonic springs, the correspondence between spring deflection and applied torque is 1:1. The NLS was designed

to generate torques up to 5 Nm, corresponding to a motor current of 22.7 A, which exceeds the 20 A peak current rating of the DZE motor controller. Additionally, applying current step changes greater than 10 A caused the polyethylene cable to snap. This is a shortcoming of the cable, not the NLS design or rubber. A 10 A step response is therefore commanded to the SEA, which corresponds to τ_{NLS}=2.2 Nm at $\Delta\theta$=29.6^o. The time required by the NLS prototype to achieve this deflection is compared with the theoretical rise time predicted by equation 6.7. For this torque, the predicted rise time is 18.3 ms, whereas the measured rise time is 23.5 ms, a relative error of 28% (Fig. 6.12). The measured response also indicates that slight strain-rate stiffening effects are present in the rubber, seen by the measured response's steeper slope compared to the predicted response. These discrepancies result from the fact that equation 6.7 used to optimize $\Delta\theta_{max}$ is an ideal model that does not account for motor and gearing inefficiencies, unmodeled SEA inertia, or rubber strain-rate stiffening effects. Augmenting the model to use better SEA inertia estimates, and motor and gearing efficiencies, η_{motor}=57% and η_{gears}=96%, respectively [87][170], the predicted rise time increases to 28 ms, which is longer than the measured 23.5 ms rise time (Fig. 6.12). This faster than predicted rise-time is beneficial for high bandwidth control and again indicates a stiffer than predicted elastic element resulting from velocity-based strain-rate stiffening effects in the rubber.

6.3.3 Velocity-Dependent Stiffness Effects

To observe velocity-dependent effects, the spring is deflected at three different frequencies f=0.1 Hz, f=0.5 Hz, and f=1.0 Hz. Higher frequencies could not be tested due to bandwidth limitations of the motor controller's position loop. Resulting torque profiles are shown in figure 6.13. The experiments indicate no significant velocity-dependent effects on the spring profile when actuating the NLS at these frequencies. Since step response experiments do suggest the presence of stiffening effects when actuating at high frequencies, future experiments, with a higher bandwidth motor controller, need to be conducted to quantify the relationship between

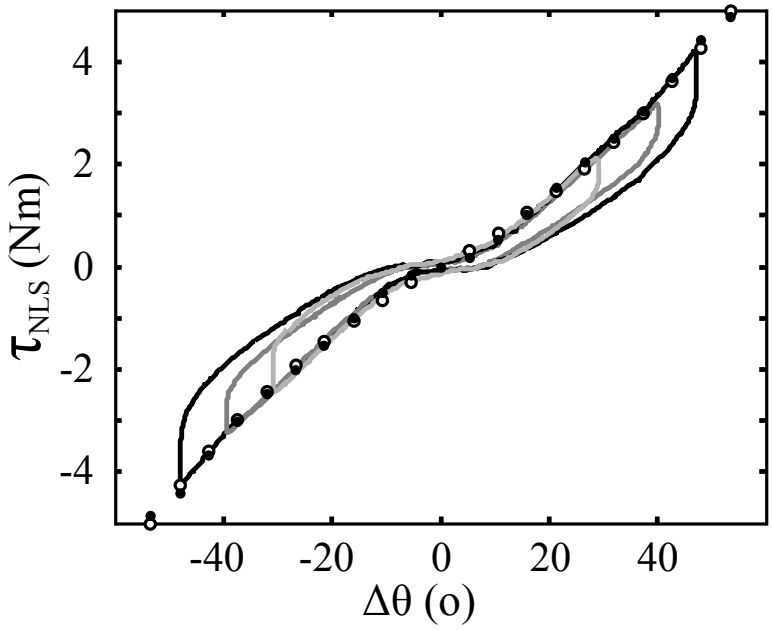

Figure 6.13: NLS profile velocity-effects. Black: f=0.1 Hz. Dark gray: f=0.5 Hz. Light gray: f=1.0 Hz.

torque profile and actuation speed. This shortcoming is addressed in the following sections.

6.4 State-Observer Development

Experiments in the previous sections showed that while the nonlinear spring prototype successfully encodes the desired torque-deflection profile during spring upstroke, significant hysteretic effects of the rubber compromise spring performance. Unaccounted for, these behaviors degrade system stability and closed-loop torque control. Since these behaviors are nonlinear and time-dependent, it is infeasible to account for them using simple pre-computed approaches, like look-up tables. Previous work showed that state observers, which account for nonlinear, time-dependent behavior online, can be used to account for hysteresis of viscoelastic materials [94]. This makes them an attractive candidate for improving the performance of our developed NLS prototype. In this section, a rubber model and corresponding state-observer are developed for a single type of urethane rubber to capture the materials viscoelastic behavior in order to provide an accurate estimate of actuator torque. For experiments conducted in this section, the actuator

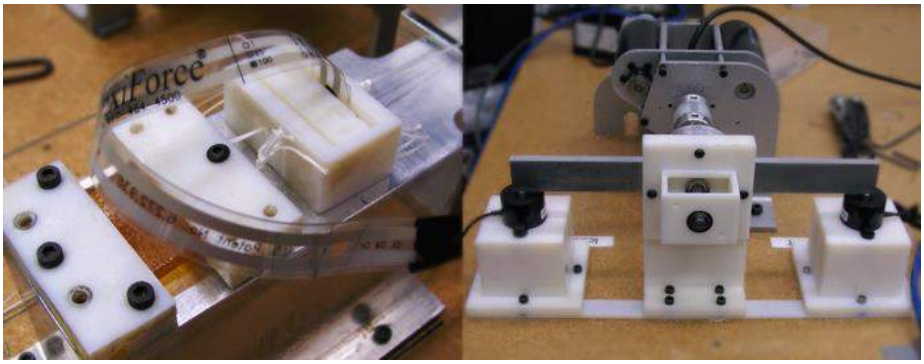

Figure 6.14: Nonlinear spring actuator testbed used for observer development and subsequent experiments. Left: close up of in-series load cell used for ground truth measurement of force in elastic element. Right: Actuator testbed setup for linear spring validation. Two load cells at the ends of a moment arm of known length are used to measure spring torque.

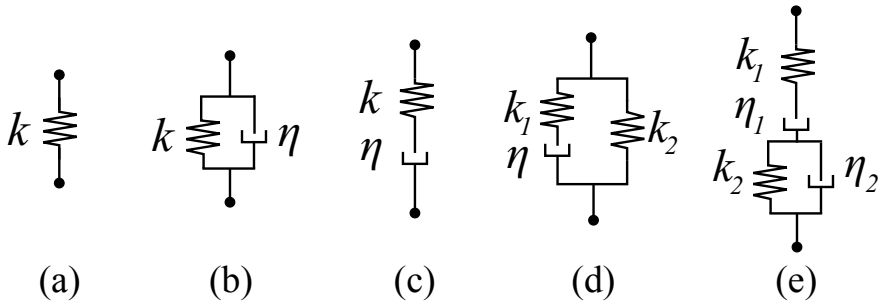

Figure 6.15: Candidate viscoelastic models. Springs and dampers represent elastic and viscous elements, respectively. a) Hooke's Law. b) Kelvin-Voight model. c) Maxwell model. d) Standard Linear Solid model. e) Burger's model.

testbed is modified to incorporate piezoresistive pressure sensors (FlexiForce A201: Tekscan) in series with the rubber to provide a ground truth tension measurement (Fig. 6.14). To develop the presented controller and perform simulation experiments that are not possible in hardware, the vastus actuator simulation model developed in section 4.3.6 is updated to use a model of the nonlinear spring.

6.4.1 Constitutive Rubber Model and Characterization

To account for the rubber's nonlinearity and hysteresis, the state-space observer design must incorporate a model of the rubber that accurately predicts these effects. The state-space observer

models rubber force, with which known geometry is used to directly calculate spring torque, and has three requirements. First, the model must capture the creep, recovery, and stress relaxation behavior seen in experiments. Creep is increasing deflection under a constant force, recovery is a non-instantaneous return to the rest length after the force is removed, and stress relaxation is decreasing force under a constant deflection [74]. Second, the model must be linear in force and deflection, so that it can be used to create a state-space observer to estimate force. Finally, since the model will run in hardware online, it should not have high-order derivatives that are difficult to estimate with discretized, potentially noisy measurements.

Linear visocelastic models are a set of rubber models that satisfy the above criteria [32]. These models represent rubber, a viscoelastic material, as a mechanical system composed of springs as the elastic elements and dampers as the viscous elements (Fig. 6.15), which can be placed in series or parallel to encode various behaviors. In general, a model with a greater number of elements provides increased modeling accuracy, at the expense of mathematical complexity and higher-order derivatives. For example, the Maxwell model encodes stress relaxation, but not creep, whereas the Kelvin-Voigt model encodes creep, but not stress relaxation. Combining these models yields the Standard Linear Solid Model and Burgers model, which encode both stress relaxation and creep. Burgers model contains higher-order derivatives on force and deflection, which results in amplified noise in hardware with discretized data. We therefore choose to use the Standard Linear model for our rubber observer, as it is the simplest model that encodes our desired behaviors.

The constitutive equation of the Standard Linear Model is given by

$$F = k_1 \frac{A_0}{L_0}\delta + \mu \frac{k_1 + k_2}{k_2}\frac{A_0}{L_0}\dot{\delta} - \frac{\mu}{k_2}\dot{F} \tag{6.12}$$

where k_1 and k_2 are stiffnesses, μ is the viscosity, F is the rubber force, and δ is the change in length of the rubber. Note that while rubber models are typically given in terms of stress σ and strain ϵ, we here give all equations in terms of F and δ. This conversion is made using

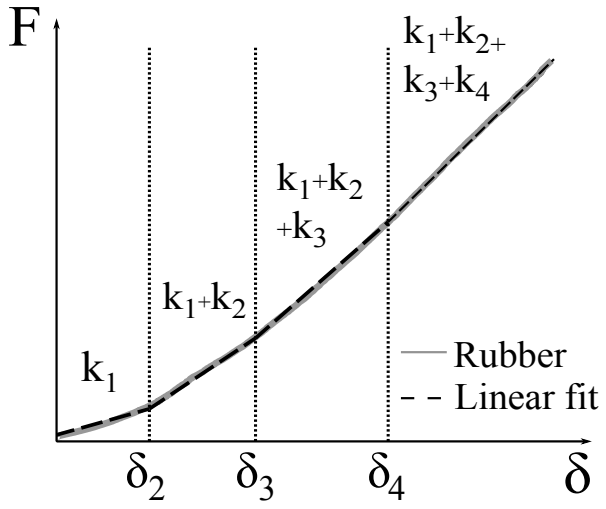

Figure 6.16: Strain-dependent stiffness, which can be approximated by linear stiffnesses.

$\sigma = \frac{F}{A_0}$ and $\epsilon = \frac{\delta}{L_0}$, where A_0 is the cross-sectional area of the rubber and L_0 is the rest length. Each term in equation 6.12 models a desired rubber behavior. The first term essentially encodes Hooke's Law and contributes to elasticity. The second term encodes to creep. The third term encodes stress relaxation.

Data from experiments in the previous section reveals that our rubber exhibits strain-dependent stiffness over the spring's deflection range (Fig. 6.16), which the Standard Linear Solid model, with its constant stiffnesses k_1 and k_2 is unable to capture. However, this behavior can be approximated by dividing the nonlinear force-deflection curve into piecewise linear stiffnesses. The force equation for such a spring can be written as

$$F^* = \frac{A_0}{L_0} \sum_{i}^{n} k_i^* H_{\delta-\delta_i}(\delta - \delta_i) \tag{6.13}$$

where $H_{\delta-\delta_i}$ is the Heaviside step function centered at δ_i. The stiffness k^* changes based on rubber deflection, behaving like n springs in parallel, each of which engages once the rubber stretches past a predefined setpoint. Replacing spring k_2 in the original model with this nonlinear spring and deriving the equations of motion yields

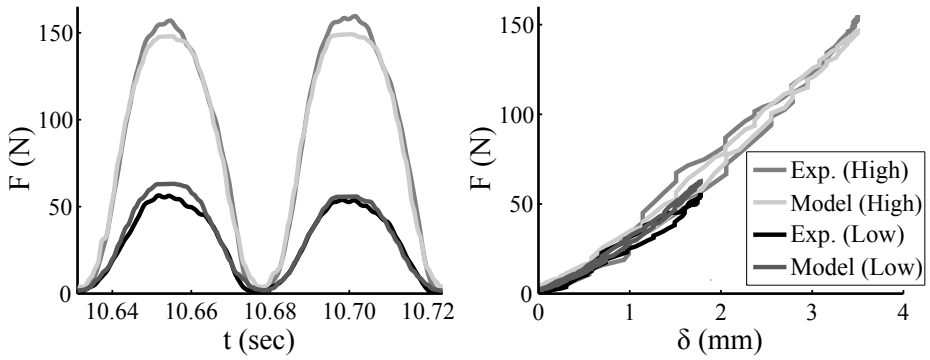

Figure 6.17: Sample fit from Std. Lin. SDS model characterization, which incorporates creep, stress relaxation, and strain-dependent stiffness. Shown fit is from a single actuation cycle at 4 Hz, for both low and high amplitudes. Left: Rubber force vs. time. Right: Force vs. change in rubber length. Jagged artifacts result from encoder discretization.

$$F^* = \frac{A_0}{L_0}\sum_i^n k_i^* H_{\delta-\delta_i}(\delta-\delta_i) + \frac{\mu}{k}\frac{A_0}{L_0}(k+\sum_i^n k_i^*)\dot{\delta} - \frac{\mu}{k}\dot{F} \tag{6.14}$$

which we here define as the Standard Linear Solid model with Strain-Dependent Stiffness (Std. Lin. SDS). This model captures strain-dependent stress while remaining linear in δ.

To fit model parameters, training data sets are collected with the modified actuator testbed. In these experiments, the SEA cam receives sinusoidal position commands with amplitudes of 15^o, 25^o, and 45^o, at frequencies between 0.1-9 Hz, the latter which represents the peak actuation frequency of the dynamically scaled RNL systems. The data sets contain cam position, motor current, and force measurements. The value of δ is calculated based on the cam rotation θ along with known cam geometry, and F is measured directly from the piezoresistive pressure sensors in series with the rubber. The derivatives $\dot{\delta}$ and $\dot{F}$ are calculated using a bi-directional low pass filter. The collected data is used to fit model parameters via stochastic optimization with CMA-ES.

The parameters for the Std. Lin. SDS model are: A_0=1 cm^2, L_0=1.25 cm, η=411 Pa·s, k=0.58 MPa, δ_1=0 mm, k_1^*=0.92 MPa, δ_2=0.20 mm, k_2^*=0.58 MPa, δ_3=1.06 mm, k_3^*=0.74 MPa, δ_4=2.65 mm, k_4^*=0.77 MPa, δ_5=9.04 mm, k_5^*=0.47 MPa, δ_6=9.05 mm, and k_6^*=1.06 MPa. A representative fit against one cycle of characterization data is shown in figure 6.17. Against the

full characterization set, the model achieves an average percent relative error of 13.6%. This is a significant improvement over the relative error when fitting the data using Hooke's Law of 25.2%. This fit corresponds to an RMSE of 0.10Nm at the spring. As a comparison, fitting the SEA testbed with the soft linear springs used in the RNL1 robot, and using Hooke's Law to estimate spring stiffness produces a fit with an RMSE of 0.07 Nm, which results from encoder discretization and noise of the load cells. Therefore, the developed rubber model approaches the limit of what can be achieved with imperfect sensing.

6.4.2 State-Space Equations & Observer Design

The rubber model in equation 6.14 relates the rubber force F to the change in rubber length δ. To further improve estimates of F, it is possible to use the motor dynamics equation, which relates the known motor torque τ_{motor} to the torque developed in the rubber τ_{rubber} based on cam geometry using the equation

$$J\ddot{\theta} = \tau_{motor} - \tau_{rubber} = nk_tI - rFsin\beta \tag{6.15}$$

where I is the motor current, k_t is the torque constant, and n is the gear reduction between the motor and the spring. Combining this with equation 6.14, and noting that the change in rubber length is a function of cam deflection, $\delta = \delta(\theta)$, yields the following state-space equations for the cam:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}(\theta)\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{E}(\theta) \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} \end{aligned} \tag{6.16}$$

where

$$\mathbf{x} = \begin{bmatrix} F \\ \delta \\ \theta \\ \dot{\theta} \end{bmatrix}, \ \mathbf{A}(\theta) = \begin{bmatrix} \frac{-k}{\eta} & \frac{k}{\eta}\frac{A_0}{L_0}\sum_{i=1}^{6} k_i^* H_{\delta(\theta)-\delta_i} & 0 & 0 \\ 0 & 0 & 0 & \frac{d\delta}{d\theta}(\theta) \\ 0 & 0 & 0 & 1 \\ \frac{-r\sin(\beta(\theta))}{J} & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} \frac{A_0}{L_0}\left(k + \sum_{i=1}^{6} k_i^*\right) & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{k_T n}{J} \end{bmatrix}, \ \mathbf{u} = \begin{bmatrix} \dot{\delta} \\ I \end{bmatrix}$$

$$\mathbf{E}(\theta) = \begin{bmatrix} \frac{k}{\eta}\frac{A_0}{L_0}\sum_{i=1}^{6} k_i^* \mathrm{H}_{\delta(\theta)-\delta_i}\delta_i \\ 0 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{C} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Note that $\boldsymbol{E}(\theta)$ is simply a constant offset matrix required to linearly interpolate between step changes in the force for the spring k^*.

Since β and δ are a function of the cam position θ, $\boldsymbol{A}$ and $\boldsymbol{E}$ are not constant. However, their values can be pre-computed for all possible values of θ, which allows for the creation of look-up tables in the observer pre-processing code for $\boldsymbol{A}(\theta)$ and $\boldsymbol{E}(\theta)$ for $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$.

With this information, a Luenberger Observer [2] is implemented to estimate spring force

$$\dot{\hat{x}} = A(\theta)\hat{x} + Bu + L(y - C\hat{x}) + E(\theta) \tag{6.17}$$

where values of the observer gain matrix L are chosen via pole placement. Poles are chosen via optimization with a cost function that minimizes the error between state estimates and actual data gathered during characterization.

The observability matrix $N = [C \quad CA \quad CA^2 \quad CA^3]^T$ is calculated for each value of θ to check for observability. N is full rank for all cases, except when β=π, in which case the rank drops to 3 and the system is unobservable. Pole placement in observable conditions is impossible, which occurs during typical use in this system when the cam passes through θ=0, and is thus unavoidable. In these cases, it is necessary to decompose the system into observable subsystems $\boldsymbol{A_{obs}}$ and $\boldsymbol{C_{obs}}$, and unobservable subsystems $\boldsymbol{A_{unobs}}$ and $\boldsymbol{C_{unobs}}$, using Kalman Decomposition [2]. The observable subsystem is composed of states F, δ, and $\dot{\theta}$. The unobservable subsystem is the state θ and has eigenvalues less than zero, so it is asymptotically stable. Since the unobservable subsystem is stable, it is possible to determine L via pole placement using the following algorithm. When $\theta \neq$0, choose L to satisfy the desired closed-loop poles for $\boldsymbol{A - LC}$. When θ=0, perform pole placement only for the observable subsystem $\boldsymbol{A_{obs} - LC_{obs}}$, leaving the unobservable subsystem poles at zero.

As the state-space matrices and observer gain matrix are a function of the cam rotation θ, the designed observer implements a form of gain scheduling. With the exception of some special cases, no method exists to prove the global stability of a system that implements gain scheduling. Instead, it is necessary to experimentally verify system stability [124]. Simulations that compare observer tracking performance to ground-truth data collected by the load cell reveal that the developed observer is indeed stable and accurately tracks rubber dynamics, achieving a 13.7% average relative error and 0.10 Nm RMSE, the same performance as the rubber model.

6.5 NLS Experimental Evaluation with State-Observation

As stated at the beginning of this chapter, the goal of the NLS is to enable high torque resolution at low torque amplitudes and high bandwidth at high torque amplitudes in a lightweight, compact design. Simulation and hardware experiments in this chapter evaluate these claims using the observer developed in the previous section to account for visocelastic behavior of the rubber. To test the spring's torque resolution, we run output impedance (zero-torque tracking)

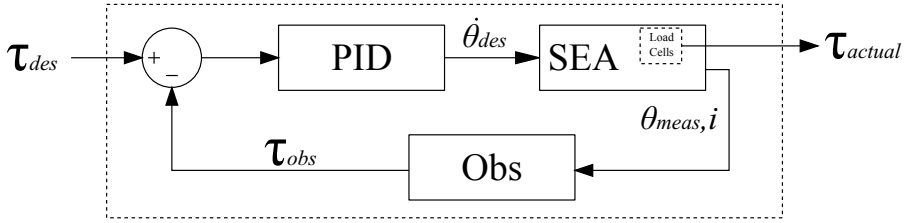

Figure 6.18: Schematic of plant used in spring comparison experiments. The observer provides a torque estimate, which is fed into a PID controller to generate a desired motor velocity. Sensors provide estimates of spring deflection θ and motor current i. Load cells are used to provide ground-truth measurements of torque transmitted by the springs.

experiments and compare the NLS performance versus the performance achieved by the soft linear springs used in the RNL1 testbed. A spring with higher torque resolution will have lower output impedance because the controller has higher-resolution feedback with which to reduce error. To further verify that the NLS behaves like a soft linear spring at low torque amplitudes, its frequency response during low amplitude torque tracking experiments is also analyzed. To test whether the NLS enables an SEA to achieve comparable actuation bandwidth to an SEA that uses a stiff linear spring, we compare the phase margin of the SEA using the NLS prototype and the phase margin of the SEA using the stiff linear springs.

The following experiments are conducted on the benchtop setup described in section 6.4. The plant of the system used for these experiments is shown in figure 6.18. To ensure consistent baseline behavior, the same actuator is used in all experiments. The only variable is whether the experiment uses a soft linear spring, a stiff linear spring, or the NLS. PID gains are tuned for each spring. Gains are first optimized in simulation and later refined on the hardware testbed. As with the characterization experiments, testbed load cells are used to acquire ground-truth force measurements, but these values are not used for feedback or control. For control purposes, Hooke's law is used for the linear springs, and the developed state observer is used for the NLS.

The stiffness of linear springs used for this experiment are limited by the available hardware that is compatible with our actuator. The soft linear spring has a measured stiffness of 1.7 Nm/rad (A5Z26M060:SDP/SI) and the stiff spring has a stiffness of 3.4 Nm/rad (S50TLCM13H060H06:

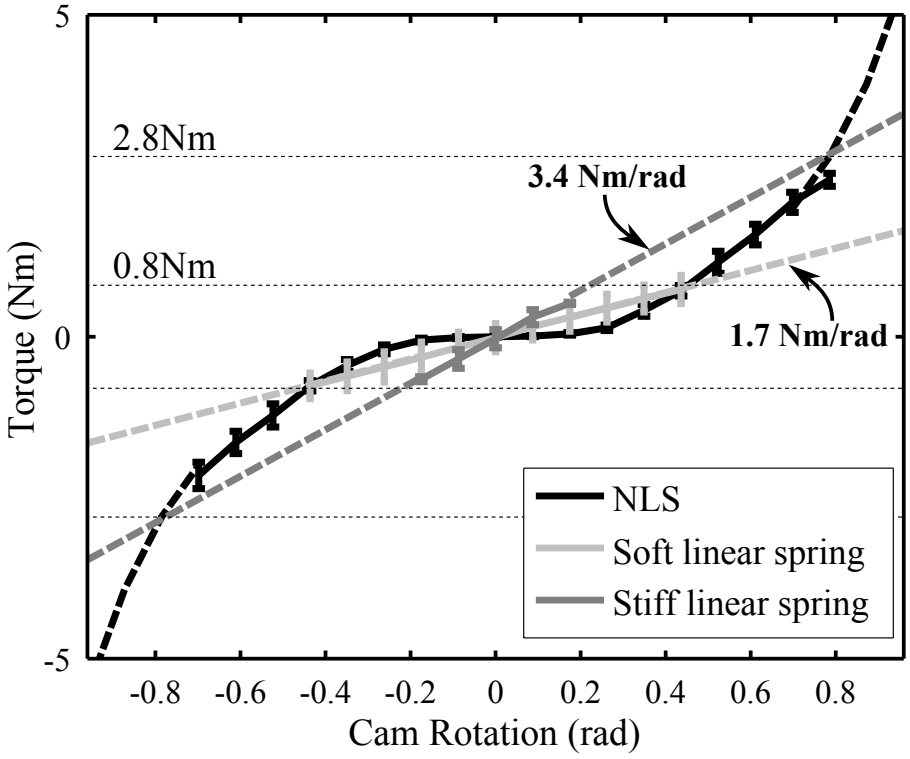

Figure 6.19: Average experimental torque profiles for each spring, with standard deviations. Dashed lines indicate extrapolated data. For comparison purposes, linear spring stiffnesses are chosen for low amplitude (τ_{des}=0.8 Nm) and high-amplitude(τ_{des}=2.8 Nm) experiments via linear fit: $k_{s,soft}$=1.7 Nm/rad, $k_{s,stiff}$=3.4 Nm/rad.

SDP/SI). "Low" and "high" torque amplitudes in these experiments are chosen based on the intersection between these stiffnesses and the NLS torque profile (Fig. 6.19).

For the output impedance experiments, the load end of the actuator is driven with a sinusoidal chirp signal at frequencies between 0.1-10 Hz while the actuator is actively commanded to realize zero torque. Therefore, low output impedance corresponds to torque RMS near zero. Since the load side of our benchtop setup is fixed, output impedance experiments were limited to run using the simulation described in section 6.4. Experimental results are shown in figure 6.20. The NLS achieves lower output impedance than both linear springs up to 9 Hz. This behavior results for two reasons. First, the NLS stiffness is comparable to the soft spring at low amplitudes, resulting in higher torque resolution in this region of operation. Second, the rubber has better shock tolerance to load movement due to its viscous properties.

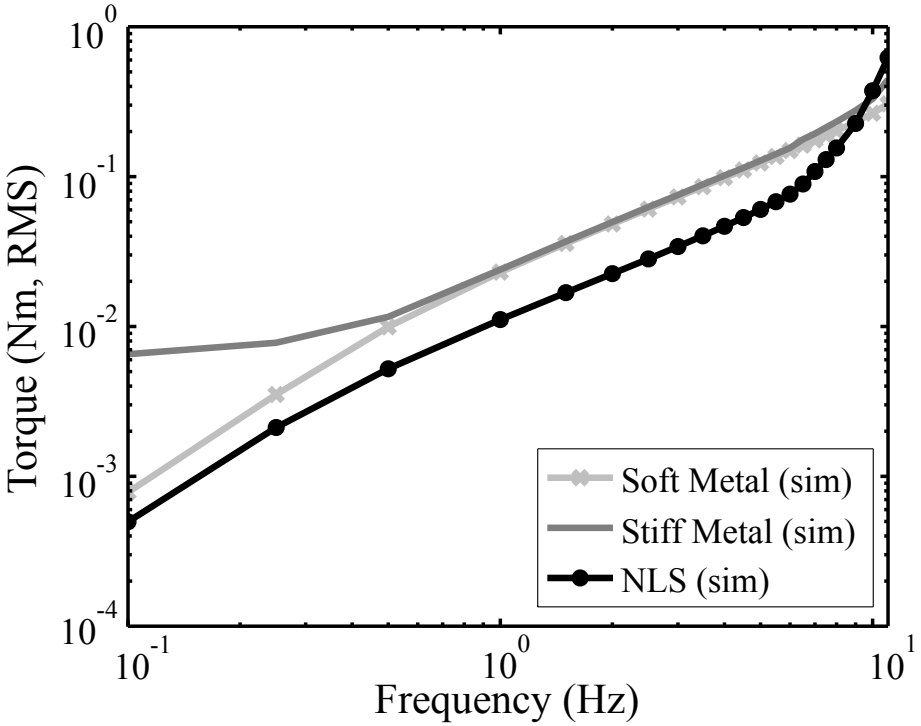

Figure 6.20: Simulated output impedance (zero-torque tracking) response. The load is moved in sinusoids of varying frequency, while the controller attempts to maintain zero torque.

For low amplitude torque tracking experiments, the SEA is commanded to follow a sinusoidal chirp signal at frequencies between 0.1-11 Hz with a torque amplitude of 0.8 Nm. The frequency response for the NLS and soft linear springs for these frequencies are shown in figure 6.21. The results show that the phase and magnitude of the soft linear spring and NLS is comparable up to 1.5 Hz. Beyond 2 Hz, the NLS does not track as well as the soft spring. This performance degradation is not due to poor tracking by the observer; instead, the observed overshoot comes from integrator windup due to a deadzone around zero torque. This deadzone comes primarily from stress relaxation in the rubber – modeled with the $\frac{\eta}{k}\dot{F}$ term in equation 6.14 – which results in a resistance to sudden force changes. Rubbers with higher viscosity η exhibit greater stress relaxation, and since this term is based on the rubber's velocity, this effect has a greater influence at high frequency. Stress relaxation is apparent even in the open-loop characterization data (Fig. 6.22), where a higher torque amplitude at the same actuation frequency results in faster rubber deflection, and thus a wider deadzone around zero torque. Based on these results, it is possible

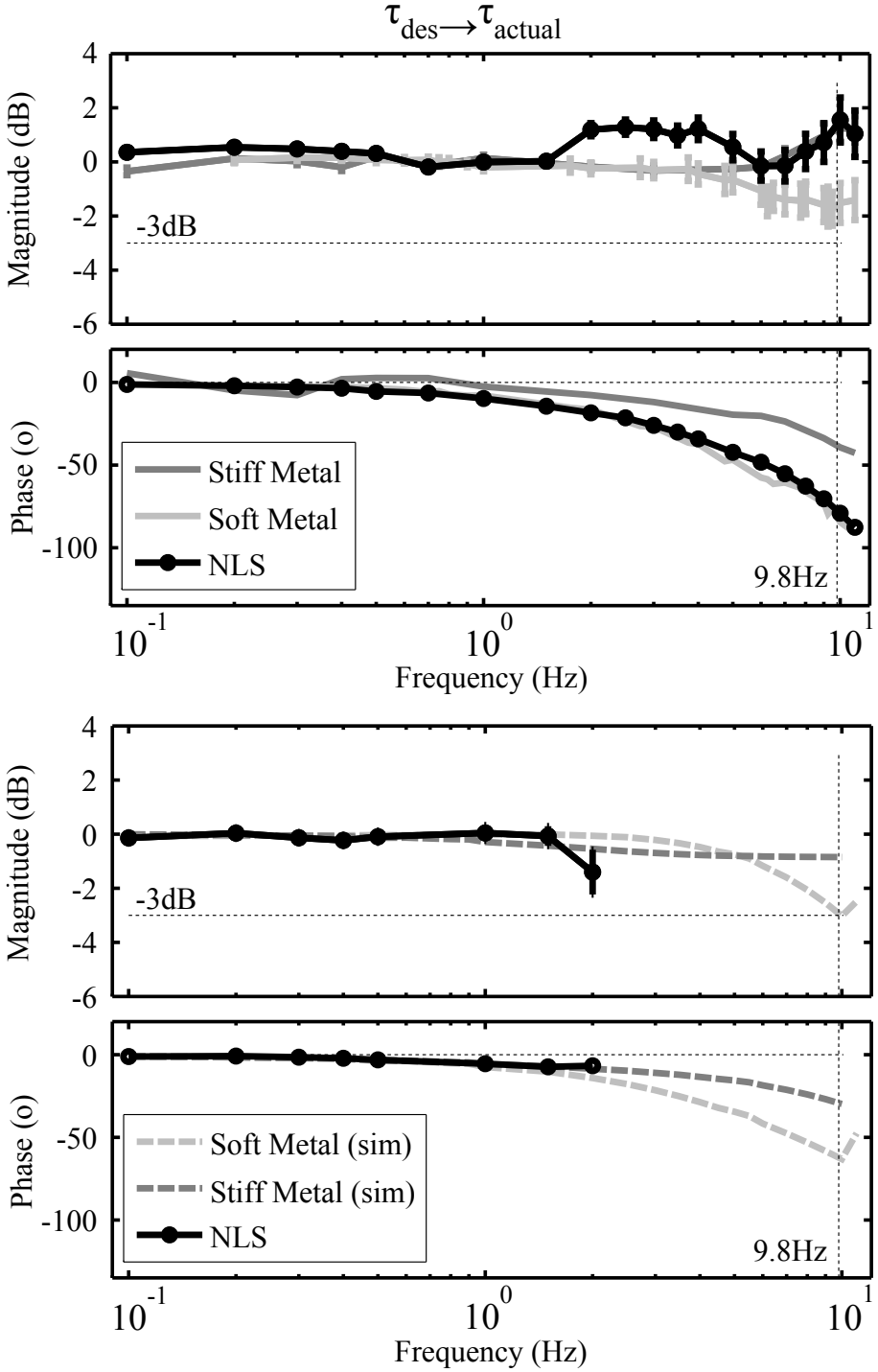

Figure 6.21: Frequency response for NLS SEA. Top: Low-amplitude hardware experiments (τ_{des}=0.8 Nm). Bottom: High-amplitude hardware NLS experiments, with simulated linear spring performance (τ_{des}=2.8 Nm).

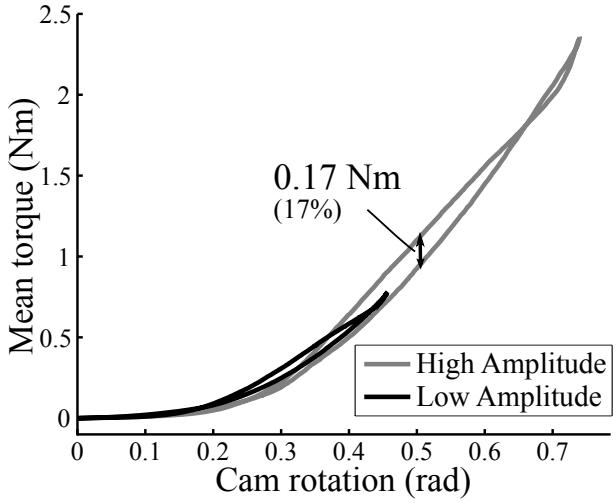

Figure 6.22: Average torque-deflection profile of NLS prototype measured using in-series load cells. Trial data was generate by commanding the motor to follow an open-loop, sinusoidal position chirp signal between 0.1-9 Hz. Chirp signal amplitude for low torque trace: $\Delta\theta$=25^o (n_{cycles}=354). Chirp signal amplitude for high torque trace: $\Delta\theta$=45^o (n_{cycles}=222).

to conclude that the NLS SEA displays comparable torque resolution to the soft linear spring up to 1.5 Hz. Torque tracking degrades at higher actuation frequencies due to rubber hysteresis.

To test whether the NLS exhibits the same bandwidth as the stiff linear spring at high torque amplitudes, the phase margins of the NLS and stiff linear spring SEAs are compared when these systems track a sinusoidal chirp signal between 0.1-11 Hz with a torque amplitude of 2.8 Nm. For these tests, no linear springs exist that are compatible with our actuator design and have the necessary torque rating, so linear spring experiments were simulated. Up to 1.5 Hz, the NLS has the same phase margin as the stiff linear spring. Beyond 2 Hz, the system with the NLS is unstable. Again, observer performance remained stable; the instability results from stress relaxation due to high rubber viscosity, which is exacerbated at high rubber velocities.

Further experiments showed that it is possible to achieve stable performance with the NLS prototype up to 11 Hz if a less aggressive integrator gain is used. However, this degrades system performance compared to the linear springs. Therefore, in order to achieve the original NLS goals, a rubber with lower hysteresis and comparable stiffness must be used.

6.6 Summary

This chapter presented a methodology to design compact, nonlinear springs with user-defined torque-deflection profiles for series elastic actuators. Through optimization it is possible to realize compact spring designs that use rubber as their torque-transmitting element. While the resulting design is smaller than state-of-the-art nonlinear SEA springs and the developed NLS accurately encodes desired torque-deflection profiles during spring upstroke, viscoelastic behavior of the rubber compromises spring performance. To overcome this limitation, a state-estimator that relies on an empirical rubber model was developed and integrated into the SEA control loop, in order to accurately estimate instantaneous rubber force. Testbed experiments show that with this observer, the NLS exhibits the advantageous behaviors of soft and stiff linear springs up to 2 Hz, demonstrating their applicability in SEA designs where size and weight matters. While these results are promising, instabilities above 2 Hz also illustrate the need to carefully select rubbers with low hysteresis during the spring design process in order to enable springs to meet their actuation bandwidth targets.

Chapter 7

Conclusion & Future Work

This thesis presented work to transfer decentralized neuromuscular control strategies of human locomotion to powered segmented robotic legs. While state-of-the-art robotic locomotion control approaches, like centralized planning and tracking in fully robotic systems and predefined motion pattern replay in prosthetic systems, enable functional steady-state locomotion, they do not enable the dynamism and reactiveness of able-bodied humans. Simulation studies suggested that a recently developed decentralized neuromuscular controller may be a promising alternative control method for both fully robotic systems and powered prostheses, but it was unclear if this control approach could be transferred to multi-segmented robotic legs. Work in this thesis investigated the feasibility of controlling a multi-segmented robotic leg with the proposed neuromuscular control approach, in order to enable powered legged robotic systems to react to locomotion disturbances dynamically and in a human-like way.

The "Robotic Neuromuscular Leg" (RNL) test platforms were designed to evaluate the performance of neuromuscular swing-leg controls on robotic hardware. The robots are multi-segment, cable-driven antagonistically actuated robotic legs with joint compliance, whose size, weight, and actuation capabilities correspond to dynamically scaled human values. Using a model-based design approach, an idealized swing-leg controller with mono-articular actuation, as well as the controller's neuromuscular interpration with bi-articular actuation, was transferred to the RNL robots and evaluated with foot placement experiments. Experiments show that both

controllers enable robust foot placement into ground targets, both when motion is undisturbed, as well as when unknown obstacles are encountered in early-, mid-, and late-swing. Compared to state-of-the-art impedance controls, the presented decentralized controllers are more scalable, requiring only a single set of gains, instead of a motion library, to achieve foot placement over a variety of ground targets. Furthermore, the proposed decentralized controls result in more-human like motions, more closely matching human ankle trajectories traced out during undisturbed swing, and respectively generating an elevating and lowering response to early and late swing disturbances, similar to behavior observed during disturbed human locomotion. Both their performance, as well as their scalability, make the proposed decentralized controllers attractive candidates for controlling locomotion of real-world robots and powered prosthetic devices.

In parallel to this control transfer, a synthesis method for compact nonlinear springs with user-defined torque-deflection profiles was presented to explore methods for improving the performance of series elastic actuators used by the RNL robots. The developed springs are designed via optimization and use rubber as their compliant element. Benchtop experiments revealed that this method can be used to develop compact nonlinear springs, but that additional control was necessary to account for viscoelastic behavior introduced by the rubber. To address this, further work developed an observer using an empirically characterized constitutive rubber model to accurately track force developed in the spring's compliant element. Benchtop experiments showed that with this control method, the developed springs achieved desired behavior at actuation frequencies up to 2 Hz, but also illustrated the importance of material selection when designing these nonlinear springs.

Based on the results of these experiments, several avenues of future work exist.

A corollary to neuromuscular swing-leg controls are neuromuscular stance controls. Single-muscle neuromuscular stance controls to regulate ankle plantar flexion torques have been implemented in a powered prosthesis, generating torque-angle profiles during level-ground walking that qualitatively match those of a biological ankle, and enabling the device to adapt to ground slopes without explicitly sensing the terrain [26]. While these results suggest that neuromuscular

stance controls are well suited for use in powered prosthetic devices, it is unknown if full neuromuscular stance control, with multiple muscles, can be transferred to hardware, and what benefits this control method provides. Simulation studies have shown that a fully neuromuscular locomotion controller is capable of generating a diverse set of behaviors, including speed transitions, slope and stair negotiation, turning, and deliberate obstacle avoidance, which can be transitioned between using a small set of high-level control commands [132]. The RNL platforms could be used to directly implement neuromuscular stance control on robotic hardware and evaluate its performance. Together, stance and swing controls could create a fully neuromuscular locomotion controller for real-world robotic devices that enables a diverse set of human-like behaviors with relatively few gains.

While the nonlinear springs encode the desired torque-deflection profile, and, in conjunction with the observer, achieve the desired behavior for actuation frequencies up to 2 Hz, hysteresis of the urethane rubber used in the nonlinear spring prototypes ultimately limited the spring's performance. Hysteresis results from additives in the urethane, which, while stiffening the rubber and enabling smaller elastic element sizes, also negatively impact spring performance [33][6]. To address these issues and create a set of springs that could be implemented into the series elastic acutators of the RNL robots, the springs would have to be redesigned using an alternative rubber, with fewer additives. Natural rubber has been used to create compact linear rubber springs for series elastic acutator modules, and exhibits very little hysteresis [116]. Due to the lack of additives, natural rubber is not as stiff as urethane rubber. To achieve similar stiffness, it would therefore be necessary to increase the cross-sectional area of future natural rubber elastic elements, which could still meet the required size envelope for integration into RNL series elastic actuators, by increasing elastic element width. In addition to redesigning the spring for this new rubber, additional design work needs to be conducted to assemble the components into a form factor that could be integrated into the robots' current actuators. While individual components of the nonlinear spring prototype are small enough for integration into RNLs' SEAs, experiments presented in this thesis were performed in a benchtop setup, and thus not integrated into a

stand-alone spring mechanism. Additional design work needs to be done to create a standalone nonlinear spring.

Specific to this thesis, three major upgrades should be made to the RNL3 robot before evaluating further controls. First, a functional foot should be designed and integrated into RNL3 that enables the robot to perform stance behavior. Several avenues could be pursued in foot design, including using an off-the-shelf prosthetic foot, or creating a new design that incorporates biologically inspired tendon structures [129]. Second, electromechanical upgrades should be applied to RNL3 that will allow the robot to achieve commanded motor velocity-torque pairs. Without redesigning the robot's series elastic acuators, this could be accomplished by using higher voltage motor controllers, and overdriving the robot's DC motors. To enable steady-state locomotion over long periods of time, this would then require active cooling, which, while adding mechanical complexity to the robot's overall design, has been shown to work well in practice [154]. Finally, force sensors to directly measure force applied by the SEA cables to the joints could be installed in RNL3. While the robot was designed to incorporate such sensors, they were not used in experiments to evaluate neuromuscular swing-leg controls. Using these sensors could provide higher force resolution than what is capable by measuring the actuators' spring deflection.

Bibliography

[1] S.K. Agrawal, S.K. Banala, K. Mankala, V. Sangwan, J.P. Scholz, V. Krishnamoorthy, and W.L. Hsu. Exoskeletons for gait assistance and training of the motor impaired. *Proceedings of the IEEE International Conference on Rehabilitation Robotics (IEEE ICORR)*, pages 1108–1113, 2007. 1.1

[2] P.J. Antsaklis and A.N. Michel. *Linear Systems*. Springer, 2006. ISBN 9780817644345. 6.4.2, 6.4.2

[3] S. Aoi and K. Tsuchiya. Stability analysis of a simple walking model driven by an oscillator with a phase reset using sensory feedback. *IEEE Transactions on Robotics*, 22(2): 391–397, 2006. 2.5.3

[4] S.K. Au, H. Herr, J. Weber, and E.C. Martinez-Villalpando. Powered ankle-foot prosthesis for the improvement of amputee ambulation. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBS)*, pages 3020–3026, 2007. doi: 10.1109/IEMBS.2007.4352965. 1.1, 2.6, 4.1

[5] J. Austin, A. Schepelmann, and H. Geyer. Control and evaluation of series elastic actuators with nonlinear rubber springs. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE/RSJ IROS)*, pages 6563–6568, 2015. 4, 6

[6] J.T. Bauman. *Fatigue, Stress, and Strain of Rubber Components: Guide for Design Engineers*. Carl Hanser Verlag GmbH Co KG, 2012. 6, 6.2.3, 7

[7] P. Berkelman, P. Cinquin, J. Troccaz, J. Ayoubi, C. Letoublon, and F. Bouchard. A compact, compliant laparoscopic endoscope manipulator. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, 2:1870–1875, 2002. 3.1.2

[8] J.A. Blaya and H. Herr. Adaptive control of a variable-impedance ankle-foot orthosis to assist drop-foot gait. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 12(1):968–977, 2004. 1.1, 4.1, 5.2

[9] R. Blickhan. The spring-mass model for running and hopping. *Journal of Biomechanics*, 22(11-12):1217–1227, 1989. doi: 10.1016/0021-9290(89)90224-8. 1.1, 2.3, 4.1

[10] A.W. Boehler, K.W. Hollander, T.G. Sugar, and D. Shin. Design, implementation, and test results of a robust control concept for a powered ankle-foot-orthosis (AFO). *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 2025–2030, 2008. doi: 10.1109/ROBOT.2008.4543504. 2.6

[11] T.G. Brown. The intrinsic factors in the act of progression in the mammal. *Proceedings*

of the Royal Society of London. Series B:Biological Sciences, pages 308–319, 1911. 2.5.3

[12] R.C. Browning, E.A. Baker, J.A. Herron, and R. Kram. Effects of obesity and sex on the energetic cost and preferred speed of walking. *Journal of Applied Physiology*, 100(2): 390–398, 2006. doi: 10.1152/japplphysiol.00767.2005. 2.5.4

[13] E. Buckingham. On physically similar systems; Illustrations of the use of dimensional equations. *Physical Review*, 4:345–376, 1914. 3.1.1

[14] G.A Cavagna. Elastic bounce of the body. *Journal of Applied Physiology*, 29(3):279–282, 1970. 2.5.1, 3.1, 3.2.4

[15] G.A. Cavagna, N.C. Heglund, and C.R. Taylor. Mechanical work in terrestrial locomotion: two basic mechanisms for minimizing energy expenditure. *American Journal of Physiology - Regulatory, Integrative, and Comparative Physiology*, 233(5):R243–261, 1977. 2.3, 3.1, 3.2.4

[16] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade. Footstep planning for the Honda Asimo humanoid. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 629–634, 2005. 2.4.1

[17] S.H. Collins and A. Ruina. A bipedal walking robot with efficient and human-like gait. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 1983–1988, 2005. 2.5.2

[18] G. Courtine, Y. Gerasimenko, R. van den Brand, A. Yew, P. Musiekno, H. Zhong, B. Song, Y. Ao, R.M. Ichiyama, I. Lavrov, R.R. Roy, M.V. Sofroniew, and V.R. Edgerton. Transformation of nonfunctional spinal circuits into functional states after the loss of brain input. *Nature Neuroscience*, 12(10):1333–1342, 2009. 2.5.3

[19] J.D. Crisman, C. Kanojia, and I. Zeid. Graspar: A flexible, easily controllable robotic hand. *IEEE Robotics & Automation Magazine*, 3(2):32–38, 1996. 3.1.2

[20] Defense Advanced Research Projects Agency (DARPA). After impressive demonstrations of robot skill, DARPA robotics challenge trials conclude, 2013. URL `http://www.darpa.mil/NewsEvents/Releases/2013/12/26.aspx`. 2.7

[21] R. Desai and H. Geyer. Robust swing leg placement under large disturbances. *Proceedings of the IEEE International Conference on Robotics and Biomimetics (IEEE ROBIO)*, pages 265–270, 2012. doi: 10.1109/ROBIO.2012.6490977. 2.5.4, 3.1, 4, 4.1, 4.2, 4.3.1, 4.3.2, 4.3.7, 4.5.1, 4.5.2, 5, 5.1, 5.1.2

[22] R. Desai and H. Geyer. Muscle-reflex control of robust swing leg placement. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 2169–2174, 2013. doi: 10.1109/ICRA.2013.6630868. 2.5.4, 3.1, 4.1, 5, 5.1

[23] K. Ding, L. Liu, M. van de Panne, and K.K. Yin. Learning reduced-order feedback policies for motion skills. Technical report, TR-2012-06, University of British Columbia, 2012. 2.2

[24] A.E. Dittrich, H. Geyer, and A. Karguth. Obstacle avoidance in a simple hopping robot. *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*, 2006. 2.5, 2.5.1, 4.1

[25] Boston Dynamics. BigDog overview, 2010. Available from the Boston Dynamics YouTube Website. Retrieved from https://www.youtube.com/watch?v=cNZPRsrwumQ. 2.5, 2.5.1

[26] M.F. Eilenberg, H. Geyer, and H. Herr. Control of a powered ankle-foot prosthesis based on a neuromuscular model. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18(2):164–173, 2010. doi: 10.1109/TNSRE.2009.2039620. 1.1, 2.5, 2.6, 2.7, 4.1, 7

[27] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG sensory feedback with policy gradient for biped locomotion for a full-body humanoid. *Proceedings of the National Conference on Artificial Intelligence*, 20(3):1267–1273, 2005. 2.5.3

[28] G. Endo, J. Nakanishi, J. Morimoto, and G. Cheng. Experimental studies of a neural oscillator for biped locomotion with QRIO. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 596–602, 2005. 2.5.3

[29] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. *The International Journal of Robotics Research*, 27(2):213–228, 2008. 2.5.3

[30] J.J. Eng, D.A. Winter, and A.E. Patla. Strategies for recovery from a trip in early and late swing during human walking. *Experiemental Brain Research*, 102(2):339–349, 1994. 4.5.2, 5.5.3

[31] S. Feng, E. Whitman, Xinjilefu X., and C.G. Atkeson. Optimization-based full body control for the DARPA robotics challenge. *Journal of Field Robotics*, 32(2):293–312, 2015. 1.1, 2.4.1

[32] W.N. Findley and F.A. Davis. *Creep and relaxation of nonlinear viscoelastic materials*. Courier Dover Publications, 2011. ISBN 978-0486660165. 6.4.1

[33] A.N. Gent. *Engineering with rubber: How to design rubber components*. Hanser Publications, 2001. 6, 6.2.3, 7

[34] J.M. Gere. *Mechanics of Materials*. Brooks/Cole, 6th edition, 2004. 6.2.2

[35] H. Geyer and H.M. Herr. A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18(3):263–273, 2010. doi: 10.1109/TNSRE.2010.2047592. (document), 1.1, 1.2, 2.5, 2.5.2, 2.5.4, 2.5.4, 2.7, 3.1, 3.1, 3.1, 3.2, 3.1, 3.1.1, 3.3.1, 3.4, 4.1, 4.2.2, 5.1.1, 5.3

[36] H. Geyer, R. Blickhan, and Seyfarth A. Natural dynamics of spring-like running: Emergence of selfstability. *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*, 2002. 2.5.1

[37] H. Geyer, A. Seyfarth, and R. Blickhan. Positive force feedback in bouncing gaits? *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 270: 2173–2183, 2002. doi: 10.1098/rspb.2003.2454. 3.1

[38] H. Geyer, Seyfarth A., and R. Blickhan. Spring-mass running: simple approximate solution and application to gait stability. *Journal of Theoretical Biology*, 232(3):315–328,

2005. 1.1, 4.5.1

[39] H. Geyer, A. Seyfarth, and R. Blickhan. Compliant leg behaviour explains basic dynamics of walking and running. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 273(1603):2861–2867, 2006. doi: 10.1098/rspb.2006.3637. (document), 1.1, 2.1, 2.3

[40] J.W. Grizzle, J. Hurst, B. Morris, H.W. Park, and K. Sreenath. MABEL, a new robotic bipedal walker and runner. *Proccedings of the American Control Conference*, pages 2030–2036, 2009. 3.1.2

[41] M. Günther and H. Ruder. Synthesis of two-dimensional human walking: a test of the λ-model. *Biological Cybernetics*, 89(2):89–106, 2003. 2.5.2

[42] M.W. Hannan and I.D. Walker. Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots. *Journal of Robotic Systems*, 20(2):45–63, 2003. 3.1.2

[43] N. Hansen. CMA-ES MATLAB implementation (Version 3.61) [Computer program]. Available at `https://www.lri.fr/~hansen/cmaes.m`, 2013. 4.3.7, 5.4, 6.1

[44] K. Hase, K. Miyashita, S.Y. Ok, and Y. Arakawa. Human gait simulation with a neuromusculoskeletal model and evolutionary computation. *The Journal of Visualization and Computer Animation*, 14(2):73–92, 2003. 2.5.4

[45] H. Herr. Exoskeletons and orthoses: classification, design challenges and future directions. *Journal of NeuroEngineering and Rehabilitation*, 6:21, 2009. doi: 10.1186/1743-0003-6-21. 2.6

[46] H. Herr. Can bionics make humans stronger, faster and more agile?, 2010. Available from the TEDMED Conference YouTube Website. Retrieved from https://www.youtube.com/watch?v=8AoRmlAZVTs. 2.6

[47] J.H. Hicks. The mechanics of the foot: II. The plantar aponeurosis and the arch. *Journal of Anatomy*, 88(1):25, 1954. 3.4

[48] A.V. Hill. The head of shortening and the dynamic constants of muscle. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 126(843): 136–195, 1938. doi: 10.1098/rspb.1938.0050. 2.5.4, 3.1, 5.1.1

[49] M. Hirose and K. Ogawa. Honda humanoid robots development. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 365(1850):11–19, 2007. 1.1, 2.4, 2.4.1

[50] J. Hitt, A.M. Oymagil, T. Sugar, K. Hollander, Boehler A., and J. Fleeger. Dynamically controlled ankle-foot orthosis (DCO) with regenerative kinematics: Incrementally attaining user portability. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 1541–1546, 2007. 1.1, 2.6, 4.1

[51] N. Hogan. Impedance control: An approach to manipulation. II: Implementation. *Dynamic Systems, Measurement, and Control*, 107(1):8–16, 1985. 5.2

[52] M.A. Holgate, A.W. Boehler, and T. Sugar. Control algorithms for ankle robots: a reflection on the state-of-the-art and presentation of two novel algorithms. *Proceedings of the*

IEEE/RAS EMBS International Conference on Biomedical Robotics and Biomechatronics, pages 97–102, 2008. 2.6

[53] P. Holmes, R.J. Full, D. Koditschek, and J. Guckenheimer. The dynamics of legged locomotion: Models, analyses, and challenges. *Siam Review*, 48(2):207–304, 2006. doi: 10.1137/S0036144504445133. 2.3

[54] S.L. Hooper. *Central Pattern Generators*. John Wiley & Sons, Ltd, 2001. ISBN 9780470015902. doi: 10.1038/npg.els.0000032. 2.5.3

[55] H. Hultborn. Spinal reflexes, mechanisms and concepts: From Eccles to Lundberg and beyond. *Progress in Neurobiology*, 78(3-5):215–232, 2006. 5.1.1

[56] K. Hunt and F. Crossley. Coefficient of restitution interpreted as damping in vibroimpact. *Journal of Applied Mechanics*, 42(2):440–445, 1974. doi: 10.1115/1.3423596. 4.2.2

[57] J.W. Hurst, J.E. Chestnutt, and A.A. Rizzi. An actuator with physically variable stiffness for highly dynamic legged locomotion. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, 5:4662–4667, 2004. 6

[58] A. Ijspeert. Central pattern generator for locomotion control in animals and robots: A review. *Neural Networks*, 21(4):642–653, 2008. doi: 10.1109/TRO.2013.2263718. 1.1

[59] R. Ikeura and H. Inooka. Variable impedance control of a robot for cooperation with a human. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 3097–3102, 1995. 5.2

[60] A. Jafari, N.G. Tsagarakis, B. Vanderborght, and D.G. Caldwell. A novel actuator with adjustable stiffness (AwAS). *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IEEE/RSJ IROS)*, pages 4201–4206, 2010. 6

[61] R. Jimenez-Fabian and O. Verlinden. Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons. *Medical Engineering & Physics*, 34(4):397–408, 2012. (document), 2.2, 2.6, 2.6, 5.2

[62] S. Kajita and B. Espiau. Legged robots. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 361–389. Springer, 2008. ISBN 978-3-540-23957-4. doi: 10.1007/978-3-540-30301-5_17. 2.4.1

[63] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain - derivation and application of the linear inverted pendulum mode. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 1405–1411, 1991. 2.3, 2.4.1, 2.4.2, 4.1

[64] S. Kajita, T. Yamaura, and A. Kobayashi. Dynamic walking control of a biped robot along a potential energy conserving orbit. *IEEE Transactions on Robotics and Automation*, 8 (4):431–438, 1992. 2.4.2

[65] S. Kajita, F. Kanehiro, K. Kaneko, Y. Kazuhito, and H. Hirukawa. The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE/RSJ IROS)*, pages 239–246, 2001. 2.3, 2.4.1

[66] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa.

Biped walking pattern generation by using preview control of zero-moment point. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 1620–1626, 2003. 1.1, 2.3, 2.4, 3, 2.4.1, 4.1

[67] S. Kajita, T. Nagasaki, K. Kaneko, and H. Hirukawa. ZMP-based biped running control. *IEEE Robotics and Automation Magazine*, 14(2):63–72, 2007. 1.1, 2.3, 2.4.1

[68] F. Karlsson and A. Persson. Modelling non-linear dynamics of rubber bushings-parameter identification and validation. *M.S. Thesis, Lund University, Sweden*, 2003. 6

[69] H. Kawamoto and Y. Sankai. Power assist system HAL-3 for gait disorder person. *Computers Helping People with Special Needs; Lecture Notes in Computer Science*, 2398(1): 196–203, 2002. 2.6

[70] K.C. Kong, J.B. Bae, and M. Tomizuka. A compact rotary series elastic actuator for human assistive systems. *IEEE/ASME Transactions on Mechatronics*, 17(2):288–297, 2012. 6

[71] B. Koopman, E.H.F. van Asseldonk, and H. van der Kooij. Speed-dependent reference joint trajectory generation for robotic gait support. *The Journal of Biomechanics*, 47(6): 1447–1458, 2014. 2.6

[72] F.B. Krasne and J.J. Wine. Extrinsic modulation of crayfish escape behaviour. *Journal of Experimental Biology*, 63(2):433–450, 1975. 2.5.3

[73] Oregon State University Dynamic Robotics Laboratory. Medulla: A simple embedded EtherCAT slave device using a xMega microcontroller, 2012. URL `https://code.google.com/p/medulla/`. 5.3

[74] R.S. Lakes. *Viscoelastic solids*. CRC Press, 1998. ISBN 9780849396588. 6.4.1

[75] B.E. Lawson. Control methodologies for powered prosthetic interventions in unilateral and bilateral transfemoral amputees. *Ph.D. Thesis, Vanderbilt University*, 1995. 4.1, 5.2

[76] B.E. Lawson, H.A. Varol, F. Sup, and M. Goldfarb. Stumble detection and classification for an intelligent transfemoral prosthesis. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBC)*, pages 511–514, 2010. 4.1, 5.2

[77] C.R. Lee and C.T. Farley. Determinants of the center of mass trajectory in human walking and running. *Journal of Experimental Biology*, 201(21):2935–2944, 1998. 2.3

[78] W.S. Lu and Q.H. Meng. Impedance control with adaptation for robotic manipulators. *IEEE Transactions on Robotics and Automation*, 7(3):408–415, 1991. 5.2

[79] M. MacKay-Lyons. Central pattern generation of locomotion: A review of the evidence. *Physical Therapy*, 82(1):69–83, 2002. 2.5.3

[80] P. Manoonpong, T. Geng, B. Porr, and F. Worgotter. The RunBot architecture for adaptive, fast, dynamic walking. *IEEE International Symposium on Circuits and Systems (IEEE ISCAS)*, pages 1181–1184, 2007. 2.5.3

[81] K. Matsuoka. Mechanisms of frequency and pattern control in the neural rhythm generators. *Biological Cybernetics*, 56(5-6):345–353, 1987. 2.5.3

[82] T. McMahon. Mechanics of locomotion. *The International Journal of Robotics Research*

(IJRR), 3(2):4–28, 1984. doi: 10.1177/027836498400300202. 2.3

[83] T. McMahon and G. Cheng. The mechanics of running: How does stiffness couple with speed? *Journal of Biomechanics*, 23(S:1):65–78, 1990. doi: 10.1016/0021-9290(90)90042-2. 1.1

[84] S.A. Migliore, E.A. Brown, and S.P. DeWeerth. Biologically inspired joint stiffness control. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 4508–4513, 2005. 3.2.4, 6

[85] S.A. Migliore, E.A. Brown, and S.P. DeWeerth. Novel nonlinear elastic actuators for passively controlling robotic joint compliance. *Journal of Mechanical Design*, 129(4):406–412, 2007. 6

[86] S. Miyakoshi, G. Taga, Y.o Kuniyoshi, and A. Nagakubo. Three dimensional bipedal stepping motion using neural oscillators-towards humanoid motion in the real world. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1:84–89, 1998. 2.5.4

[87] Maxon Motor. RE40 Datasheet, 2010. 6.3.2

[88] P. Musienko, R. van den Brand, O. Maerzendorfer, R.R. Roy, Y. Gerasimenko, V.R. Edgerton, and G. Courtine. Controlling specific locomotor behaviors through multidimensional monoaminergic modulation of spinal circuitries. *Journal of Neuroscience*, 31(25):9264–9278, 2011. 2.5.3

[89] P.D. Neuhaus, J.H. Noorden, T.J. Craig, T. Torres, J. Kirschbaum, and J.E. Pratt. Design and evaluation of Mina: A robotic orthosis for paraplegics. *Proceedings of the IEEE International Conference on Rehabilitation Robotics (IEEE ICORR)*, pages 1–8, 2011. 1.1, 4.1

[90] N. Ogihara and N. Yamazaki. Generation of human bipedal locomotion by a bio-mimetic neuro-musculo-skeletal model. *Biological Cybernetics*, 84(1):1–11, 2001. 2.5.4

[91] A.M. Oymagil, J.K. Hitt, T. Sugar, and J. Fleeger. Control of a regenerative braking powered ankle foot orthosis. *Proceedings of the IEEE International Conference on Rehabilitation Robotics (IEEE ICORR)*, pages 28–34, 2007. doi: 10.1109/ICORR.2007.4428402. 2.6

[92] D.J. Paluska. Design of a humanoid biped for walking research. *M.S. Thesis, Massachusetts Institute of Technology*, 2000. 3.1.2

[93] M.G. Pandy. Simple and complex models for studying muscle function in walking. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358 (1437):1501–1509, 2003. doi: 10.1098/rstb.2003.1338. 2.3

[94] F. Parietti and H. Geyer. Reactive balance control in walking based on a bipedal linear inverted pendulum model. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 5442–5447, 2011. 2.4.2, 6.4

[95] J.H. Park. Impedance control for biped robot locomotion. *IEEE Transactions on Robotics and Automation*, 17(6):870–882, 2001. 5.2

[96] Renishaw PLC. The accuracy of angle encoders. *Renishaw PLC Technical Documenta-*

tion, 2009. 4.4

[97] G.A. Pratt and M.M. Williamson. Series elastic actuators. *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IEEE/RSJ IROS)*, pages 399–406, 1995. 3.1.2, 3.2.3, 3.2.3, 3.2.4, 6

[98] J. Pratt. Virtual model control of a biped walking robot. *M.S. Thesis, Massachusetts Institute of Technology*, 1995. 2.4.3, 2.4.3

[99] J. Pratt, P. Dilworth, and G. Pratt. Virtual model control of a bipedal walking robot. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 193–198, 1997. 2.4.3, 2.4.3

[100] J. Pratt, C.M. Chew, A. Torres, P. Dilworth, and G. Pratt. Virtual model control: An intuitive approach for bipedal locomotion. *The International Journal of Robotics Research (IJRR)*, 20(2):129–143, 2001. 2.4.3

[101] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture point: A step toward humanoid push recovery. *Proceedings of the IEEE/RSJ International Conference on Humanoid Robots (IEEE/RSJ Humanoids)*, pages 200–207, 2006. 2.3, 2.4.2, 2.4.2, 2.4.2, 4.1

[102] J. Pratt, T. Koolen, T. De Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus. Capturability-based analysis and control of legged locomotion, part 2: Application to M2V2, a lower body humanoid. *The International Journal of Robotics Research (IJRR)*, 2012. 2.4.2, 4.1

[103] J.E. Pratt and G.A. Pratt. Intuitive control of a planar bipedal walking robot. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 2014–2021, 1998. 2.4.3, 2.5.2, 3.1.2

[104] J.E. Pratt and R. Tedrake. Velocity-based stability margins for fast bipedal walking. In *Fast Motions in Biomechanics and Robotics*, pages 299–324. Springer, 2006. 2.4.2

[105] J.E. Pratt, B. Krupp, and C. Morse. Series elastic actuators for high fidelity force control. *Industrial Robot: An International Journal*, 29(3):234–241, 2002. 6

[106] M. Raibert. Hopping in legged systems - modeling and simulation for the two dimensional one-legged case. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(3):451–463, 1984. doi: 10.1109/TSMC.1984.6313238. 2.5, 2.5.1, 4.1

[107] M. Raibert. Legged robots that balance. *MIT Press*, Cambridge, 1986. 2.5.1, 2.7, 4.1

[108] M. Raibert, B.H. Brown, M. Chepponis, E.F. Hastings, S.E. Shreve, and F.C. Wimberly. Dynamically stable legged locomotion: Annual technical report to DARPA. *Carnegie Mellon University Technical Report*, CMU-RI-TR-81-09, 1981. 2.5.1

[109] M. Raibert, B.H. Brown, M. Chepponis, E.F. Hastings, S. Murthy, and F.C. Wimberly. Dynamically stable legged locomotion: second report to DARPA. *Carnegie Mellon University Technical Report*, CMU-RI-TR-83-1, 1983. 2.3, 2.5, 2.5.1, 2.5.2

[110] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter. BigDog, the rough-terrain quadruped robot. *Proccedings of the International Federation of Automatic Control World Congress*, 17:10822–10825, 2008. doi: 10.3182/20080706-5-KR-1001.01833. 2.5, 2.5.1

[111] J. Rebula, F. Canas, J. Pratt, and A. Goswami. Learning capture points for humanoid push recovery. *Proceedings of the IEEE/RSJ International Conference on Humanoid Robots (IEEE/RSJ Humanoids)*, pages 65–72, 2007. 2.4.2

[112] M.S. Redfern and T. Schumann. A model of foot placement during gait. *Journal of Biomechanics*, 27(11):1339–1346, 1994. 4.1

[113] T. Reil and P. Husbands. Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6(2): 159–168, 2002. 2.5.3

[114] R.E. Ritzmann and J.M. Camhi. Excitation of leg motor neurons by giant interneurons in the cockroach periplaneta americana. *Journal of Comparative Physiology*, 125(4):305–316, 1978. doi: 10.1007/BF00656865. 2.5.3

[115] D.W. Robinson. Design and analysis of series elasticity in closed-loop actuator force control. *Ph.D. Thesis, Massachusetts Institute of Technology*, 1996. 3.1.2, 6

[116] D. Rollinson, S. Ford, B. Brown, and H. Choset. Design and modeling of a series elastic element for snake robots. *ASME Dynamic Systems and Control Conference*, pages V001T08A002–V001T08A002, 2013. 6, 6.3.1, 7

[117] A. Schepelmann, M.D. Taylor, and H. Geyer. Development of a testbed for robotic neuromuscular controllers. *Proceedings of Robotics: Science and Systems*, VIII:385–392, 2012. 3

[118] A. Schepelmann, K.A. Geberth, and H. Geyer. Compact nonlinear springs with user defined torque-deflection profiles for series elastic actuators. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 3411–3416, 2014. doi: 10.1109/Schepelmann. 6

[119] A. Schepelmann, J. Austin, and H. Geyer. Evaluation of decentralized reactive swing-leg control on a powered robotic leg. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE/RSJ IROS)*, pages 381–386, 2015. 3

[120] N. Schmit and M. Okada. Design and realization of a non-circular cable spool to synthesize a nonlinear rotational spring. *Advanced Robotics*, 26(3-4):234–251, 2012. 6

[121] N. Schmit and M. Okada. Optimal design of nonlinear springs in robot mechanism: simultaneous design of trajectory and spring force profiles. *Advanced Robotics*, 27(1):33–46, 2013. 6

[122] A. Seyfarth and H. Geyer. Natural control of spring-like running: optimised selfstabilisation. *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*, 2002. 2.5.1

[123] A. Seyfarth, H. Geyer, and H. Herr. Swing-leg retraction: a simple model for stable running. *Journal of Experimental Biology*, 206(15):2547–2555, 2003. doi: 10.1242/jeb.00463. 2.5, 2.5.1, 4.1

[124] J.S. Shamma and M. Athans. Gain scheduling: Potential hazards and potential remedies. *IEEE Control Systems Magazine*, 12(3):101–107, 1992. 6.4.2

[125] J. Shan and F. Nagashima. Neural locomotion controller design and implementation for

humanoid robot HOAP-1. *Proceedings of the Annual Conference of the Robotics Society of Japan*, 20, 2002. 2.5.3

[126] C.S. Sherrington. On reciprocal innervation of antagonistic muscles. Third Note. *Proceedings of the Royal Society of London*, 60(359-367):414–417, 1896. 2.5.4

[127] M.L. Shik, F.V. Severin, and G.N. Orlovski. Control of walking and running by means of electric stimulation of the midbrain. *Biofizika*, 11(4):659–666, 1966. 2.5.3

[128] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009. ISBN 9781846286414. doi: 10.1007/978-1-84628-642-1. 2.4, 4.3.6

[129] S. Song and H. Geyer. The energetic cost of adaptive feet in walking. *Proceedings of the IEEE International Conference on Robotics and Biomimetics (IEEE ROBIO)*, pages 1597–1602, 2011. 3.4, 7

[130] S. Song and H. Geyer. Regulating speed and generating large speed transitions in a neuromuscular human walking model. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 511–516, 2012. doi: 10.1109/ICRA.2012.6225307. 2.5.4, 3.1

[131] S. Song and H. Geyer. Generalization of a muscle-reflex control model to 3D walking. *Proceedings of the IEEE Engineering in Medicine and Biology Society (IEEE EMBS)*, pages 7463–7466, 2013. doi: 10.1109/EMBC.2013.6611284. 2.5.4, 3.1

[132] S. Song and H. Geyer. A neural circuitry that emphasizes spinal feedback generates diverse behaviours of human locomotion. *Journal of Physiology*, 593(16):3493–3511, 2015. doi: 10.1113/JP270228. 7

[133] S. Song, R. Desai, and H. Geyer. Integration of an adaptive swing control into a neuromuscular human walking model. *Proceedings of the IEEE Engineering in Medicine and Biology Society (IEEE EMBS)*, pages 4915–4918, 2013. doi: 10.1109/EMBC.2013.6610650. 1.1, 2.5.4, 2.7, 3.1, 4.1, 4.5.1

[134] A. Sproewitz, R. Moeckel, J. Maye, and A.J. Ijspeert. Learning to move in modular robots using central pattern generators and online optimization. *The International Journal of Robotics Research*, 27(3-4):423–443, 2008. doi: 10.1177/0278364907088401. 2.5.3

[135] B. Stephens. Integral control of humanoid balance. *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IEEE/RSJ IROS)*, pages 4020–4027, 2007. 2.4.2

[136] B. Stephens. Humanoid push recovery. *Proceedings of the IEEE/RAS International Conference on Humanoid Robots (IEEE/RAS Humanoids)*, pages 589–595, 2007. 2.4.2

[137] B. Stephens. Push recovery control for force-controlled humanoid robots. *Ph.D. Thesis, Carnegie Mellon University*, 2011. 2.2, 2.4.2

[138] F. Sup, A. Bohara, and M. Goldfarb. Design and control of a powered knee and ankle prosthesis. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 4134–4139, 2007. (document), 5.5, 5.2, 5.2

[139] F. Sup, A. Bohara, and M. Goldfarb. Design and control of a powered transfemoral pros-

thesis. *International Journal of Robotics Research (IJRR)*, 27(2):263–273, 2008. (document), 1.1, 2.7, 2.6, 5.2

[140] F. Sup, H.A. Varol, J. Mitchell, T. Withrow, and M. Goldfarb. Design and control of an active electrical knee and ankle prosthesis. *Proceedings of the IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics (IEEE BioRob)*, pages 523–528, 2008. 1.1, 2.6, 5.2

[141] F. Sup, H.A. Varol, and M. Goldfarb. Powered sit-to-stand and assistive stand-to-sit framework for a powered transfemoral prosthesis. *Proceedings of the IEEE International Conference on Rehabilitation Robotics (IEEE ICORR)*, pages 645–651, 2009. 2.6, 5.2

[142] F. Sup, H.A. Varol, and M. Goldfarb. Multiclass real-time intent recognition of a powered lower limb prosthesis. *IEEE Transaction on Biomedical Engineering*, 57(3):542–551, 2010. 2.6, 5.2

[143] F. Sup, H.A. Varol, and M. Goldfarb. Upslope walking with a powered knee and ankle prosthesis: initial results with an amputee subject. *IEEE Transaction on Neural Systems and Rehabilitation Engineering*, 19(1):71–78, 2011. 1.1, 2.6, 2.7, 4.1, 5.2

[144] G. Taga. Emergence of bipedal locomotion through entrainment among the neuro-musculo-skeletal system and the environment. *Physica D: Nonlinear Phenomena*, 75 (1):190–208, 1994. (document), 2.4, 2.5, 2.5.4, 2.5.4

[145] G. Taga. A model of the neuro-musculo-skeletal system for human locomotion. *Biological Cybernetics*, 73(2):97–111, 1995. 2.5.4

[146] G. Taga, Y. Yamaguchi, and H. Shimizu. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics*, 65(3):147–159, 1991. 2.5.3, 2.5.4

[147] M.D. Taylor. A compact series elastic actuator for bipedal robots with human-like dynamic performance. *M.S. Thesis, Carnegie Mellon University*, 2011. 3.2.3

[148] N. Thatte and H. Geyer. Toward balance recovery with leg prostheses using neuromuscular model control. *IEEE Biomedical Engineering*, PP(99), 2015. 5.5.3

[149] G. Tonietti, R. Schiavi, and A. Bicchi. Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, 1:526–531, 2005. 6

[150] M.A. Townsend. Biped gait stabilization via foot placement. *Journal of Biomechanics*, 18 (1):21–38, 1985. 2.4.2, 4.1

[151] N.G. Tsagarakis, M. Laffranchi, B. Vanderborght, and D.G. Caldwell. A compact soft actuator unit for small scale human friendly robots. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 4356–4362, 2009. 6

[152] Defense Advanced Research Projects Agency: Tactical Technology Office (DARPA TTO). DARPA robotics challenge. *Broad Agency Announcement*, DARPA-BAA-12-39, 2012. URL `https://www.fbo.gov/utils/view?id=74d674ab011d5954c7a46b9c21597f30`. 2.7

[153] J. Urata, K. Nshiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba. Online decision

of foot placement using singular LQ preview regulation. *Proceedings of the IEEE/RSJ International Conference on Humanoid Robots (IEEE/RSJ Humanoids)*, pages 13–18, 2011. 2.4.2, 2.4.2, 4.1

[154] J. Urata, K. Nshiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba. Online walking pattern generation for push recovery and minimum delay to commanded change of direction and speed. *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IEEE/RSJ IROS)*, pages 3411–3416, 2012. 2.4.2, 2.4.2, 7

[155] H. Vallery, E.H.F. van Asseldonk, M. Buss, and H. van der Kooij. Reference trajectory generation for rehabilitation robots: Complementary limb motion estimation. *IEEE Transaction on Neural Systems and Rehabilitation Engineering*, 17(1):23–30, 2009. 1.1, 3.2.3

[156] N. van der Noot, A.J. Ijspeert, and R. Ronsse. Biped gait controller for large speed variations, combining reflexes and a central pattern generator in a neuromuscular model. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 6267–6274, 2015. 5.4

[157] R. van Ham, B. Vanderborght, M. van Damme, B. Verrelst, and D. Lefeber. MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot. *Robotics and Autonomous Systems*, 55(10): 761–768, 2007. 6

[158] R. van Ham, T.G. Sugar, B. Vanderborght, K.W. Hollander, and D. Lefeber. Compliant actuator designs. *IEEE Robotics & Automation Magazine*, 16(3):81–94, 2009. 3.2.4

[159] H.A. Varol, F. Sup, and M. Goldfarb. Real-time gait mode intent recognition of a powered knee and ankle prosthesis for standing and walking. *Proceedings of the IEEE/RAS EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 66–72, 2009. doi: 10.1109/BIOROB.2008.4762860. 2.6, 5.2

[160] C.L. Vaughan. Theories of bipedal walking: an odyssey. *Journal of Biomechanics*, 36(4): 513–523, 2003. 2.5.3

[161] J.F. Veneman, R. Ekkelenkamp, R. Kruidhof, F.C.T. van der Helm, and H. van der Kooij. A series elastic-and Bowden-cable-based actuation system for use as torque actuator in exoskeleton-type robots. *The International Journal of Robotics Research (IJRR)*, 25(3): 261–281, 2006. 3.1.2, 6

[162] M. Vukobratović. How to control artificial anthropomorphic systems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(5):497–507, 1973. doi: 10.1109/TSMC.1973.4309277. 1.1, 2.3, 2.4.1

[163] M. Vukobratović, A.A. Frank, and D. Juricic. On the stability of biped locomotion. *IEEE Transactions on Biomedical Engineering*, 17(1):25–36, 1970. doi: 10.1109/TBME.1970.4502681. 2.1, 2.4.1

[164] J.M. Wang, D.J. Fleet, and A. Hertzmann. Optimizing walking controllers. *Association for Computing Machinery Transactions on Graphics (ACM TOG)*, 28(5):168, 2009. 2.5.2

[165] J.M. Wang, S.R. Hamner, S.L. Delp, and V. Koltun. Optimizing locomotion controllers using biologically-based actuators and objectives. *Association for Computing Machinery*

Transactions on Graphics (ACM TOG), 31(4):25, 2012. 2.5.2

[166] L. Wang, S. Wang, E.H.F. van Asseldonk, and H. van der Kooij. Actively controlled lateral gait assistance in lower limb exoskeleton. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE/RSJ IROS)*, pages 965–970, 2013. 2.6

[167] M. Wehner, B. Quinlivan, P.M. Aubin, E. Martinez-Villalpando, M. Baumann, L. Stirling, K. Holt, R. Wood, and C. Walsh. A lightweight soft exosuit for gait assistance. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 3362–3369, 2013. 2.6

[168] E.C. Whitman. Coordination of multiple dynamic programming policies for control of bipedal walking. *Ph.D. Thesis, Carnegie Mellon University*, 2013. 2.4.1

[169] D.L. Wight, E.G. Kubica, and D.W. Wang. Introduction of the foot placement estimator: A dynamic measure of balance for bipedal robotics. *Journal of Computational and Nonlinear Dynamics*, 3(1), 2008. 2.4.2

[170] C.E. Wilson and J.P. Sadler. *Kinematics and Dynamics of Machinery*. Prentice Hall, 3rd edition, 2003. 6.3.2

[171] D.A. Winter. Biomechanics and motor control of human movement. *Wiley*, 1(4), 2009. (document), 3.3, 3.2.4, 5.1.1, 5.1.1, 5.2

[172] S. Wolf and G. Hirzinger. A new variable stiffness design: Matching requirements of the next robot generation. *Proceedings of the IEEE International Conference on Robotics and Automation (IEEE ICRA)*, pages 1741–1746, 2008. 6

[173] A. Wu and H. Geyer. The 3D spring-mass model reveals a time-based deadbeat control for highly robust running and steering in uncertain environments. *IEEE Transactions on Robotics*, 29(5):1114–1124, 2013. doi: 10.1109/TRO.2013.2263718. 1.1, 2.5.1

[174] G. Wyeth. Control issues for velocity sourced series elastic actuators. *Proceedings of the Australasian Conference on Robotics and Automation*, 2006. 3.2.3

[175] K.K. Yin, K. Loken, and M. van de Panne. SIMBICON: simple biped locomotion control. *ACM Transactions on Graphics*, 26(3):105: 1–10, 2007. doi: 10.1145/1276377.1276509. 1.1, 2.5, 2.5.2, 2.5.4

[176] R. Zaier and F. Nagashima. Motion pattern generator and reflex system for humanoid robots. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 840–845, 2006. 2.5.3

[177] A.B. Zoss, H. Kazerooni, and A. Chu. Biomechanical design of the Berkely lower extremity exoskeleton (BLEEX). *IEEE/ASME Transactions on Mechatronics*, 11(2):128–138, 2006. 2.6