4-8-2012

Development Of A Dynamic Bipedal Climbing And Multi-Modal Robotic Platform

James David Dickson
The Florida State University

THE FLORIDA STATE UNIVERSITY

COLLEGE OF ENGINEERING

DEVELOPMENT OF A DYNAMIC BIPEDAL CLIMBING AND MULTI-MODAL ROBOTIC PLATFORM

By

JAMES DICKSON

A Thesis submitted to the
Department of Mechanical Engineering
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Spring Semester, 2012

James Dickson defended this thesis on March 21, 2012.

The members of the supervisory committee were:

Jonathan Clark
Professor Directing Thesis

Kunihiko Taira
Committee Member

Patrick Hollis
Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the thesis has been approved in accordance with the university requirements.

ACKNOWLEDGMENTS

I would like to thank my research advisor, Dr. Jonathan Clark, for his support and direction during the research for this thesis. I would also like to thank my Committee Members Dr. Kunihiko Taira for his assistance in determining the aerodynamic characteristics of the glider portion of the robotic system and Dr. Patrick Hollis for his assistance in developing 3D dynamic simulations of bipedal climbing. Thank you to members of the Scansorial and Terrestrial Robotics and Integrated Design (STRIDe) Lab, specifically Bruce Miller, Chris Kulinka, and Jason Newton for their ongoing support and assistance in the development of this thesis.

The development of the ICAROS platform presented in this thesis was supported by the Air Force Research Lab (AFRL) under grant FA8651-10-1-0007 as part of the United States Air Force Campus Challenge Problem Solving Competition in miniature self-deploying systems in cluttered/confined environments. The research for this thesis was also supported by the Florida Center for Advanced AeroPropulsion (FCAAP).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Animals often exhibit the ability to operate in and transition between multiple modes of locomotion efficiently and elegantly. On the other hand, robotic platforms have typically focused on a single mode of locomotion. This thesis presents the conceptual development, design, and verification of a robotic platform capable of locomotion in scansorial and aerial regimes based on biological analogs. A review of related work is conducted on animals, previous climbing platforms, and multi-modal robots. A 2D dynamics simulation is developed and the effect of sprawl angle simulated. The development of a miniature bipedal dynamic climbing platform is discussed and an experimental investigation on the effect of sprawl angle on dynamic climbing conducted. The platform design for a multi-modal climbing and gliding robot is presented and a discussion on the trade-offs for multi-modal locomotion presented. The multi-modal platform, ICAROS, is experimentally operated to verify the design specifications. The resulting ICAROS platform demonstrates climbing prepared vertical surfaces and transitioning to a glide path with performance characteristics comparable to its biological counterparts.

CHAPTER 1

INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have transformed modern reconnaissance and search and rescue by enabling prolonged and inexpensive air surveillance. These UAVs are limited in their operational abilities to open, uncluttered environments and have limited operating times due to limited fuel supply. In response to these deficiencies, a new class of robots are beginning to be developed that are small, agile, and capable of multi-modal operation to increase adaptability. This thesis presents preliminary work in the development of Adaptive Robotic Multi-Modal Systems (ARM^2S) capable of rapid adaptation and of operation in urban, or natural cluttered environments. While this thesis presents a multi-modal platform capable of climbing vertical surfaces and transitioning to gliding, future instantiations will provide superior search and monitoring capabilities by exploiting multiple modes of locomotion (such as active flying in confined spaces, running over rough terrain, and climbing vertical surfaces) as shown in Figure 1.1 as well as increased sensing capabilities.

Across the animal kingdom, there are examples of animals capable of impressive feats of multi-modal locomotion including climbing and flying that are fast, agile, and efficient. A theory on the evolution of flight proposes a 'top-down' approach (animals jumping from height) with an intermediary gliding phase for arboreal climbing animals [41]. We present a similar strategy for the development of a multi-modal climbing platform capable of launching from a vertical surface and gliding to the ground (or another perch). Heretofore, a platform capable of fast vertical climbing and gliding has yet to be developed due to the difficulty in achieving high speed vertical climbing, overcoming scaling and mass constraints when integrating the two forms of locomotion, and the difficulties in transitioning between modes of operation. This thesis presents the design, characterization, and operation of a small platform that overcomes these difficulties.

While climbing and gliding is not necessarily more efficient than simply running, it is often much faster and allows for overcoming obstacles, escaping predators, and avoiding rough, complicated terrain. Previous robotic climbing platforms have typically utilized multiple degree of freedom limbs for slow, quasi-static vertical climbing. In order to maintain the rapid mobility conferred by climbing and gliding, a high speed bipedal climbing design was adopted as the basis for the development of the climbing platform. The bipedal climbing design utilized presented the opportunity for significant scale reduction and mechanism simplification to develop the smallest dynamic climbing platform to date. We also show that this platform proves amenable to integration with flight components.

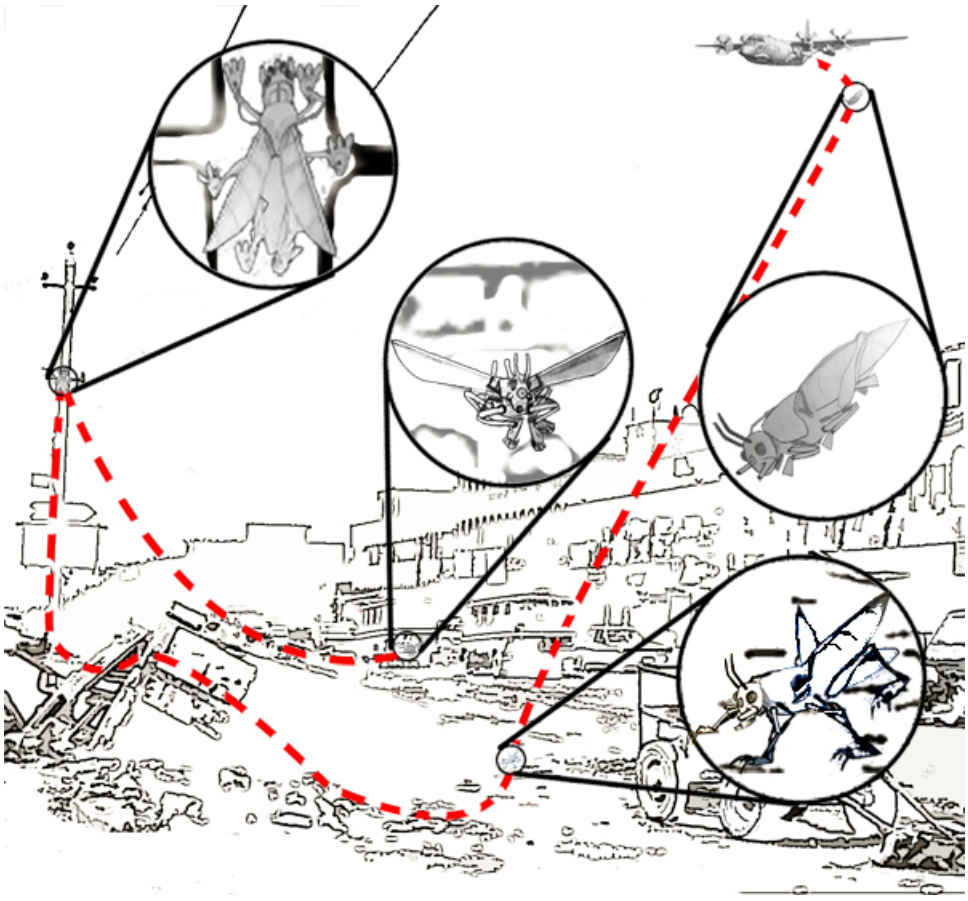


Figure 1.1: Expected operation of a fully developed Adaptive Robotic Multi-Modal System (ARM^2S) in the field. The platform will be air deployable and capable of autonomous flight, running, and climbing in cluttered, unstructured environments such as disaster areas or war zones.

The proposed operation of our platform, ICAROS, is outlined in Figure 1.2. On the vertical climbing substrate the robot will move upward quickly to position itself for launching from the surface(a). Once it has reached a sufficient height for launching from the surface, a detachment mechanism will remove the claws from the surface (b) to allow it to dive vertically (c). Next, when the platform has reached sufficient forward velocity as determined from flight characterization experiments, it will initiate a pitch up maneuver (d) to enter a horizontal glide trajectory (e).

This thesis presents a discussion on the background of multi-modal animals, robots, and climbing platforms in Chapter 2. In Chapter 3 a 2D dynamics simulation is introduced mirroring the expected climber design and a preliminary investigation of the effect of sprawl angle is conducted. The mechanical design of the miniature dynamic climbing platform is presented in Chapter 4 and an experimental study of the effect of sprawl angle is conducted

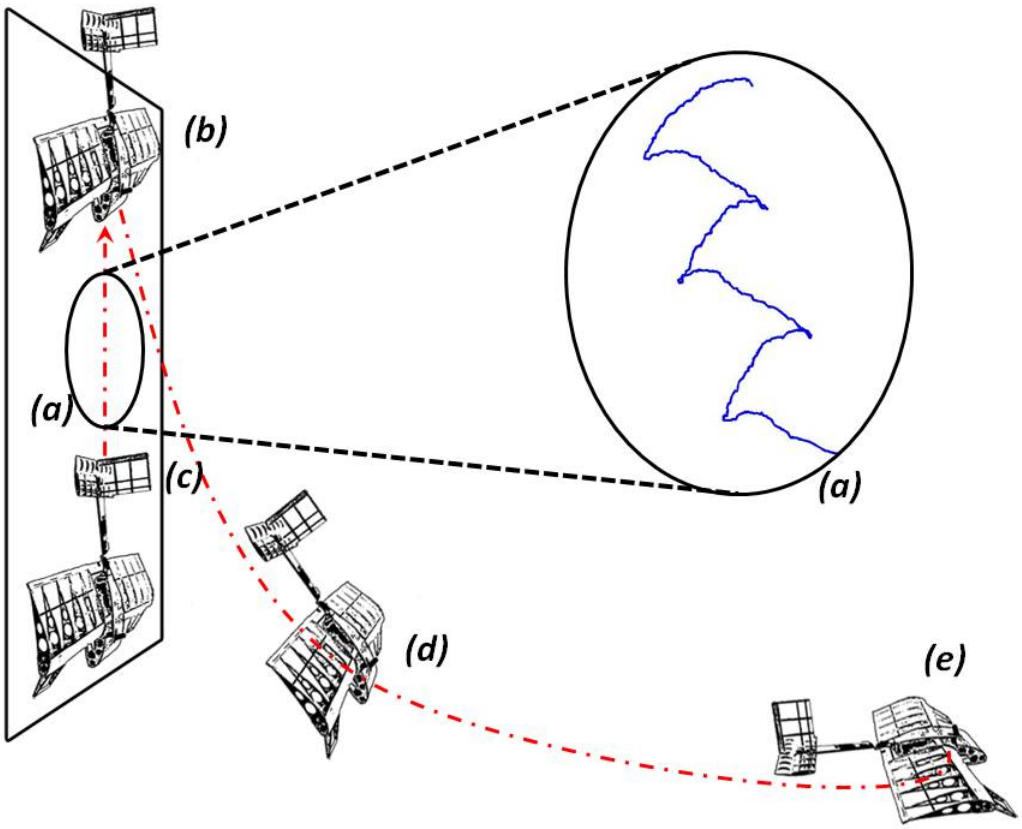


Figure 1.2: Proposed operation of the integrated platform: (a) Dynamic climbing on a vertical surface for launch positioning (b) Detach claws from climbing surface using tail actuation (c) Ballistically dive to gain sufficient forward velocity for transition to gentle glide slope (d) Initiate pitch up maneuver through actuation of the tail elevator (e) Return tail elevator to optimal position for maximum gliding distance.

on the independent climber platform in Chapter 5. The design of the multi-modal platform is presented in Chapter 6 and the experimental performance is discussed in Chapter 7. Finally, Chapter 8 concludes and gives some of the areas for future work that this platform enables.

CHAPTER 2

RELATED WORK

2.1 Biological Precedent

2.1.1 Biological Inspiration for Scansorial Locomotion

Regardless of regime, animals have shown the ability to move fast, agilely, and stably in natural terrain. Robotics has approached the velocities and stability of biological analogs in terrestrial locomotion, but robots capable of scansorial locomotion have lagged behind for some time. In part this is due to a lack of understanding of the dynamics with which animals climb vertically. While locomotion results from complex, high-dimensional, nonlinear, dynamically coupled interactions, to understand the underlying dynamics of a system a system model can be developed to remove redundancy of legs and enforce symmetry. Specifically, a template represents the simplest model possible (least number of variables and parameters) to exhibit a specific behavior [24].

While templates of horizontal ground walking such as the inverted pendulum [13] template and later running such as the Spring-Loaded-Inverted-Pendulum (SLIP) template [24] have been established for some time, a single, general dynamic template for vertical climbers was developed much later [28]. Initially it was thought that animals that utilize a diverse number of attachment strategies and ranging vastly in morphology could not have an overriding dynamic template for scansorial locomotion. Biological studies on two extremely dissimilar species suggested a link between dynamics during vertical running [7] [6].

Biological studies of climbing revealed a similar ground reaction force and velocity profile in species as dissimilar as the gecko *Hemidactylus garnotii* and the cockroach *Blaberus discoidalis*. Initially, data was taken for only the *Hemidactylus garnotii* for initial studies to understand gecko adhesion [8] before focusing on dynamics of vertical running [7]. Patterns in the ground reaction forces were initially noticed during these studies which gave rise to the hypothesis for an overriding dynamic template of scansorial locomotion. The cockroach *Blaberus discoidalis* was chosen as a comparison to the gecko to develop this template for a number of reasons. The *Blaberus discoidalis* is the same body mass as *Hemidactylus garnotti*, both species are nimble, sprawled posture runners capable of climbing vertical substrates quickly, and a large source of previous data is available for *Blaberus discoidalis*. An important aspect of testing the generality of the dynamics between these two species is the dramatic deviation in the number of legs (four versus six) and attachment strategy (dry adhesion via van der Waals versus claws) [28].

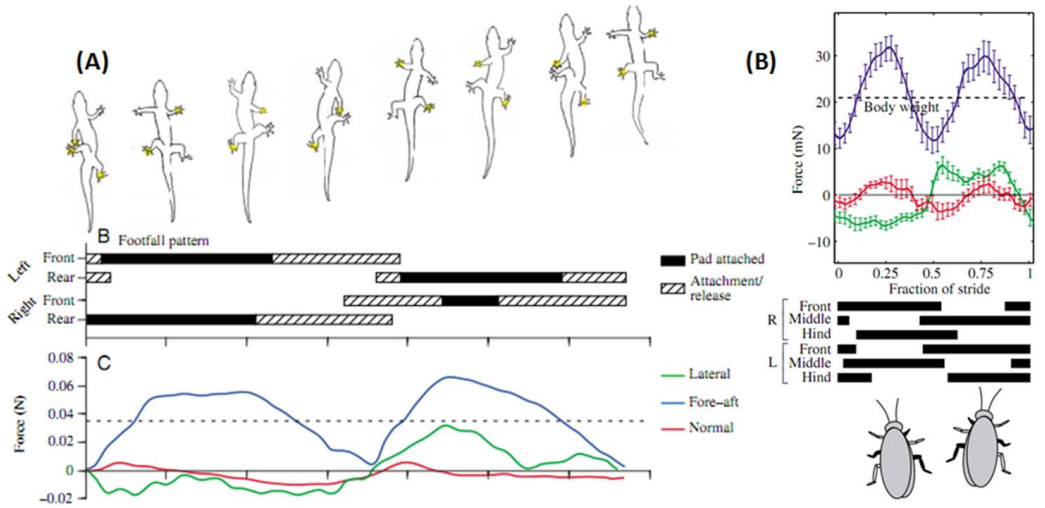


Figure 2.1: **(A)** Generated ground reaction forces integrated across all feet in stance during a single stride for *Hemidactylus garnotii*. The dotted line represents the weight of the animal. Reproduced with permission from [7]. **(B)** Generated ground reaction forces integrated across all feet in stance during a single stride for *Blaberus discoidalis*. The dotted line represents the weight of the animal. Reproduced with permission from [28].

The ground reaction force profile of both animals showed a repeatable, oscillatory pattern for forces fore-aft and lateral to the climbing surface that follow stride frequency. The ground reaction forces generated during a single stride for *Hemidactylus garnotii* and *Blaberus discoidalis* are shown in Figure 2.1 A and B respectively. Both plots show an oscillating fore-aft force that remains, on average, above the body weight of the animal. The lateral forces also show an initial negative force transitioning to a positive lateral force approximately half way through the stride (transitioning between foot stance on the left to the right side). The gecko *Hemidactylus garnotii* was found to attain vertical velocities of $77cms^{-1}$ with a stride frequency of 15 Hz utilizing a trotting gait on a smooth, vertical surface [7]. The cockroach *Blaberus discoidalis* was found to attain vertical velocities of $35cms^{-1}$ with a stride frequency of 13.45 Hz utilizing an alternating tripod gait [28]. Between both species, a oscillatory wave pattern is prominent in both the vertical velocity pattern and lateral velocity pattern.

The similarities between these two dissimilar species enabled researchers to develop a two degrees of freedom template of dynamic, scansorial locomotion. The Full-Goldman (F-G) template developed removes redundant legs, utilizing a bipedal morphology that is actuated through the action of a spring in series with a linear actuator that pulls the body from side to side. The template utilizes a rotationally free pin-joint for attachment that is set to the climbing surface at a fixed phase during oscillation. The linear actuator length is then reduced, pulling the body upwards, and towards the foot attachment point. This cycle repeats for each leg pulling the template up the climbing surface [28]. The comparison of the gecko *Hemidactylus garnotii* dynamics to the cockroach *Blaberus discoidalis* and the

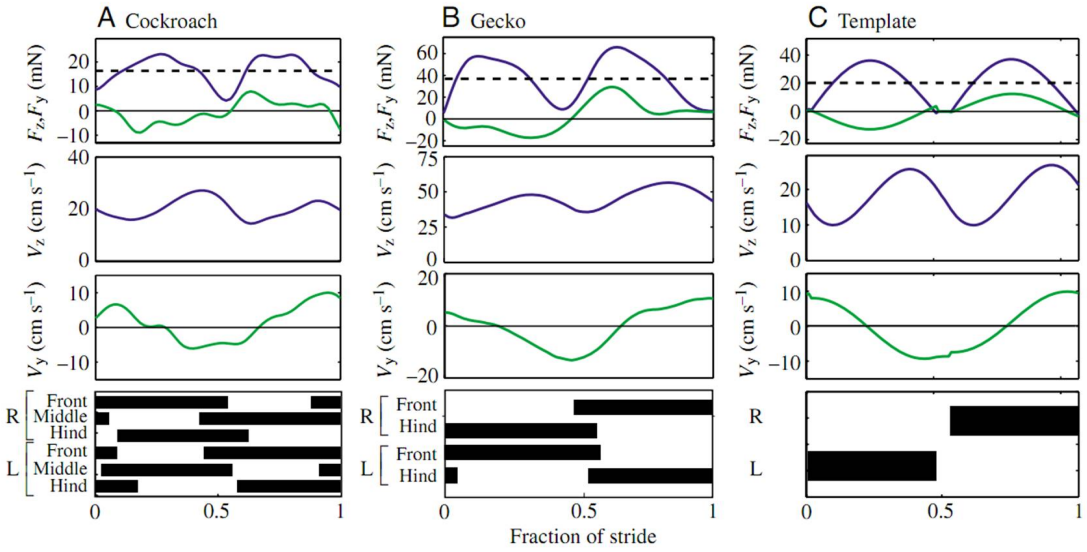


Figure 2.2: Fore-aft (blue) and lateral (green) center of mass (COM) wall reaction forces, COM fore-aft and lateral instantaneous velocity for two steps (one stride) during climbing for **(A)** a cockroach, **(B)** a gecko, and **(C)** the spring-mass model (template). Reproduced with permission from [28].

template is shown in Figure 2.2 A, B, and C respectively. During the development of the template, it was also hypothesized that animals generate specific dynamics to passively control body oscillations through a fixed leg angle referred to as sprawl angle, or β.

2.1.2 Multi-modal Animals

The *Draco* genus of lizards has developed the ability to glide to such an extent that they have been referred to as 'flying dragons'. There are more than 40 species in the *Draco* genus, with body sizes ranging in mass between 3 and 20g [38] [39]. Draco lizards use a patagium, or gliding membrane, when flying, which is unique in the animal kingdom due to their ability to actively control the membrane's aerodynamic properties using elongated thoracic ribs and specialized musculature [38] [45]. Draco lizards have also shown other physical adaptations along their necks, legs, and edges of their hind quarters that are used for maximizing the aerodynamic surface [38].

Reptiles with radically different morphologies are also capable of gliding. Snakes in the genus *Chrysopelea* have shown the ability to glide using behavioral and physical adaptations to create an airfoil shaped body to substantially decrease their descent accelerations. There are five species of 'flying snakes' that flatten their bodies, increasing the surface area and forming a concave bottom surface creating an effective whole-body wing [48]. The *Chrysopelea paradisi*, or the paradise tree snake, has even been shown to temporarily achieve a positive (upward) acceleration during flight [48].

Mammals have also shown similar adaptations for gliding flight as reptiles. Flying

Table 2.1: The glide performance of a number of gliding animals based on biological studies. Glide velocity represents the mean magnitude air velocity over the 'wing'.

Common Name	Biological Name	Mass (g)	Wing Load (Nm^{-2})	Glide Ratio	Glide Velocity (ms^{-1})
Thai Flying Dragon	Draco taeniopterus [39]	3.4 ± 1.2	10 ± 1.5	2.14 ± 0.4	6.1 ± 1.1
Fringed Flying Dragon	Draco fimbriatus [39]	18.7 ± 2.9	23.5 ± 0.75	1.82 ± 0.01	6.4 ± 0.7
Paradise Tree Snake	Chrysopelea paradisi [49]	40.5 ± 23.1	29 ± 9	2.2 ± 1.1	8.9 ± 1.4
Northern Flying Squirrel	Glaucomys sabrinus [47]	125.9 ± 12.63	50	1.5 ± 1.20	7.2 ± 0.9
Indian Giant Flying Squirrel	Petaurista philippensis [32]	1539 ± 250	-	2.32 ± 0.96	6.96 ± 1.94

squirrels have shown physical adaptations evidenced by the stretched patagial skin between their fore and aft limbs used for controlled gliding [25]. They have been observed to have glide ratios of up to approximately 3; e.g. a 30m glide over a 10m drop. Some species of flying squirrel have shown the ability to modify the shape of their gliding membranes in flight using the abductor of the thumb. Due to this adaptation, small flying squirrels, in particular *Glaucomys volan*, are extremely agile and maneuverable [52]. These animals have been reported to have glide durations of up to 15s have been reported [4] [12].

These diverse animals have all developed the ability to climb and glide in varying ways, vastly increasing their locomotive capabilities. While the animals' bodies are not necessarily optimal for either the scansorial or aerial domain, an examination of their relative performance gives an estimate of expected performance trade-offs inherent in combining these. The glide characteristics (i.e. wing loading, glide ratios, and flight velocities) of these animals, summarized in Table 2.1, provides a baseline for which to compare the empirical results of the ICAROS platform. Based on biological precedent, we can expect glide ratios of 1-3, a wing loading that increases as a function of mass, and a glide velocity between 6 and $8ms^{-1}$ regardless of scale. A comparative study of Draco lizards showed similar patagial dimensions and glide velocities regardless of scale between the species. This study suggests that while the glide velocity is independent of scale, the glide ratio decreases with size [37].

Based on these examples, the platform was designed to be capable of a glide ratio of about 2 and a glide velocity of about $7ms^{-1}$. The wing loading of the animals discussed varies widely, but averaged to a target value of $28Nm^{-2}$. Although we refer to biology to estimate realistic performance objectives, we, at least initially, restrict ourselves to a more traditional aviation morphology. In future iterations, we are interested in utilizing more biologically similar wings, such as the patagium utilized by Draco lizards and flying squirrels.

2.2 Climbing Robots

Scansorial robots have utilized a number of locomotive strategies including wheels, treads, and legs. Of these locomotive strategies, legged climbing platforms hold the greatest potential for the development of agile, multi-modal robots. Legs are capable of both

terrestrial and scansorial locomotion with minimal physical adjustment and can easily be retracted for flight. Regardless of attachment strategy, a large portion of previous legged climbing platforms have focused primarily on quasi-static climbing, ignoring the momentum of the system leading to slow vertical climbing. Platforms such as the RiSE robots have shown the ability to climb on a diverse number of surfaces [51]. While these robots can successfully climb surfaces such as brick walls and trees, their vertical speeds pale in comparison to their biological counterparts. For example, RiSE version 2 (3.8 kg), climbs at $0.89 cms^{-1}$ [51] while animals such as the gecko Hemidactylus garnoti (1.9 ± 0.7 g) have shown vertical running at speeds up to $77 cms^{-1}$ [7].

Table 2.2: Climbing velocities of robotic and biological systems when scaled to mass of 200 g based on dynamic similarity scaling discussed in Section 3.1.

Platform	Mass(kg)	Velocity (cms^{-1})	Scaled Velocity(cms^{-1})
SpinyBot [5]	0.4	2.3	2.05
StickyBot [46]	0.37	6.0	3.61
CMWhegs [17]	0.087	5.8	6.63
RiSE V2 [51]	3.8	0.895	0.55
RiSE V3 [29]	5.4	21	12.12
ROCR [44]	0.55	15.7	13.26
Dynoclimber [35]	2.6	66	43.04
B. discoidalis [28]	0.003	20	40.27
H. garnoti [7]	0.002	53	115.16

Though quasi-static climbers are reliable, their vertical speeds pale in comparison to biological climbers such as squirrels, geckos, and cockroaches. Dynamic climbing robots, on the other hand, actively manage their kinetic energy while running on vertical substrates and have the potential to approach the climbing speeds exhibited in nature. Using dynamic scaling relationships, discussed in Section 3.1, the climbing velocities of a number of scansorial robotic platforms and biological analogs are compared in Table 2.2. As this table shows, most scansorial platforms are significantly slower than animals when scaled to the same mass. Bio-inspired dynamic climbing presents an opportunity to significantly reduce this disparity.

While there are a number of dynamic climbing robots, such as ROCR [44] and DSAC [18], the application of the biological template, discussed in Section 2.1.1, has led to the development of the fastest climbing robot, Dynoclimber. This platform, shown in Figure 2.3, exploits the momentum of the system and actively controls the kinetic energy to rapidly scale vertical surfaces at speeds up to 1.5BL/s [35].

Dynoclimber is a 2.6kg, bipedal dynamic climbing robot that utilizes two linearly actuated legs to climb vertical surfaces rapidly. Using tuned control parameters, Dynoclimber has reached speeds of up to 66cm/s [35]. The simplicity of the template and the Dynoclimber platform has raised the potential for miniaturization and integration into platforms capable of other forms of locomotion.

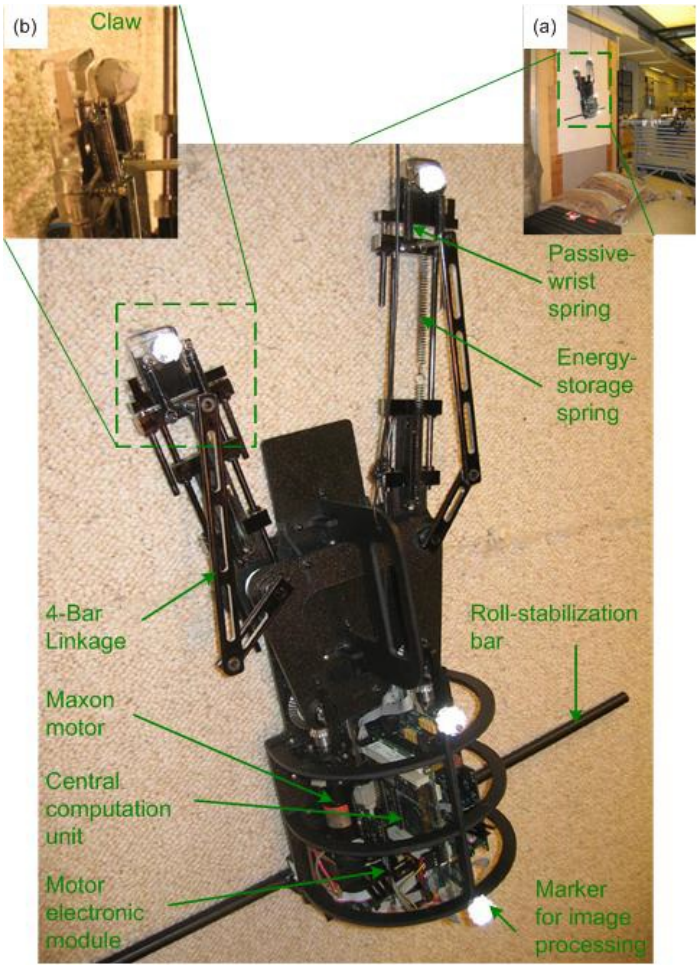
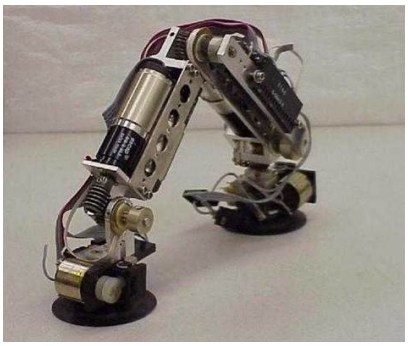


Figure 2.3: The dynamic climbing platform developed by [15] based on the template developed by [28] and discussed in Section 2.1.1.

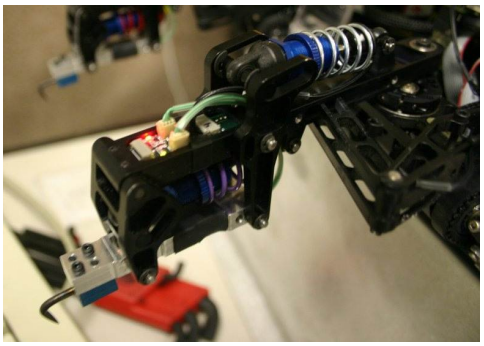
Attachment Mechanisms. Attachment is an important consideration for any climber or scansorial platform. Attachment strategies, until recently, have been focused on quasi-static climbing robots since previous climbing robots have moved slowly, requiring the preservation of static equilibrium throughout their gait [6, 50].

Quasi-static climbers include the RiSE family, a series of robots varying in mass from 2kg to 3kg [29, 51, 6], SpinyBot, a 0.4kg robot [31], as well as LemurBot, a 7kg free climbing robot capable of using footholds on a prepared surface [11]. These robotic platforms use integrated dactyl claws or prepared footholds while other platforms have utilized electromagnets [43], suction [40] or hook and loop attachment [15].

Many insects, arthropods, and reptiles capable of climbing vertical surfaces have small spines integrated into their legs [16, 31]. These spines can engage asperities, or irregularities on the surface to support the animal's weight. Application of these principles led to the development of compliant micro-spine arrays capable of attaching surfaces including drywall,



(a)



(b)

(c)

Figure 2.4: Attachment mechanisms for scansorial locomotion. (a) Biped climbing robot that utilizes suction cup attachment. Reproduced with permission from [40] ©2000 IEEE. (b) Dactyl claw of the RiSE platform. Reproduced with permission from [30] Robotics: Science and Systems II edited by Gaurav S. Sukhatme, Stefan Schaal, Wolfram Burgard, and Dieter Fox published by The MIT Press. (c) Micro spine arrays developed originally for Spinybot platform. Reproduced with permission from [30] Robotics: Science and Systems II edited by Gaurav S. Sukhatme, Stefan Schaal, Wolfram Burgard, and Dieter Fox published by The MIT Press.

stucco, and brick [5, 31]. The Spinybot platform was the first robot to implement the specialized feet on a 0.4kg platform [31]. These feet were later adapted to the 2kg platform RiSE and the UAV platform of the Perching Project [51, 21].

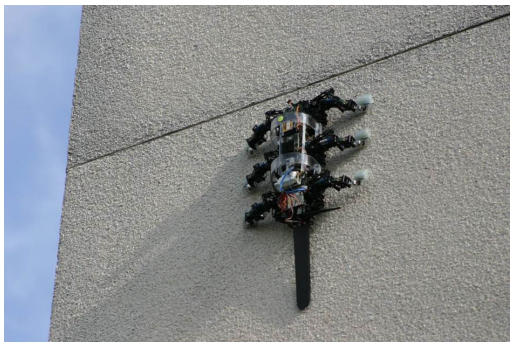
2.3 Multi-Modal Robots

There are a number of platforms capable of locomotion over differing terrain but require outside modifications for transitioning. These include AQUA, RiSE v2.0, WaterRunner, and the Self Deploying Microglider. Some of these platforms have shown the ability to transition modes of locomotion independently, but many require outside modifications for multi-modal operation.

The Rotary Hexapod platform, RHex, has over the last decade proven to be the best legged platform for locomotion over rough, unstructured terrain. Another version of the RHex platform, termed Aquabot, replaces the typical C-legs with paddles for locomotion underwater. The platform has also shown limited capabilities for transitioning to land through walking on the paddle legs, though not as robust as the RHex platform [26].

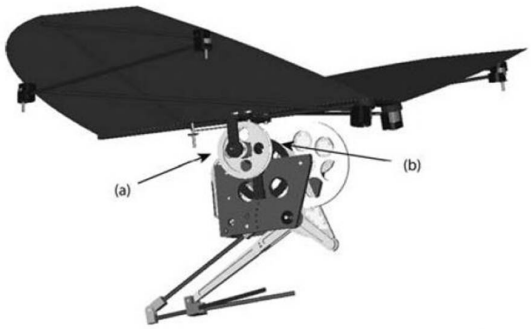
The RiSE family of robots has shown capabilities of climbing vertical surfaces of diverse compositions, transitioning to horizontal walking [6], and pole climbing at high speed [29]. RiSE v2.0 with integrated spiny claws [51] has shown the ability utilizing its stabilizing tail to transition from vertical scansorial locomotion to terrestrial walking at the top of a climb.

(a) AQUA robot



(b) RiSE v2.0

(c) WaterRunner platform



(d) EPFL Jump Glider

Figure 2.5: Multi-modal platforms. (a) Reproduced with permission from [27] ©2009 IEEE. (b) Reproduced with permission from [30] Robotics: Science and Systems II edited by Gaurav S. Sukhatme, Stefan Schaal, Wolfram Burgard, and Dieter Fox published by The MIT Press. (c) Reproduced with permission from [23] ©2008 IEEE. (d) The EPFL Jump Glider [3] reproduced with kind permission from Spring Science and Business Media in Flying Insects and Robots, 2009, pg. 282, Chapter 19, Towards a Self-Deploying and Gliding Robot by Mirko Kovac, Figure 19.15, ©Springer-Verlag Berlin Heidelberg 2009.

The WaterRunner platform draws inspiration from the Basilisk lizard, *Basiliscus basiliscus*, that utilizes drag forces exerted by fast motion of its feet to run over the surface of water. The robotic platform is a quadrupedal design that uses a compliant foot pad design

to create similar drag forces for running over the surface of the water. The platform has also been shown to be capable of running over terrestrial environments, though it requires a footpad of significantly different stiffness. In order to transition between these two regimes, the foot pads must be replaced for the specific terrain [42].

The Self Deploy Microglider, developed at École Polytechnique Fédérale de Lausanne (EPFL) in the Laboratory of Intelligent Systems (LIS), has drawn biological inspiration from grasshoppers and cicadas for the development of a platform capable of jumping and gliding for short distances. Utilizing a micro jumper platform capable of reaching heights of 76 cm on its own, the jump glider can jump into the air and glide for a limited distance. An early CAD model of the Jump Glider is shown in Figure **??**. Currently iterations of the platform utilize fixed glider wings which cause significant drag on the platform during the initial jump phase. Researchers are currently working on wings capable of folding and deploying quickly to reduce the drag and increase initial jump height [33].

While there are few examples of robotic platforms capable of multi-modal locomotion, there are fewer examples of platforms that integrate flying onto a platform capable of legged locomotion in a terrestrial or scansorial environment. The Morphing Micro Air-Land Vehicle (MMALV) integrates Mini-Whegs^{TM} and a MAV into a platform capable of moving on flat terrain and flying [10]. The Perching Project integrates the ability of a plane to perch on vertical surfaces, but cannot climb [19]. There have been no examples of robotic platforms that integrate climbing and flight to date.

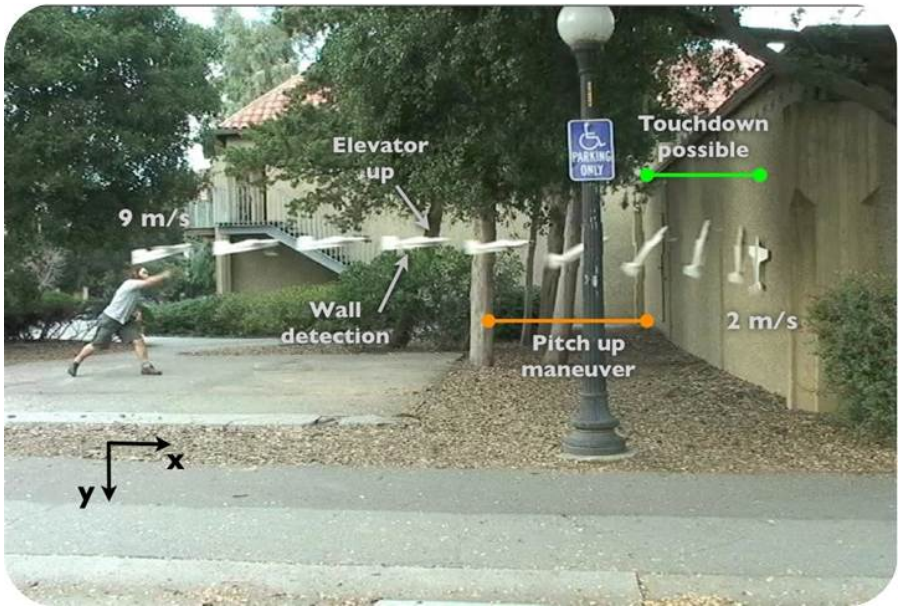
The MMALV, shown in Figure 2.6, integrates Mini-Whegs^{TM} and a flexible wing Micro Air Vehicle (MAV) into a platform capable of moving on flat terrain and flying [10]. The Morphing Micro Air-Land Vehicle transitioned from walking to flying by diving from a sufficient height to generate enough lift to fly. To transition to walking, the platform performed a crash landing maneuver on flat ground. During terrestrial locomotion, the platform was capable of folding its wings into a stowed configuration [10], but could not fly when outfitted with folding wings, climb vertical surfaces, or take off from the ground.

Specialized, compliant feet originally developed on the Spinybot platform [5] and later integrated into the RiSE project [51] were adapted for use on an Unmanned Air Vehicle (UAV) as part of the Stanford Perching Project [21, 19]. Implemented on the Flatana aerobatic plane, shown operating in Figure 2.7a, the microspine arrays allow the UAV to land on vertical surfaces of stucco or brick [21]. An ultrasonic sensor measures the platform's distance from the wall and engages a stall maneuver to pitch the plane upwards and into the wall. To absorb the shock of impact during landing, the microspines were combined with a purpose built suspension to allow the platform to a land on the vertical surface at a given velocity and engage the feet across as many asperities as possible. To transition from perching to flight, the suspension had to be actively disengaged from the vertical surface. A wire made from Nitinol, a shape memory alloy (SMP), was attached to a lever that actively pulls the spines away from the wall in such a way as to reduce the probability of jamming on cracks or pits during take off [21].

(a) Folding wings for terrestrial locomotion

(b) MMALV take-off from roof

Figure 2.6: MMALV platform developed at University of Florida and Case Western Reserve University that incorporates Mini-Whegs into a MAV platform. Figures reproduced from [10] ©2005 IEEE.



(a) Landing procedure for perching project

(b) Take off procedure for perching project

Figure 2.7: Components of the Perching Platform project. (a) The Stanford Perching Project reproduced with kind permission from Spring Science and Business Media in Journal of Intelligent and Robotics Systems, 2009, pg. 322, Section 5, Landing and Perching on Vertical Surfaces with Microspines for Small Unmanned Air Vehicles by Alexis Lussier Desbiens and Mark R Cutkosky, Figure 8, ©Spring Science + Business Media B.V. 2009. (b) Multiple exposure photograph of the takeoff sequence. Reproduced with permission from [20] ©2011 Sage

CHAPTER 3

SIMULATION

3.1 Dynamic Scaling and Design Considerations

The desire for operability in cluttered environments motivated making the platform as small as possible. For climbing, a smaller scale is advantageous as it reduces power density concerns [14] and aids in attachment to the substrate [5]. The lower bound for scale was constrained by the manufacturing methods chosen and the availability of off-the-shelf electronic and transmission components. Consideration of these constraints led to a target mass of 200g. Dynamic scaling laws were utilized to ensure preservation of dynamic stability properties while scaling the platform from the original 2g Full-Goldman template originally presented in [28].

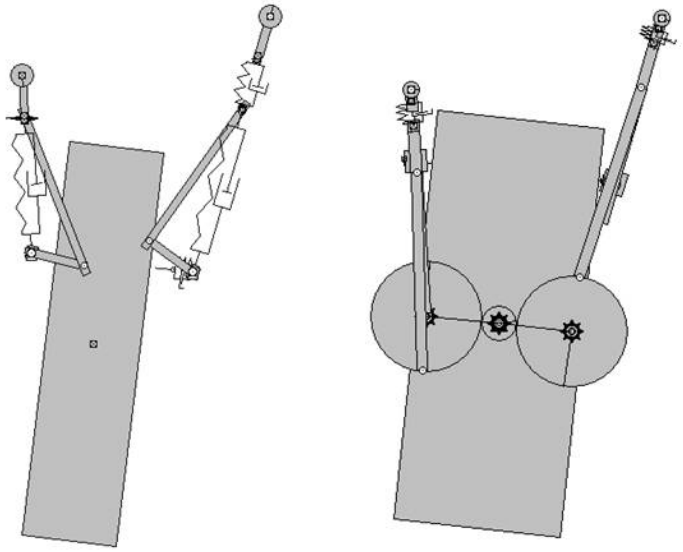
In dynamic scaling, all geometric properties are changed by the same scaling factor, α_L; however, the stiffness, mass, damping, and frequencies must be scaled by alternate ratios. A more detailed discussion of dynamic similarity and a derivation of the associated scaling laws adapted here is given by Clark, et al [14] and Alexander [2]. The resulting scaling relations and results from scaling the FG template are summarized in Table 3.1.

Table 3.1: Dynamic Similarity Scaling Factors and Template and Scaled Quantities based on Template Presented in [28]

Property	$f(\alpha_L)$	$\alpha_L = 4.6$	Template	Scale
Length (m)	α_L	4.642	0.04	0.19
Mass (kg)	α_L^3	100	0.002	0.20
Frequency (Hz)	$\alpha_L^{-1/2}$	0.464	9	4.18
Stiffness (Nm^{-1})	α_L^2	21.548	6	129.29
Velocity (cms^{-1})	$\alpha_L^{1/2}$	2.154	17	36.62
Damping ($Ns * m^{-1}$)	$\alpha_L^{5/2}$	46.426	0.09	4.18
Power Density (Wkg^{-1})	$\alpha_L^{1/2}$	2.154	6.33	13.64

3.2 Simulation

The under actuated bipedal climber was initially simulated and designed based on the Working Model 2D simulation of Dynoclimber, shown in Figure 3.1b. This simulation incorporated either a frequency or torque controller for two motors, each of which drove an arm. While this simulation gave acceptable results after applying dynamic scaling and utilizing a frequency controller, the use of a torque controller was not applicable due to the simplified transmission system proposed for the miniature bipedal climber.



(a) Dynoclimber 2D Simulation in Working Model 2D

(b) Miniature bipedal climber 2D Simulation in Working Model 2D

Figure 3.1: 2D simulation developed for Dynoclimber (Left) and the 2D simulation developed for miniature bipedal climber (Right), both in Working Model 2D

In order to capture the dynamics of the simplified, under actuated system a new simulation was developed in Working Model 2D, shown in Figure 3.1a. The simulation utilizes a geared transmission system in which a single motor supplies torques to both gears. Each gear serves as a crank for the four bar mechanism that drives the linear guide rail that are angled at a prescribed sprawl angle. A wrist spring of 131 N/m is incorporated into the system connecting the foot to the end of the guide rail. For simulations where a wrist spring is not incorporated the wrist spring stiffness is increased an order of magnitude. The foothold is approximated as a pin joint during stance and turned off during flight. The stance conditions are calculated based on the angular position of the gears driving the four bar mechanism.

The developed simulation was compared to the dynamically scaled Dynoclimber simulation for validation. The two simulations utilized a frequency controller of 4.2 Hz which led to vertical velocities of 0.328 ms-1 for Dynoclimber and 0.368 ms-1 for the miniature

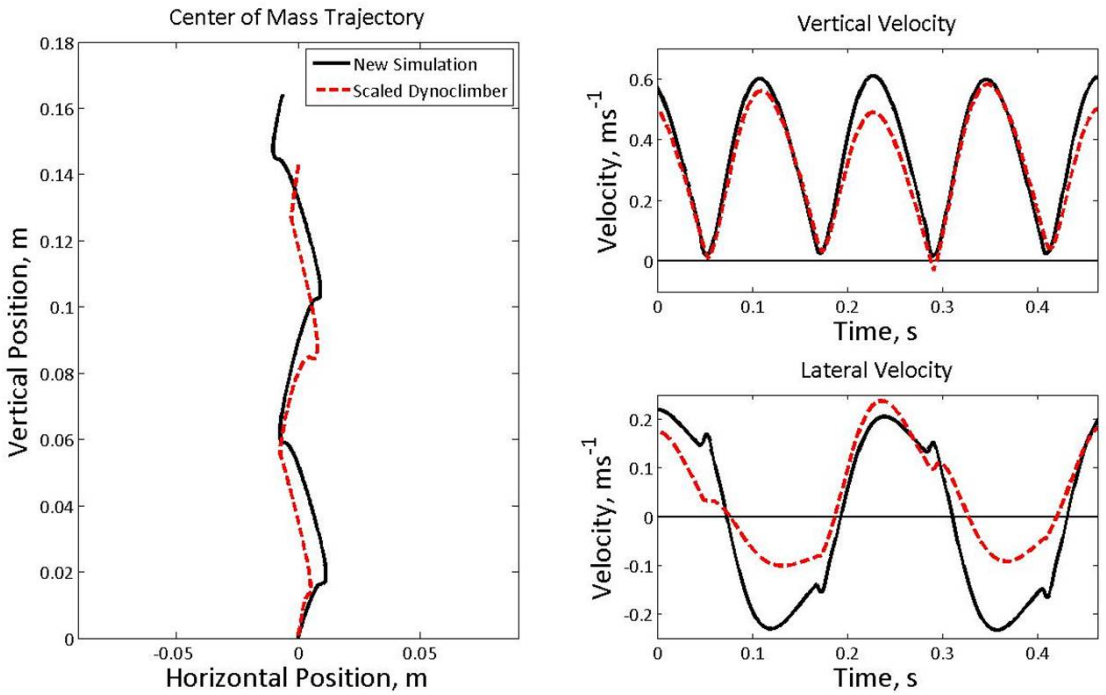


Figure 3.2: Comparison of original Dynoclimber 2D simulation (dotted red line) and the developed miniature biped 2D simulation (solid black line) utilizing a fixed frequency controller of 4.2 Hz and a wrist spring of 129 N/m

biped. The max lateral velocities predicted are 0.240 ms-1 for Dynoclimber and 0.214 ms-1 for the miniature biped. As Figure 2 shows, the dynamics of the system are captured more readily by the developed miniature biped simulation. The new simulation approximates the vertical velocity at the predicted dynamically scaled value as well as the less variability in velocities at steady state.

Once the validity of the dynamics was verified between the two simulations utilizing a frequency controller, a simple motor model was developed for the under actuated system. The torque requirements of the platform were developed utilizing a crank-slider mechanism numerical simulation based on derivations from [9] and a motor selected based on these requirements. The specifications of the selected motor were included in the motor model developed and operated along the motor torque curve shown in Figure 3. A torque limited speed controller was implemented to spin the motor at the desired speed if required torque was less than stall torque or spin at the maximum speed possible if required torque was greater. Utilizing the torque limited speed controller, the simulation reached 0.375ms-1 vertical velocity and 0.216ms-1 lateral velocity agreeing with previous simulations utilizing the frequency controller.

The primary motivation for the addition of the motor model to the 2D simulation is the analysis of the effects of friction drag on the climbing surface when the wall inclination is reduced from vertical. The simulation was run with sprawl angles between -30 and 30 on wall inclinations varying between 90 (vertical) and 50 from horizontal incrementing by 10

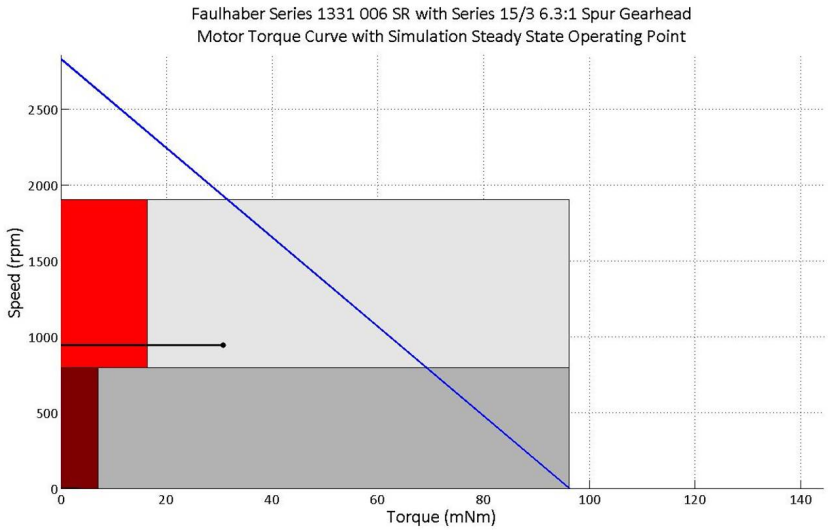


Figure 3.3: Motor torque curve for the Faulhaber Series 1331 006 SR with Series 15/3 6.3:1 Spur gearhead with design point generated from steady state operation of the 2D miniature bipedal dynamic climbing platform. Initial torque requirements for the motor were generated using a numerical four-bar mechanism simulation developed based on [9]

each time. The simulation utilized the torque limited speed controller previously discussed for each data set.

$$F_f = \mu(9.81 * mass) * cos(\psi) \tag{3.1}$$

The frictional force due to drag on the carpet was implemented as a force opposing vertical motion equal to the normal force of the robot, expressed as gravitational constant (9.81) multiplied by the mass of the robot and the cos of the angle of the wall (ψ), on the wall multiplied by the coefficient of friction, as shown in (1). The coefficient of friction was empirically calculated for the carpeted climbing wall as 0.2650.

3.3 Effect of Sprawl Angle on Dynamic Climbing

3.3.1 Definition of Sprawl Angle

The development of the Full-Goldman template incorporated an important characteristic of biological scansorial locomotion to describe fast vertical climbing: lateral forces and movement. This characteristic of vertical climbing seen in animals is due to sprawled posture during climbing that introduces lateral forces for passive stabilization. The development of the Full-Goldman template, discussed in Chapter 2, initially utilized a sprawl angle of 10° to match the observed dynamics of the gecko and cockroach.

The sprawl angle, commonly delineated as β, corresponds to the angle from vertical of the characteristic ground reaction force vectors [36]. The simplicity of the bipedal design of the Full-Goldman template simplifies the specific applied definition of a sprawl angle,

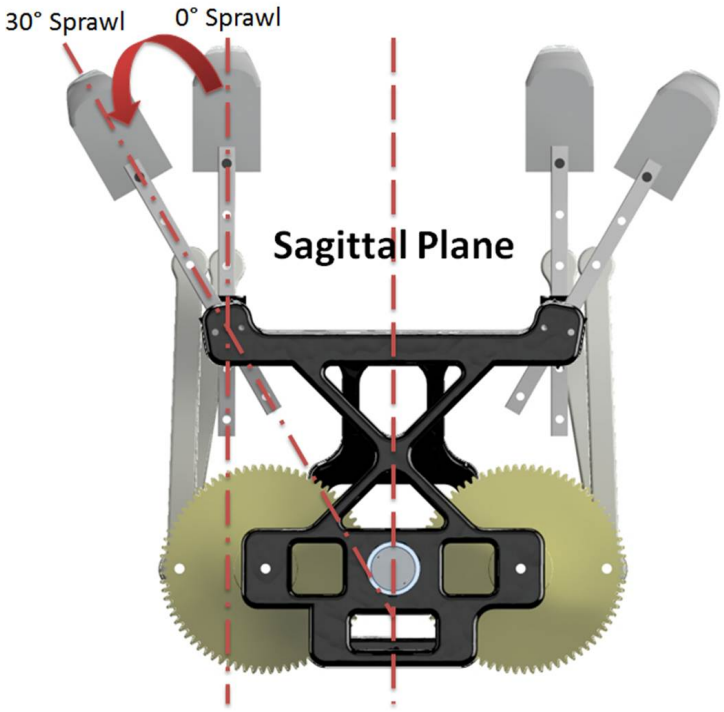


Figure 3.4: Bipedal dynamic climber definition of sprawl based on the simple morphology of the Full-Goldman template.

due to the linear motion of the legs, to the angle between the sagittal plane and the leg, as shown in Figure 3.4.

3.3.2 Previous Studies of the Effect of Sprawl Angle on Dynamic Climbing

Studies on the effect of sprawl angle have been conducted utilizing simulations of bipedal dynamic climbing platforms using the Full-Goldman template. Initial studies focused on the selection of sprawl angle suitable for robotic platforms, while more recent studies have looked at the effect on velocity and power requirements.

Clark et al performed a preliminary investigation of the effect of sprawl angle utilizing a 2D simulation of the Full-Goldman template with fixed stiffness, damping, and touchdown timing parameters while varying sprawl angle, β. The results, shown in Figure 3.5, showed the platform reached a maximum velocity at a sprawl of 30° as well as the angular velocity and peak lateral forces do not increase monotonically, but display minimums at approximately 10°. It was also found that the corresponding minimum approximated the effective angle or force ratio that the cockroach and gecko utilize during climbing, 14° and 11° respectively. The same study also found benefits to perturbation rejection utilizing sprawl angles greater than 10° [15].

Lynch et al investigated the effect of sprawl angle on a world-switched motor-powered

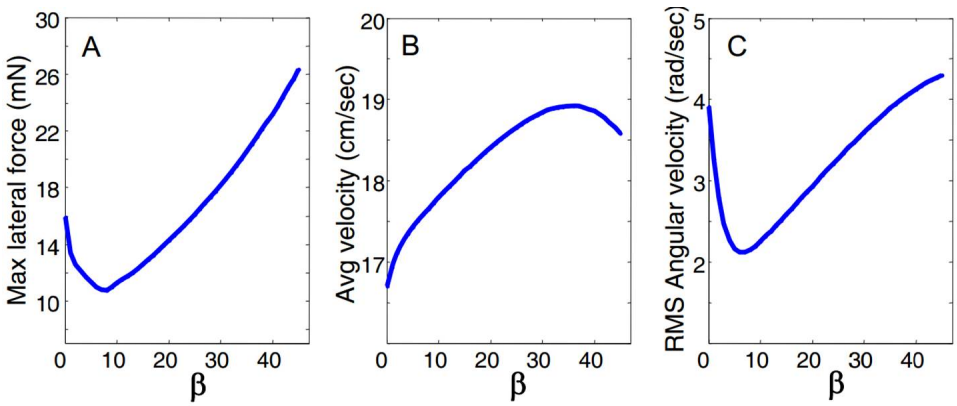


Figure 3.5: Variation of template dynamics as β, the angle of the leg, increases from 0 to 45 degrees. (a)Lateral force (b) Average climbing velocity (c) Angular velocity. Figure reproduced with permission from [15].

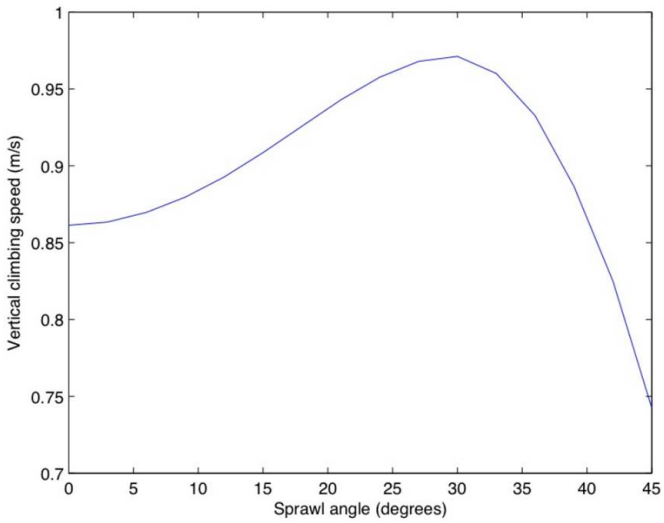


Figure 3.6: Steady-state vertical climbing velocity plotted versus sprawl angle for the world-switched motor-powered climber. Note that a sprawl angle of roughly 30° maximizes vertical climbing speed. Figure reproduced with permission from [34].

climber and a world-switched muscle-powered climber simulation. The results showed for the motor-powered simulation a maximum vertical velocity, shown in Figure 3.6, attained utilizing a sprawl angle of approximately 30°, agreeing with the previous study by Clark et al [15]. Utilizing the muscle-powered climber simulation, the vertical velocity was shown to decrease with increasing sprawl angle [36].

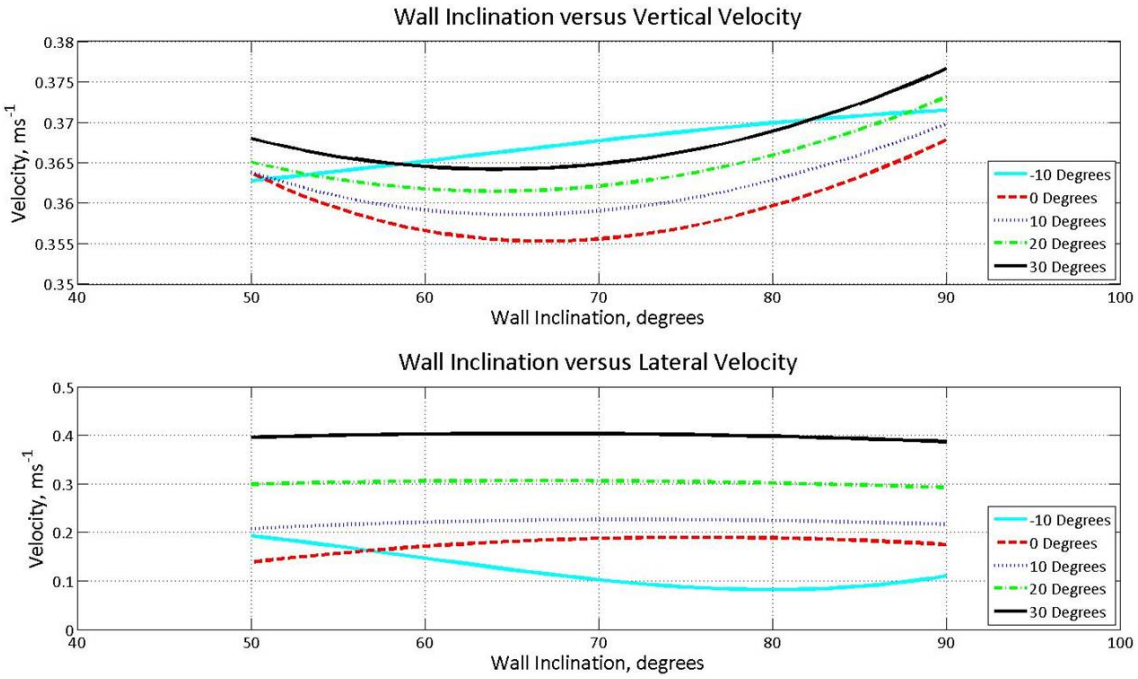


Figure 3.7: Vertical and lateral velocities of the bipedal dynamic simulation utilizing a mass of 200 g, a torque limited speed controller operating at 4.2 Hz, and wrist spring stiffness of $129Nm^{-1}$ run on varying wall inclinations between 90° (vertical) and 50° incrementing by 10° with sprawl angle between $-10°$ and 30° incrementing by the same amount.

3.3.3 Simulation Results for the Effect of Sprawl Angle on Dynamic Climbing

Once the validity of the 2D Working Model simulation was verified through comparison with simulations of the F-G template and Dynoclimber, a number of simulations were run varying the variables of interest: sprawl angle and wall inclination. The simulation was run utilizing sprawl angles varying between $-10°$ and 30° incrementing by 10° each time for each wall inclination which was varied between 90° (vertical) and 50° incrementing by 10°. The simulation utilized a 200 g platform mass with a $129Nm^{-1}$ wrist spring and a torque-limited speed controller set to a frequency of 4.2 Hz.

The velocities of the simulation, shown in Figure 3.7, show perceptible trends with the alteration of sprawl angle. The simulation shows a maximum vertical velocity utilizing a sprawl angle of 30° for most wall inclinations. A sprawl angle of $-10°$ does not follow the same patterns shown by the other sprawl angles and attains an overall maximum velocity compared to sprawl angles between 0° and 30° at wall inclinations between 60° and 80°. For sprawl angles between 0° and 30° a minimum vertical velocity is shown at a wall inclination of approximately 70°. This wall inclination represents a maximum resistive force on the platform through the combination of the gravitational force and drag force of the wall. Overall, the variation in vertical velocities is minute when compared to their

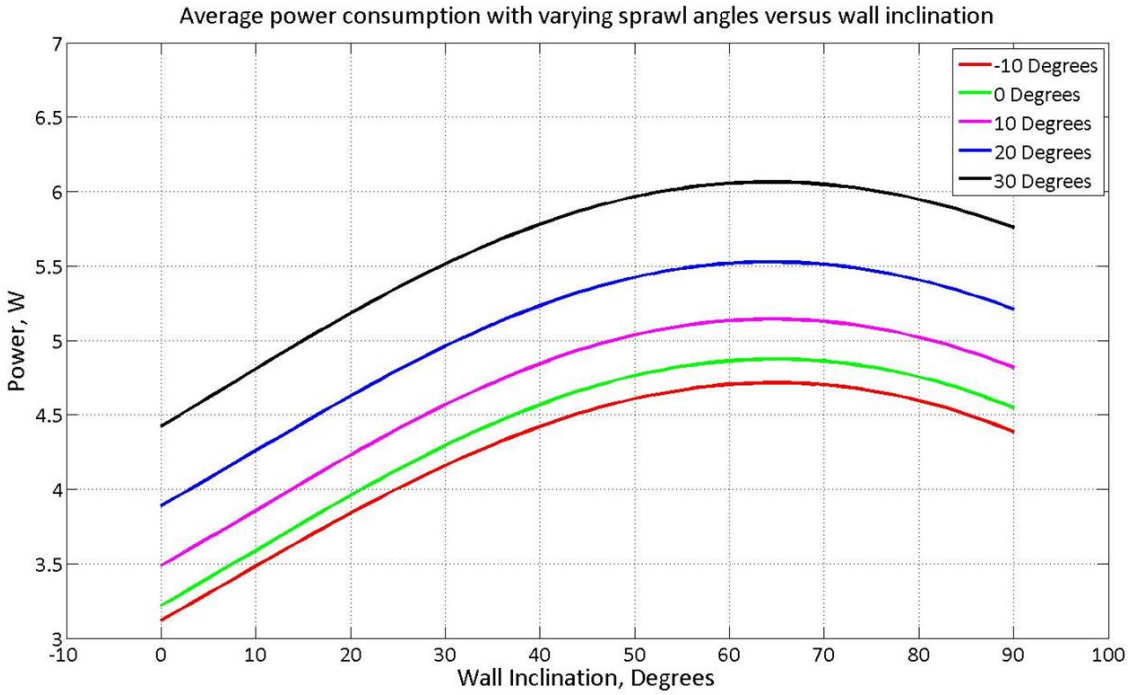


Figure 3.8: Comparison of the power consumption of the bipedal dynamic simulation for each sprawl angle over wall inclinations varying between 90° (vertical) and 0°.

overall magnitude. The lateral velocity shows an expected upward trend in maximum lateral velocity with increasing sprawl angle. Again, a sprawl angle of $-10°$ defies the perceived trend and shows a larger max lateral velocity at low wall inclinations but switches to minimum max lateral velocity at wall inclinations greater than approximately 58°.

Analysis of the average power consumption of the bipedal simulation, shown in Figure 3.8, shows a strong trend in increasing sprawl angle. As the sprawl angle increases from $-10°$ to 30° the power consumption increases over all wall inclinations. The difference in power consumption between sprawl angles also shows an increase at larger sprawl angles, while sprawl angles of $-10°$ and 0° are fairly close in power consumption. All sprawl angles also show a maximum power consumption at a wall inclination of approximately 65°. This wall inclination, similar to the minimum observed in vertical velocity in Figure 3.7 is most likely due to the maximum resistive force through the combination of gravitational force and drag force of the wall.

The trends shown in the simulation of sprawl angle show similarities to previous studies on the effect of sprawl angle on dynamic climbing. It is expected that a sprawl angle of 30° will maximize vertical velocity and max lateral velocity on the experimental platform. It is also expected that a sprawl angle of 0° will minimize the lateral velocities. The power consumption of the platform is expected to increase with increasing sprawl angle.

CHAPTER 4

DEVELOPMENT OF MINIATURE DYNAMIC CLIMBING PLATFORM

The design of the dynamic climbing platform was directly derived from the Dynoclimber robot. The previous robotic platform developed based on the F-G template utilized two DC motors and dual four bar mechanisms for actuation. A complex control system developed for the climber offset the position of each crank, to ensure out of phase synchronization when climbing at different speeds [35].

To reduce the mass of the ICAROS platform, a single DC motor provides actuation for both arms. The arms are physically locked 180° out of phase to enforce a 50% duty cycle. The DC motor is mounted orthogonal to the ABS plastic body. Based on torque requirements determined through simulation, a 6 V Faulhaber DC Motor (# 1331 006 SR) with a Faulhaber Series 15/3 6.3:1 ratio spur gear head was chosen to drive the climber at the design target of 200 g.

The arms are driven by 5 cm gears on either side of the motor that provide an additional 3.75:1 gearing to the drive system. For the linear motion of the system, a miniature linear guide rail and bearing block (Misumi # SSEB6-40) provide minimal friction at a set sprawl angle of 10° from the saggital plane. A 10° sprawl angle is thought to provide maximal stability during steady state climbing, though a more intensive investigation is necessary [15]. A single claw was bolted directly to the end of each linear guide rail for attachment to the climbing surface. It was observed from empirical results that attachment success was related to the angle of the body from the wall, requiring an offset at the rear of the climber for improved attachment. An offset of 35 mm from the climbing surface was empirically determined to provide optimal results, as simulations neglect dynamics out of the plane. The dynamic climbing platform is shown in Figure 4.1.

While this attachment scheme presents a platform capable of locomotion on a limited number of surfaces, it decouples attachment from the demonstration of the operational capabilities of the platform. Future iterations of the platform are intended to incorporate hierarchical feet, similar to those developed at Stanford for the RiSE Project [5], to increase the adaptability and operational capability of the platform. Early experiments with our platform demonstrated that attachment success was related to the pitch of the body from the wall, requiring an offset at the rear of the climber for improved attachment. An offset of 35mm from the climbing surface was empirically determined to provide optimal results.

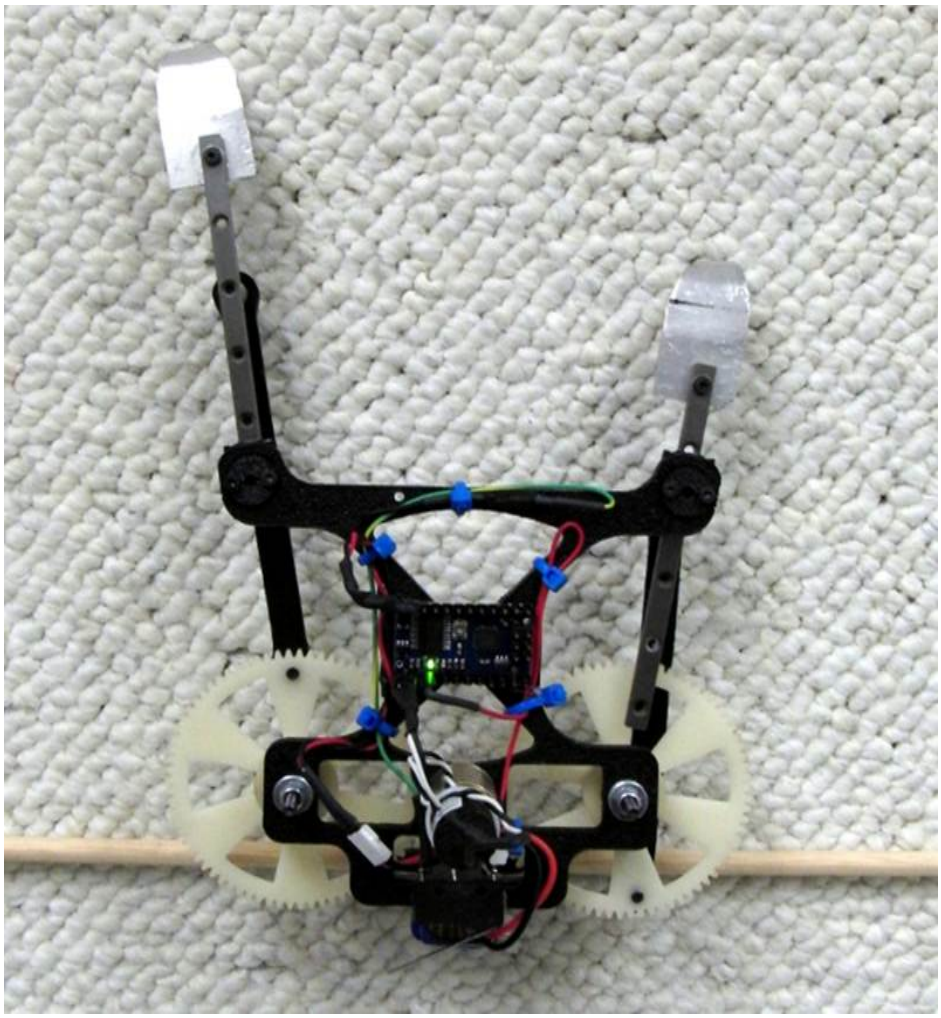


Figure 4.1: Indepedent dynamic climbing platform developed based on template presented by [28] and robotic platform developed by [15]. The climber utilizes a single actuator and four-bar mechanism with linear guide rails for linear arm motion during dynamic climbing.

CHAPTER 5

EXPERIMENTAL ANALYSIS OF SPRAWL ANGLE ON DYNAMIC CLIMBING

To evaluate the performance of the platform's scansorial motion, a 1.22m x 2.5m carpeted wall was built and oriented vertically, the wall is designed to be capable of adjusting its inclination between 90° and 50°. A motion capture system was implemented for tracking and for orientation information of the platform using two LEDs set 45 mm above and below the center of mass of the platform and a Casio Exilim Pro EX-F1 high-speed camera at 300 fps.

The platform was run at each wall inclination between 90° and 50° with sprawl angle varying between 0° and 30°, each angle being run 10 times. For each run, the platform utilized a ramp function of approximately 1 second to aid in attachment during the transition between transient and steady state operation. Initial runs of the platform did not utilize a wrist spring, instead bolting the dactyl hook directly to the linear guide rail.

The platform was first compared to the 2D Working Model dynamic simulation discussed in Chapter 3.2. The simulation utilized the same physical constraints as the platform including mass, wrist spring, frequency, and torque limitations. It was found that at a sprawl angle of 10° on a vertical wall the simulation predicted the platform's trajectory and velocity profiles closely. Figure 5.1 shows the comparison of the platform run on a vertical wall with a sprawl angle of 10° and the Working Model simulation run with the same parameters. The comparison of the simulation and platform also revealed a limping gait during steady state operation of the platform. The experimental platform was shown to climb at a vertical velocity of $37.0 \pm 3.4 cms^{-1}$ approaching the scaled velocity of a cockroach climbing at the same mass. This vertical velocity sets this miniature dynamic climbing platform as the second fastest climbing robot following the original dynamic climber, Dynoclimber, capable of velocities up to $66 cms^{-1}$ [35].

After comparing the simulation to the physical platform, 200 individual climbs were run over the number of sprawl angles and wall inclinations. The results of these runs are shown in Figure 5.2 and Figure 5.3.

The overall trends found on the physical platform show a maximum vertical velocity at a sprawl angle of 10° over all wall inclinations, as Figure 5.3 illustrates. It has also been shown that over wall inclinations between 90° and 50° the maximum vertical velocity increases with decreasing wall inclination, as Figure 5.2 shows. An expected trend which has been identified, shown in the max lateral velocity plot in Figure 5.3, is that with increasing

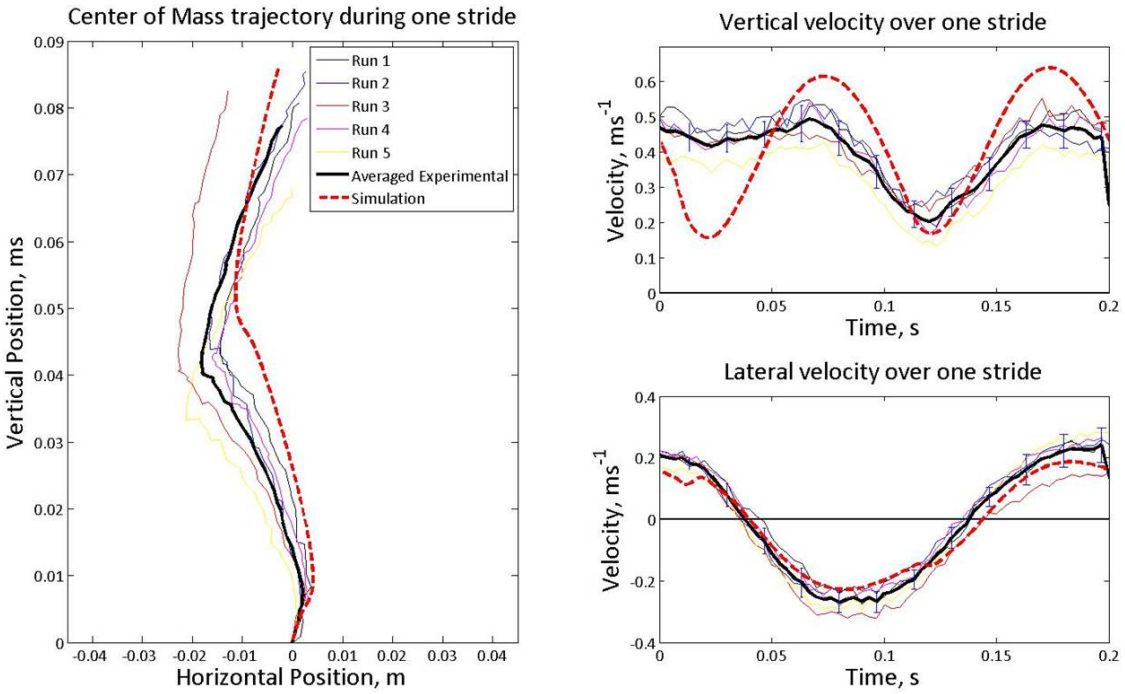


Figure 5.1: Comparison of the center of mass trajectory and velocity profiles of the Working Model 2D simulation discussed in Chapter 3.2 and the physical platform discussed in Chapter 4. The sloid line represents the averaged values of the individual runs. The dotted line represents the simulation prediction of the platform's behavior utilizing the same parameters.

sprawl angle the maximum lateral velocity increases.

The results of the experiments show a discrepancy between the experimental data and simulation data. Initially, based on the presented simulation and previous simulation studies, a sprawl angle of 30° was expected to produce the highest vertical velocity. The experimental data shows a maximum vertical velocity attained utilizing a 10° sprawl angle over all wall inclinations. This discrepancy is most likely due to increased lateral forces imparted to the linear guide block bearings as sprawl angle increases. The friction of the guide block increases as lateral force on it increases, thus reducing the overall vertical velocity. The sprawl angle of 10° most likely represents the point where increased vertical velocity due to sprawl angle can still overcome the increased friction due to the guide block, where as the benefit of increased sprawl angle on vertical velocity is lost after 10°.

A force sensor was later integrated into the climbing wall to measure the ground reaction forces of the bipedal climbing platform during steady state runs. Initial analysis of the ground reaction forces for sprawl angles of 0° and 10° showed unexpected force patterns in the fore-aft direction.

Analysis of the step during steady state climbing while attaching over the force plate, ground reaction forces compared in Figure 5.4, showed an unexpected byproduct of foregoing

a wrist spring on the platform. The compliance of the carpeted climbing wall coupled with the stiffness of the attachment point of the platform led to a fore-aft bounce during touchdown giving rise to the double hump. In order to remove this artifact, a simple wrist spring design was implemented on the linear guide rails of the platform.

Shown in Figure 5.5, the wrist utilizes space on the end of the linear guide rail to slide approximately 2 cm. A wrist spring of 131 Nm^{-1}, determined based on dynamic scaling discussed in Chapter 3.1, provides the platform the ability to remove the bounce through exerting the fore-aft force over a longer period of time. The wrist mount and slider are manufactured of ABS platstic utilizing a Rapid Prototyping machine.

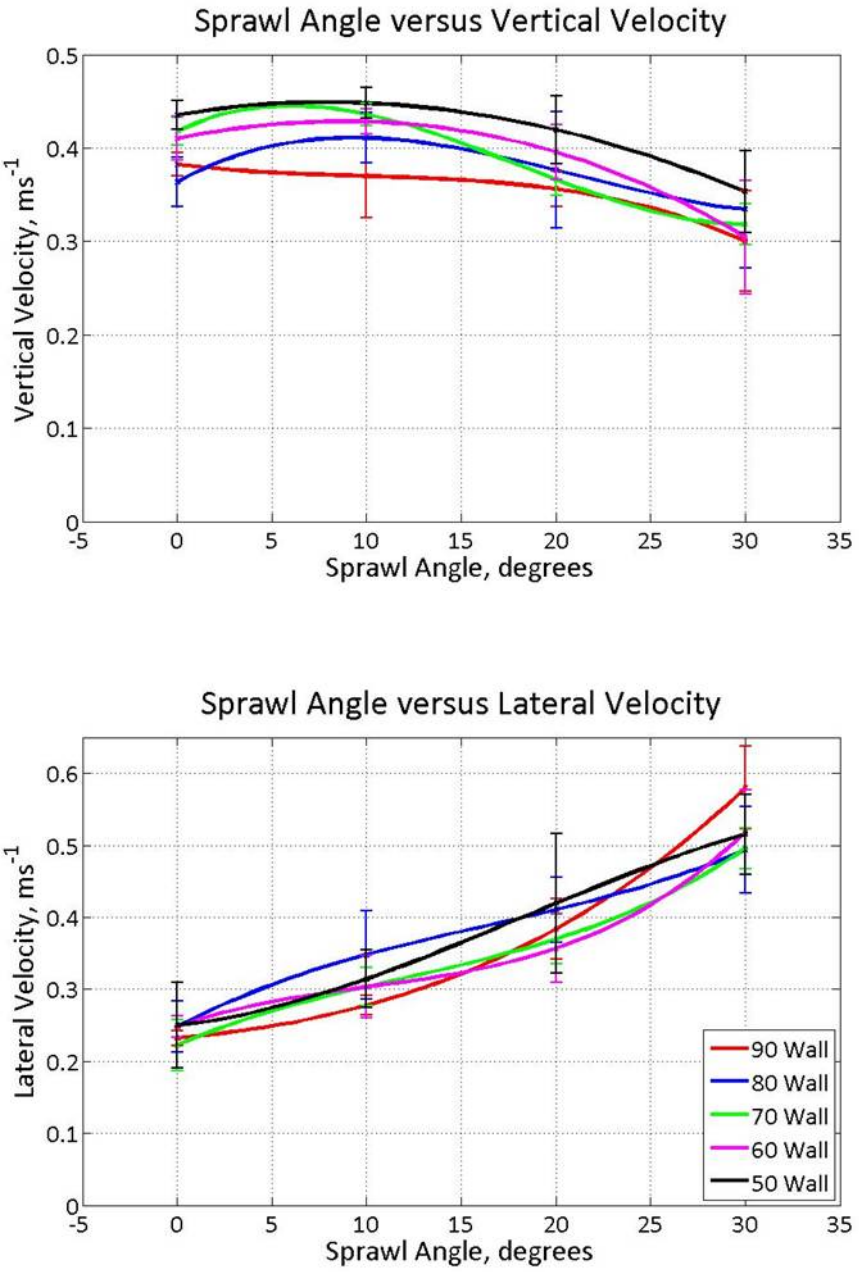


Figure 5.2: Comparison of all wall inclinations plotted over sprawl angle. **(Top)**Sprawl angle versus vertical velocity plotted over all wall inclinations. Over all wall inclinations, a local maximum trend is visible at a sprawl angle of approximately 10°. An overall trend also shows an increase in vertical velocity with decreasing wall inclination, 50° showing the fastest vertical velocities. **(Bottom)**Sprawl angle versus maximum lateral velocity plotted over all wall inclinations.

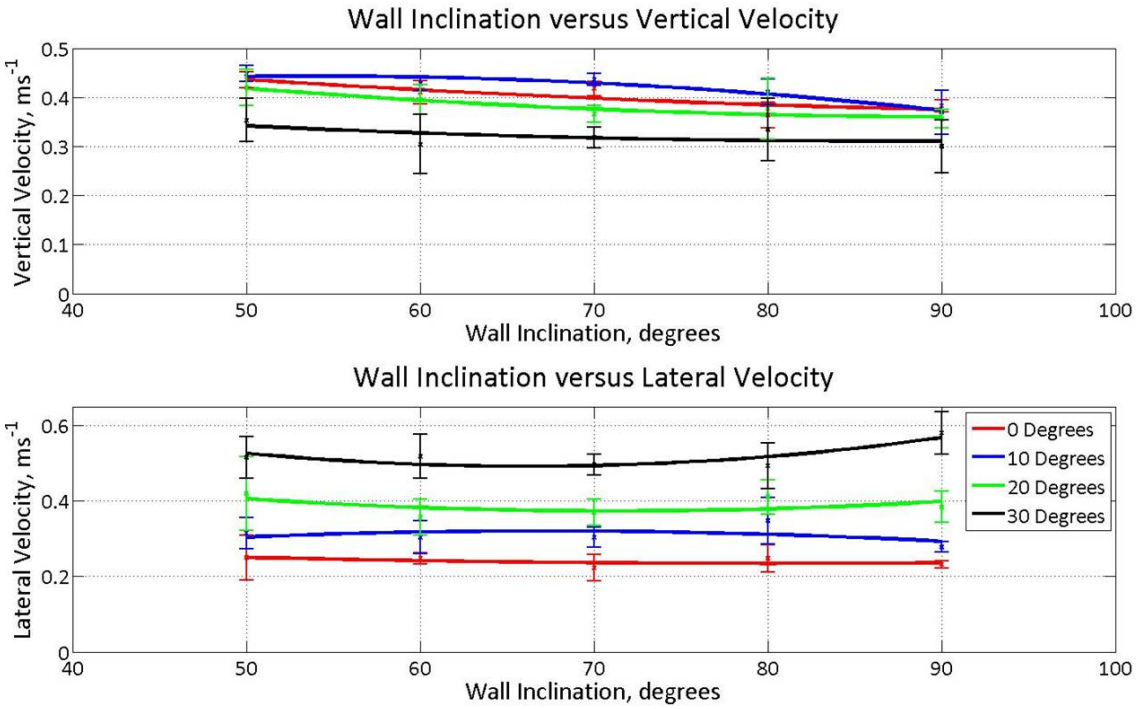
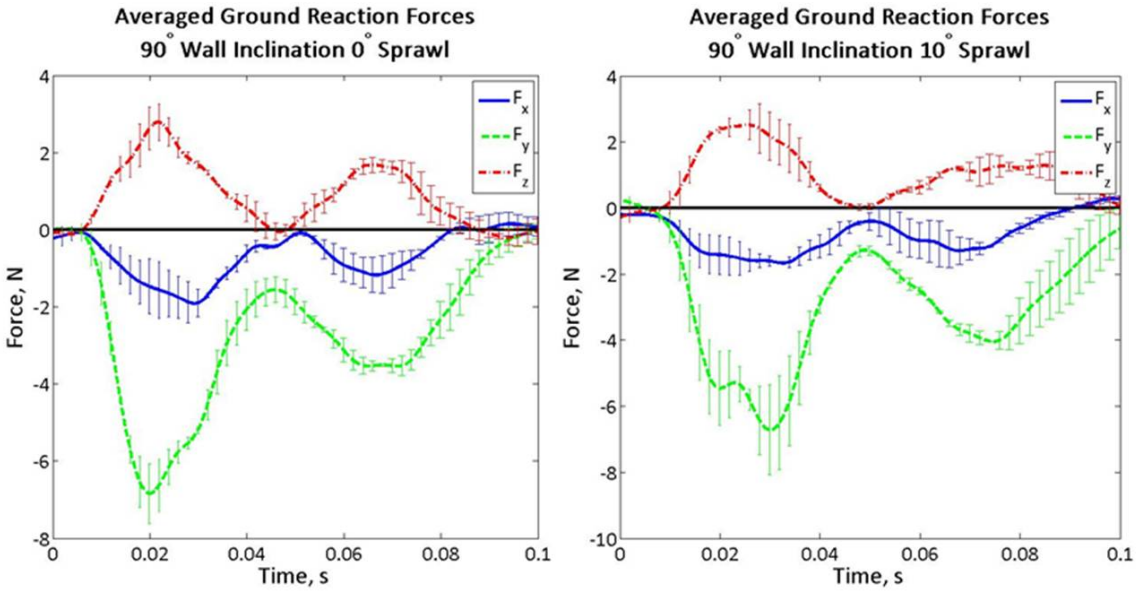


Figure 5.3: Comparison of wall inclination versus vertical velocity and max lateral velocity plotted over all sprawl angles. **(Top)** Wall inclination versus vertical velocity plotted over all sprawl angles. The trends show overall a maximum velocity over all wall inclinations at a sprawl angle of 10°. **(Bottom)** Wall inclination versus max lateral velocity plotted over all sprawl angles. An overall trend of increasing lateral velocity with increasing sprawl angle is found.



(a) Averaged ground reaction forces for 0° sprawl angle during a single step.

(b) Averaged ground reaction forces for 10° sprawl angle during a single step.

Figure 5.4: Comparison of ground reaction forces generated during a run on a vertical wall utilizing sprawl angles of **(a)** 0° and **(b)** 10°. Both plots show a double hump profile in the fore-aft direction (F_y) during steady state climbing.

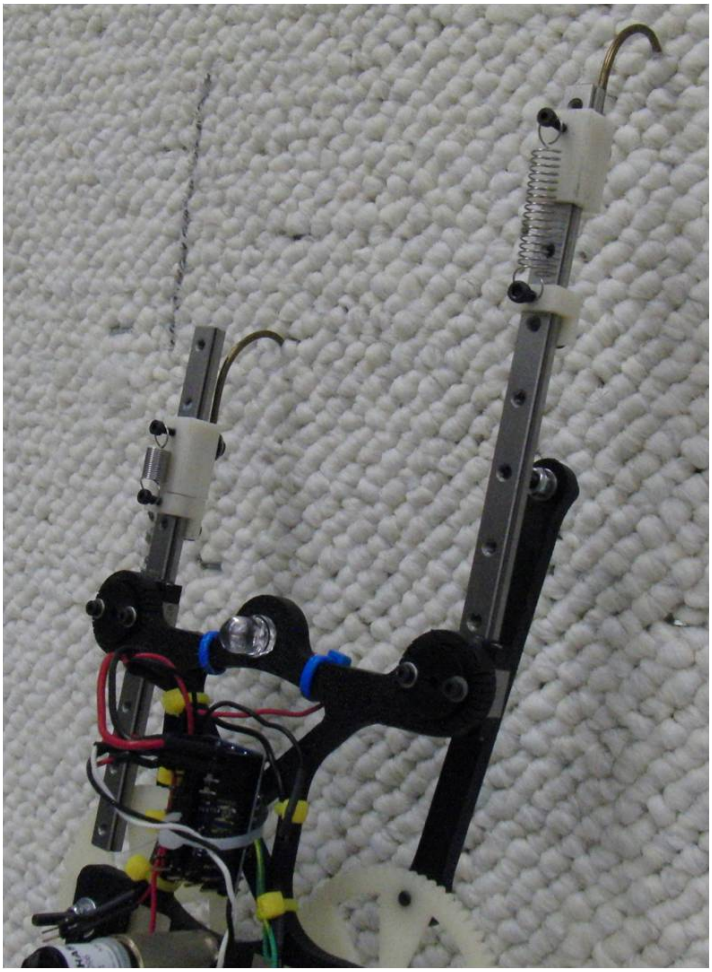


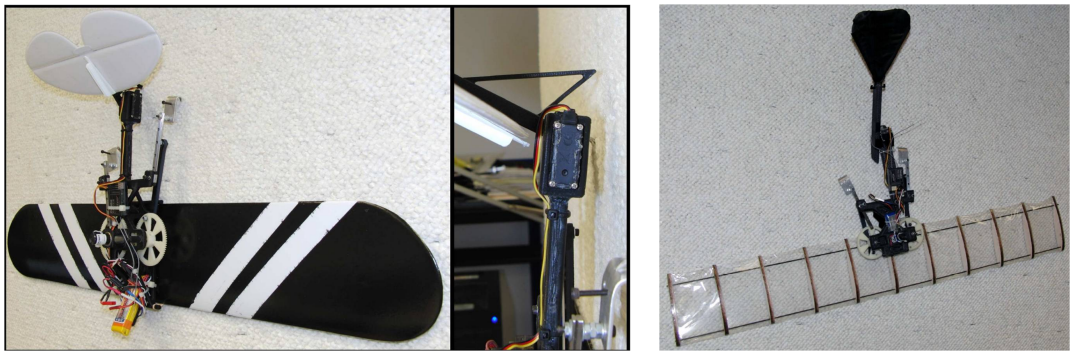
Figure 5.5: Wrist spring design addition to the linear guide rails. The slider and mount are manufactured of ABS plastic utilizing a Rapid Prototyping machine. The wrist spring utilized is 131 Nm^{-1} determined based on dynamic scaling of the F-G template. The hook, which is bolted to the slider, is able to slide approximately 2 cm during steady state operation leading to smoothing of the fore-aft ground reaction forces.

CHAPTER 6

PLATFORM DESIGN OF MULTI-MODAL ROBOT

6.1 Fixed Wing Glider Design

Early instantiations of the platform focused on incorporating an airfoil on a dynamic climbing platform following a minimalist design approach. These versions of the ICAROS platform attempted to utilize existing, commercially available wings for integration with the climbing platform, as discussed in [22] and shown in Figure 6.1a. The styrofoam RC airplane wings utilized weighed 50.6g with a wing span of 80 cm and a chord length of 12cm but did not generate sufficient lift for the integrated platform. Shown in Figure 6.1b, a skeletal wing design of the same size utilizing a Clark Y airfoil was developed which reduced weight. These early versions of the platform did not generate sufficient lift to reach glide ratios greater than 1, thus a different design strategy was adopted for the design and integration of the components of the multi-modal platform.



(a) Version 1 (b) Version 2

Figure 6.1: (a)Version 1 of the ICAROS platform that utilized a rapid prototype ABS plastic body and rigid attachment points for an underbody styrofoam RC airplane airfoil as well as an actuated tail stabilizer. (b)Version 2 of the ICAROS platform with rigid attachment point for Clark Y airfoil skeletal wing and an actuated tail servo.

The current platform was designed utilizing a strategy of incorporating a dynamic climbing platform within a manufactured rigid airfoil, rather than the initial design approach of adding an existing airfoil to a dynamic climbing platform utilized in [22]. This approach reduced drag and maximized lift while minimizing the addition of mass as much as possible. To improve lift at low speed, airfoils designed for high-lift applications were manufactured. A skeletal rib design utilizing planar Eppler 422 airfoils was integrated onto a primary wingspan spar of carbon fiber and dense foam. A stabilizing spar consisting of a 1.6mm carbon fiber rod was placed at the rear of the airfoil while 0.8mm balsa wood provided stability and a rigid leading edge on the front of the airfoil. The skeletal wings were skinned using a lightweight shrink wrap covering, Coverite Microlite^{TM}, with a density of only $22gm^{-2}$. Based on the mass of the platform, a wing area of $0.25m^2$ was selected, with a wing span of 0.72m and chord length of 0.18m. While the manufactured wings are larger than the commercially available wings originally used for integration, their mass was only 49.2g compared to 50.6g and they provide a higher lift to mass ratio.

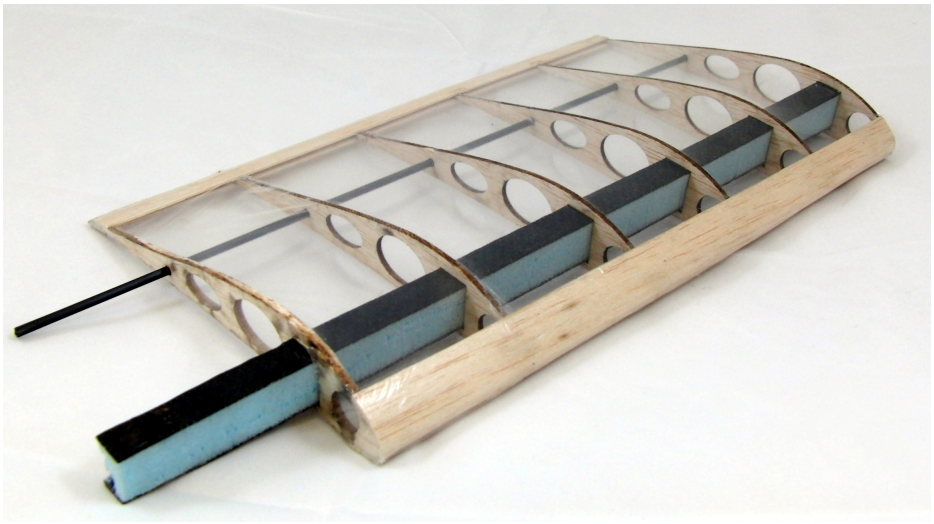
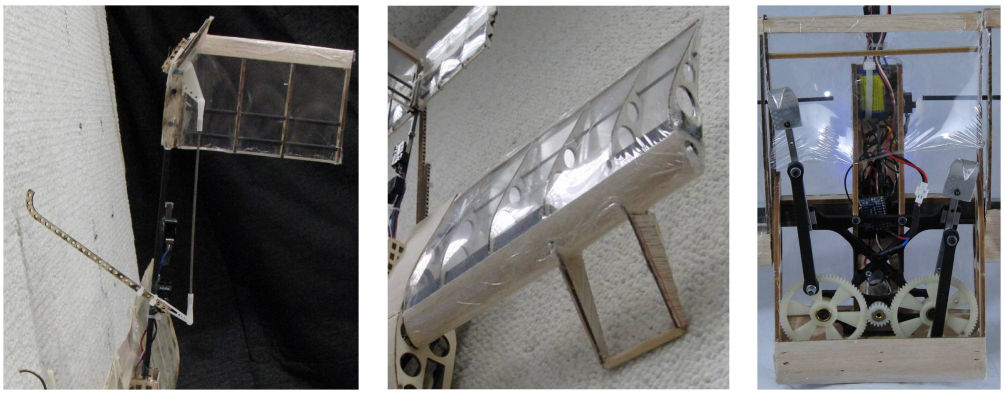


Figure 6.2: Skeletal wing design with Coverite Microlite^{TM} skin. Central blue and black spar is a dense foam and carbon fiber composite for lightweight structural support. Overall wingspan is 0.72 m with a total wing weight of 49.2 g providing a higher lift to weight potential than commercially available wings utilized in early instantiations of the platform.

To reduce the overall drag on the glider, the fuselage was designed similar to an airfoil shape for streamlining, though the size of the climber platform constrained its dimensions. The fuselage was required to be a minimum of 0.14 m wide to encapsulate the drive gears for climbing as well as a minimum of 0.04 m tall to allow for the proper body offset from the wall for climbing. The overall length of the fuselage was based on the position of the center of mass of the integrated platform with relation to the airfoil. The center of mass was located 1/3 of the chord length from the front of the leading edge of the wing.



(a) Detachment mechanism (b) Wing pontoon (c) Glider offset

Figure 6.3: (a)Detachment mechanism for platform that utilizes the actuation method for the rear tail elevator. (b)Wing pontoon designed to replace roll stabilization bar initially utilized in independent climber design. (c)The glider body replaced the body offset utilized in the independent climber to provide the proper body orientation for dynamic climbing.

6.2 Integration of Platform Components

The efficient use of mass was the primary focus of integration of the two forms of locomotion utilized in this platform. The first consideration employed for mass mitigation was in the design of the fixed-wing glider assembly. Lightweight materials including balsa and bass wood, carbon-fiber reinforced foam, and lightweight plastic wrap were utilized to minimize mass while maintaining structural integrity. The design of the dynamic climber portion of the platform also reduced structural components to minimum geometries and stiffnesses necessary for stable dynamic climbing.

The second consideration for mass mitigation was the intelligent combination of structures for both forms of locomotion. These structures were designed to serve multiple functions, not only for both modes of locomotion, but also for transitioning between the two. The fixed wing glider was designed to provide the required offset from the wall for the climbing platform as well as covering the components of the climber in a streamlined body to improve lift and decrease drag. The stand-alone dynamic climber required a 60 cm roll stabilization bar at the posterior of the platform. To replace this bar, a pontoon design extending from the leading edge of the wings provided the necessary offset and distance for roll stabilization, eliminating a stand-alone horizontal bar. The transition mechanism employed on the platform utilized the actuation method for the rear tail elevator and an extension bar, shown in Figure 6.3a. Actuation of the bar towards the wall pushes the platform's feet off of the climbing surface and initiates a ballistic dive. Figure 6.4 shows the integrated platform perched on a vertical climbing surface. The physical parameters of the ICAROS platform are given in Table 6.1.

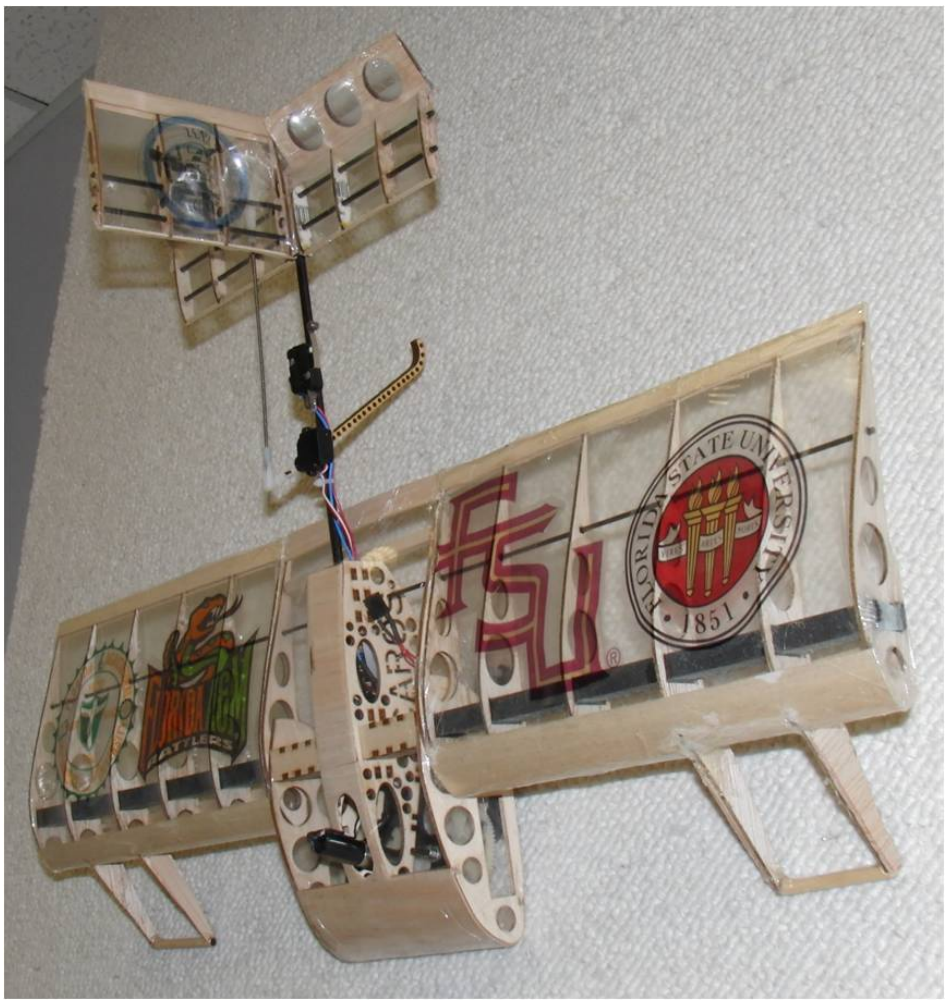


Figure 6.4: Integrated ICAROS platform perched on vertical climbing wall. Pontoons extending from the leading edge of the wings provide roll stability during dynamic climbing.

6.3 Mass Constraints

Initial designs of the integrated platform specified a total mass of 200 grams. This target was unable to be met due to manufacturing constraints on the development of the fixed wing airfoil design for gliding as well as the actuation constraints of the climber. The climbing platform developed for the system independently weighs 185 grams. The utilization of a traditional, fixed-wing airfoil and streamlined fuselage led to an independent glider mass of 190 grams. The combination of the two subsystems allowed for a reduction in overall mass of 25 grams due to utilization of structures for multiple functions. The final platform mass of 350 grams reflects a necessary trade off for multi-modal operation but negatively impacts the expected behavior of the system. The effect of increased mass on expected vertical stability and velocity was analyzed utilizing an existing dynamic climbing

Table 6.1: Physical Parameters of ICAROS platform

Fuselage Size	0.25 x 0.14 x 0.04 m
Platform mass	350 g
Wing span	0.72 m
Chord Length	0.18 m
Wing load	$26.5 Nm^{-2}$
Motor	Faulhaber Series 1331 006 SR
Gear head	Faulhaber Planetary Gearhead Series 14/1 14:1
Spur gear	3.75:1 reduction
Leg stroke	0.051 m

simulation presented in [15]. Based on the simulation, it was found that increasing mass will decrease vertical velocity to $0.10\frac{m}{s}$, but the platform will still maintain stability and expected climbing behavior.

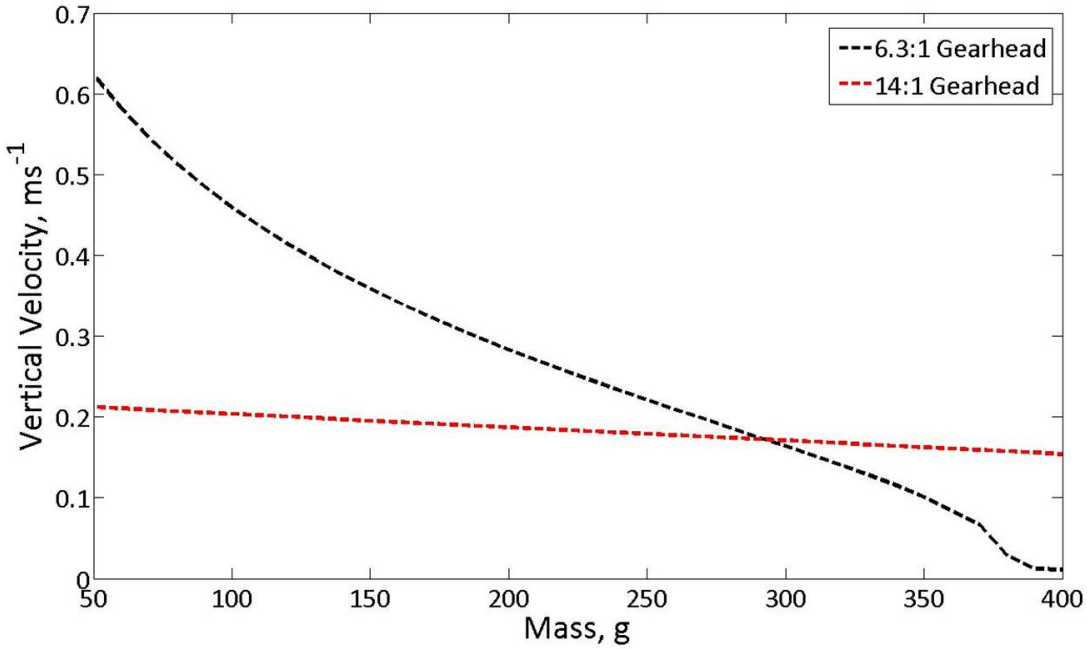


Figure 6.5: Effect of increasing mass on vertical climbing velocity for the ICAROS platform utilizing a Faulhaber 6.3:1 spur gear head (solid line) and a Faulhaber 14:1 planetary gear head (dotted line) on a 6 V Faulhaber DC Motor (# 1331 006 SR) as simulated utilizing the simulation presented by [15].

Due to the increased mass, it was desirable to utilize a larger gear reduction for the motor selected to increase torque. The relationship between vertical climbing velocity and

mass is shown in Figure 6.5 as well as the vertical velocities of the platform utilizing the original Faulhaber Series 15/3 6.3:1 ratio spur gear head and the replacement Faulhaber Series 14/1 14:1 ratio planetary gear head. Utilizing the 14:1 gear reduction, the simulation predicts an improved vertical velocity of $0.17ms^{-1}$ at a platform mass of 350 g.

6.4 Electronics

The ICAROS platform was designed to initially use minimal control and sensing to reduce the mass of the overall system and simplify the electronics configuration. The requirements of the system called for a programmable controller capable of driving one brushed DC motor, and a servo, as well as a number of ports for sensing. The controller integrated into the platform is the Pololu Baby Orangutan B-328 Robot controller. The Baby Orangutan includes a 20 MHz ATmega328p AVR microcontroller, a dual H-bridge for control of two DC motors, 18 I/O lines, and an extensive library including the ability to drive servo motors. This microcontroller incorporates all of these specifications into a board that is 30 mm x 18 mm and weighs 3.5 g.

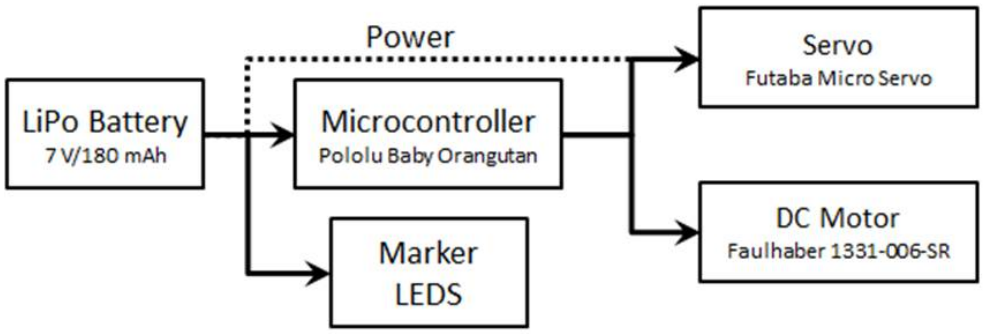


Figure 6.6: Representation of the electronics control hierarchy for the integrated platform. The on-board Lithium Polymer battery provides direct power for the Baby Orangutan, two LEDs, and Futaba Micro Servo. The microcontroller provides power and control for the brushed DC motor and a position for the Futaba Micro Servo.

The DC climbing motor is run directly from the on-board motor driver of the Baby Orangutan. The battery provides power for the two Super Bright LEDs used for motion tracking. The servo used for control of the tail elevators draws power directly from the on-board battery while receiving position commands from one of the output ports of the microcontroller. A 2-Cell, 7.4 V 180 mAh Lithium Polymer battery weighing 12 g provides power.

CHAPTER 7

EXPERIMENTAL RESULTS

The climbing, transition, and flying performance of the combined system were characterized independently in order to determine the performance with added mass and prior to developing a control architecture for the system. To accomplish this, the final, integrated platform was run scansorially on a prepared climbing surface, dropped from a fixed height, and flown using a launcher in a horizontal orientation. A motion capture system was implemented for tracking and for orientation information of the platform using two LEDs and a Casio Exilim Pro EX-F1 high-speed camera at 300 fps. These independent sets of system data allowed for development of control policies for combined operation.

7.1 Climbing Characterization

Climbing characterization tests were performed on a vertical climbing surface with the final integrated platform. To evaluate the performance of the platform's scansorial motion, a 1.22m x 2.5m carpet wall was built and oriented vertically. To track the position and orientation of the platform, LEDs were located 25 mm above and below the center of mass of the platform along its spine. The high-speed camera was set 2 m from the climbing wall and aligned with the climbing surface to capture high speed video of each run.

The climbing platform was run twenty times on the climbing surface and each run was analyzed over a 1 second interval, approximately two strides, at the same position in the steady state run of the platform. The empirical results were compared to the dynamical simulation presented by [15] for verification. Figure 7.1a shows the comparison of the center of mass trajectory for steady state climbing of the platform and the simulation. At 350 grams, we expected $0.17 ms^{-1}$ but the average vertical velocity of the platform attained was $0.135 \pm 0.01 ms^{-1}$. Figure 7.1b shows the discrepancy between the vertical velocities of the simulation and experimental results. This discrepancy is most likely attributed to lost vertical displacement due to the compliance of the foot connection on the climbing wall. Figure 7.1c shows the horizontal velocity of the simulation closely predicts the behavior of the system.

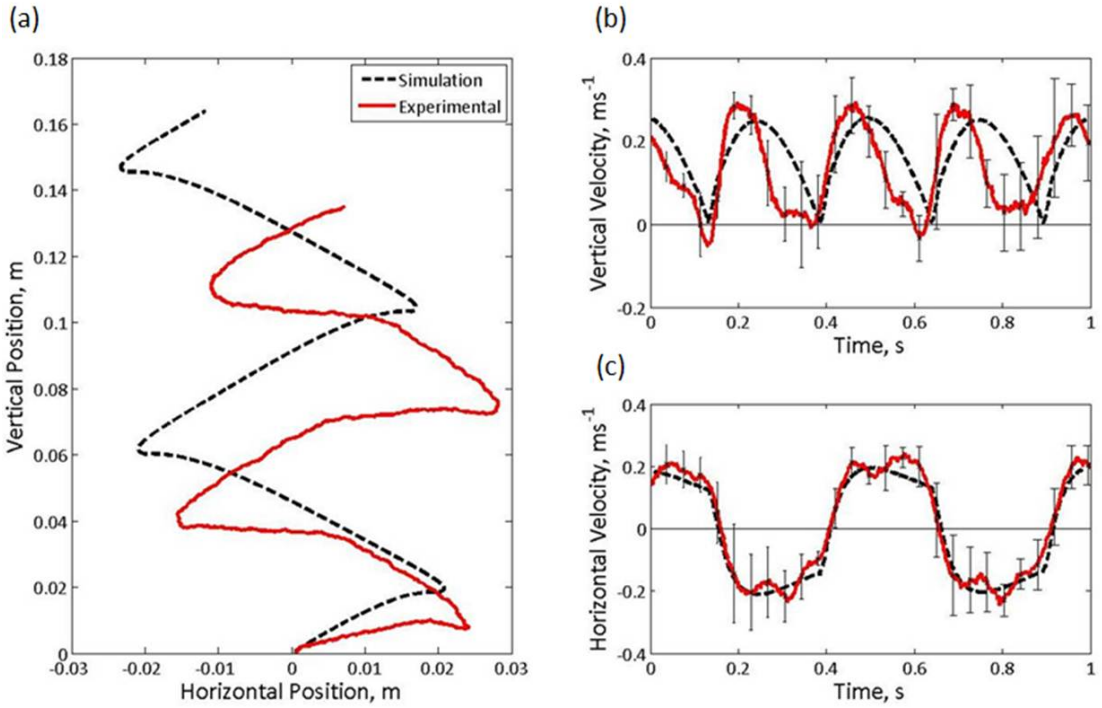


Figure 7.1: Comparison of steady state climbing of dynamical simulation and robotic platform over 1 s or two strides. The solid line represents the experimental results of the platform while the dotted line represents the simulation using the same physical parameters as the final platform, e.g. mass and motor constraints. The center of mass trajectory (a) over the 1 s climbing interval shows the experimental results closely follow the predicted behavior from the numerical simulation. The vertical velocity (b) and horizontal velocity (c) of the integrated platform shows a similar speed and frequency to the numerical simulation.

7.2 Flight Characterization

In order to determine the glide velocity and glide ratio as well as to develop a control system capable of maximizing glide properties, the integrated platform was characterized in horizontal glide experiments. The horizontal gliding ability of the fixed wing glider was evaluated using the same motion capture system used for climbing as well as a custom built launching system. An elastic driven carriage launched the platform horizontally at a controlled orientation and velocity with two LEDs mounted on the outer edge of the wing airfoil to provide information on position and orientation of the glider. The two LEDs were located on the airfoil to place the center of mass halfway between them to simplify calculations. The position, orientation, velocity, and acceleration were calculated from the motion capture data and analyzed using MATLAB. The angle of the tail elevator was varied between 0° to 6° to determine the maximum glide slope of the platform.

Figure 7.2 shows a schematic depicting the relevant forces and angles of the wing. The

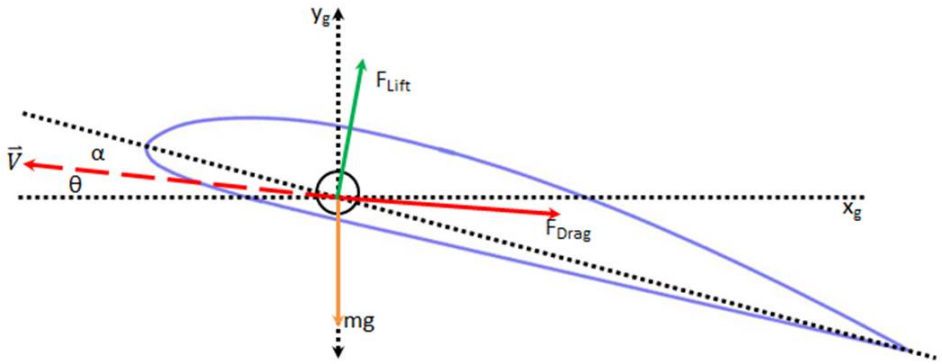


Figure 7.2: Free body diagram of aerodynamic forces on wing. θ represents the angle of velocity with respect to the horizontal; α represents the angle of attack from the velocity vector. The airfoil is set in a global coordinate frame, X_g and Y_g.

angle θ represents the angle the velocity vector makes with the horizontal; the angle α represents the angle of attack, or the angle of the airfoil with respect to the velocity vector. Utilizing Newton's second law, the equation relating vertical forces,

$$\Sigma F_y = ma_y = F_{Lift}\cos(\theta) - F_{Drag}\sin(\theta) - mg \tag{7.1}$$

was developed. Rearranging the equation to solve for the force of drag as a function of the lift force gave:

$$F_{Drag} = \frac{-ma_y + F_{Lift}\cos(\theta) - mg}{\sin(\theta)}. \tag{7.2}$$

Following the same procedure relating horizontal forces,

$$\Sigma F_x = ma_x = -F_{Lift}\sin(\theta) - F_{Drag}\cos(\theta) \tag{7.3}$$

and rearranging the equation to solve for the force of lift gave:

$$F_{Lift} = \frac{-ma_x - F_{Drag}\cos(\theta)}{\sin(\theta)} \tag{7.4}$$

Substituting the derived value for the force of drag resulted in

$$F_{Lift} = (\frac{1}{1+\frac{\cos(\theta)^2}{\sin(\theta)^2}})(\frac{-ma_x}{\sin(\theta)} + \frac{ma_y\cos(\theta)}{\sin(\theta)^2} - \frac{mg\cos(\theta)}{\sin(\theta)^2}). \tag{7.5}$$

Utilizing the calculated forces from the motion captured data, the coefficient of lift

$$C_L = \frac{F_{Lift}}{\frac{1}{2}\rho V^2 A} \tag{7.6}$$

was able to be calculated for the three tail elevator cases, where ρ is air density, V is the air stream velocity over the wing, and A is the plan area of the wings [1].

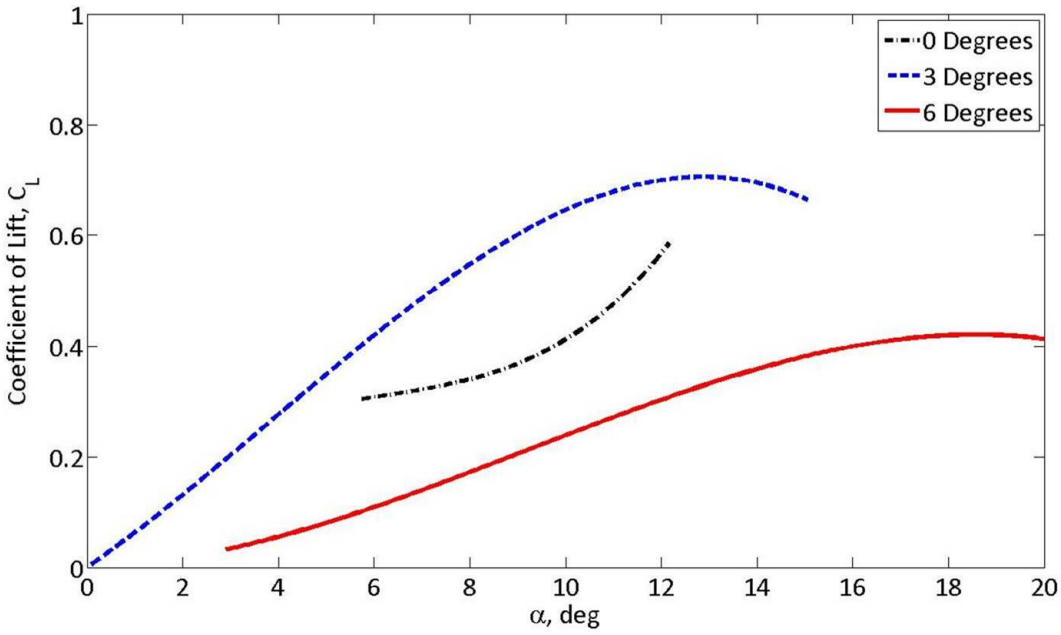


Figure 7.3: Angle of Attack versus Coefficient of Lift for three tail elevator angles averaged over a number of runs. The R^2 value is 0.62 for 0°, 0.75 for 3°, and 0.56 for 6°.

Since the aerodynamic properties of the platform were evaluated for a complete glider assembly, the results of the characterization differ from typical aerodynamic properties of airfoils examined in controlled wind tunnel tests. The calculated coefficient of lift, shown in Figure 7.3 versus angle of attack, was shown to have a maximum with a 3° tail elevator angle over a range of angles of attack. The R^2 values for the coefficient of lift curve were 0.62 for 0°, 0.75 for 3°, and 0.56 for 6°. While the correlations are far from ideal in this analysis only a relative comparison between the tail angle aerodynamics was necessary to determine a 'best option' for operation.

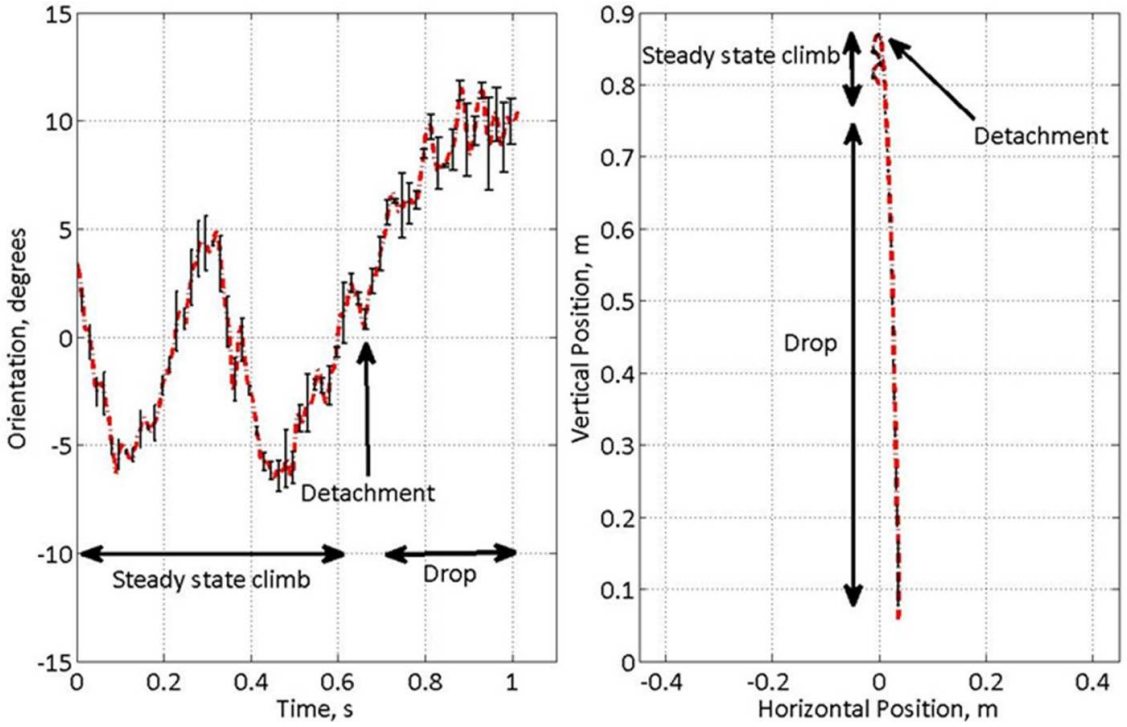
The Eppler 422 airfoil is expected to produce maximum lift at an angle of attack of approximately 15°; a 0° tail elevator angle maintained an angle of attack between approximately 5° and 12° while a 3° tail elevator angle maintained an angle of attack closer to 15° for a longer portion of its flight. A 6° tail elevator angle provided a larger range of angles of attack during the flight path, but entered stall on each run due to the severe pitch of the airfoil. Based on this characterization of the glider platform, a tail angle of approximately 3° was determined to be optimal for maximum glide range after the platform had transitioned into a horizontal glide path. The characterization of the platform glide behavior also determined the glide velocity to be $5.3 \pm 0.23 ms^{-1}$.

7.3 Transitioning

Transitioning between forms of locomotion presents the greatest challenge for multimodal platforms. The characterization of this platform's transition encompasses two identified stages: detachment from the climbing surface and diving. These two stages were analyzed independently utilizing the motion capture system previously described.

7.3.1 Detachment

The platform was run on a vertical climbing surface. At the end of its climbing run, the platform actuated its detachment mechanism to initiate the ballistic dive, as discussed in Section 7.3.2.



(a) (b)

Figure 7.4: (a) Platform orientation immediately preceding and following the actuation of the detachment mechanism. (b) Platform trajectory immediately preceding and following the actuation of the detachment mechanism. The steady state climbing portion shows the same trajectory illustrated in Figure 7.1. Once the platform initiates detachment from the surface, it falls vertically, deviating only slightly to the right.

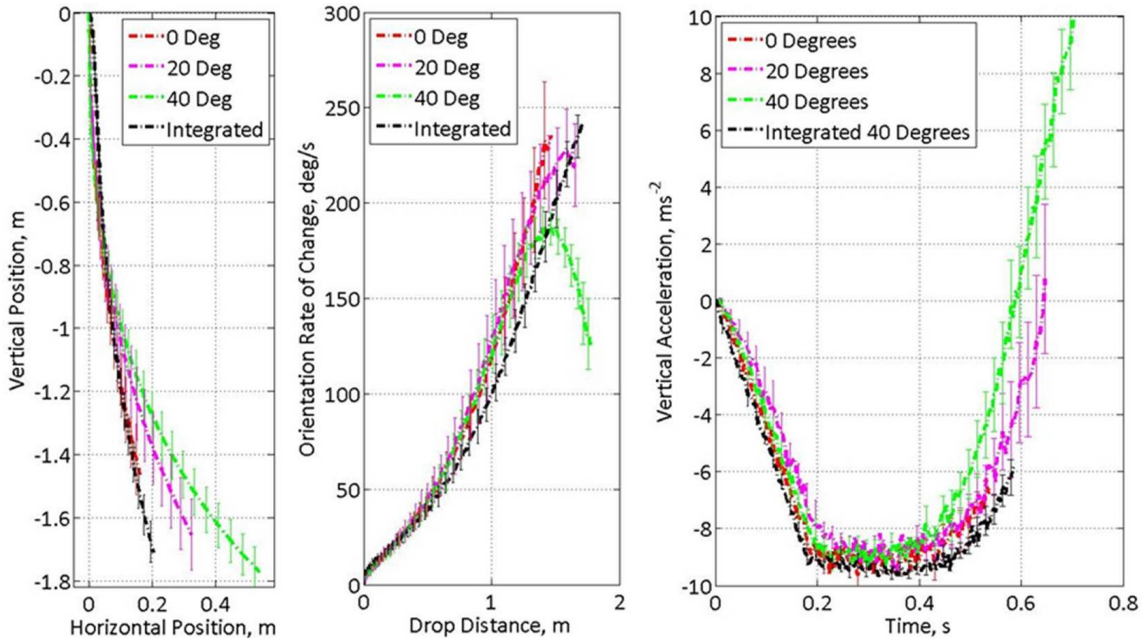
The platform showed consistent detachment from the climbing surface utilizing the actuated boom. Figure 7.4a shows the platform orientation and Figure 7.4b trajectory immediately preceding and following actuation of the detachment mechanism. The platform

showed a slight yaw of approximately 10° following detachment. This yaw can most likely be attributed to inconsistent foot detachment from the surface.

7.3.2 Ballistic Dive

The platform utilizes the simplest mechanism possible for transitioning to a horizontal glide slope: a passive drop from the climbing surface before actuating the tail elevator to initiate a pitch up. This behavior enables the platform to demonstrate multi-modal operation; however, in the future active launching mechanisms will be explored to decrease the launch height necessary.

The drop transition was characterized utilizing a 200g glider platform as well as the fully integrated platform. Each platform was dropped from a height of 2.4m with a fixed tail angle between 0° and 40° incrementing each time by 20°. The tail elevator angle of 40° represents the maximum actuation possible utilizing the current design. Each trial was dropped from rest. This procedure gives a qualitative characterization for the actual drop, though it is expected that increased air velocity over the airfoils will produce a faster pitch up maneuver due to the increased lift.



(a) (b) (c)

Figure 7.5: Results of transitioning from a dive to horizontal gliding. (a) Trajectory of platform during drop for tail elevator angles between 0° and 40°. Initial orientation for each drop is vertical (90°). (b) Rate of orientation change over vertical drop distance for the four tail elevator cases. (c) Vertical acceleration profiles for each tail elevator case plotted over the drop time.

The trajectory of the three tail elevator angles, shown in Figure 7.5a, shows that in-

creasing the tail angle decreases the vertical height necessary to achieve the same platform orientation. The rate of change in orientation of the airfoil, shown in Figure 7.5b, shows that 40° is reaching a steady-state at the bottom of the drop as the rate of change begins to drop precipitously. The vertical acceleration of the platform is changed significantly utilizing the maximum elevator angle, as shown in Figure 7.5c.

The results of the fully integrated platform with a 40° tail elevator angle drop tests show a similar trajectory and acceleration profile as a 0° tail elevator angle of the un-integrated glider. The acceleration profile, following similar trends as the un-integrated platform, can be expected to reach $0ms^{-2}$ vertical acceleration within 1 s of initiating the pitch up maneuver, as extrapolated from Figure 7.5ca. Based on this analysis, the platform is commanded to actuate the rear tail elevator angle to 40° for 1 second to achieve a $0ms^{-2}$ vertical acceleration to enter a stable, horizontal glide path.

7.4 Multi-modal Operation

The control system was developed to provide a stable and fast climbing phase, a transition phase, and glide phase. Using the results of the system characterization, a simple time-based control system was implemented on the on-board micro controller unit. The platform was desired to climb a distance of approximately 2m during operation. Using the characterized climbing velocity of $0.135ms^{-1}$, a time of approximately 15s is required. To aid in reduction in transient system responses, a simple velocity ramp was incorporated into the initialization of climbing. After 20s of climbing has elapsed, the platform initiates the transition phase to detach from the climbing surface.

To detach from the climbing surface, the platform actuates the servo to its maximum extent towards the wall, pushing the detachment boom into the surface and lifting the claws off the wall. The platform enters a ballistic dive immediately following detachment from the wall. Based on the glide characterization presented in Section 7.2 the platform's optimal glide velocity was found to be $5.3 \pm 0.23ms^{-1}$. Utilizing a simple ballistic motion model, it was determined that the platform should drop for approximately 1 second to develop the desired velocity for optimal gliding ability. The tail elevator is actuated to approximately 40° for 1 second to transition from a ballistic dive to a horizontal glide, as determined in Section 7.3.2. Based on the results of the flight characterization, the tail elevator is returned to a angle of 3° for optimal glide distance.

The integrated platform operation required a large space for climbing and transitioning to gliding flight. Platform operation was conducted within a large atrium of a three story building to provide the necessary height for transitioning but remove environmental factors such as wind. The length of the atrium restricted the maximum glide ratio to 2. A 1.22m x 2.5 m carpet wall was mounted on the third floor to provide a climbing surface for the platform to operate on.

A high speed camera system was setup to capture the entire sequence in one frame to demonstrate the abilities of the platform. Figure 7.6 shows a time lapse image of the single frame over the course of a single run during operation of the platform. The platform was run in the defined operation for 10 runs to verify reliability. Over the course of these runs, the platform was found to attain a glide ratio of approximately two as it reached the

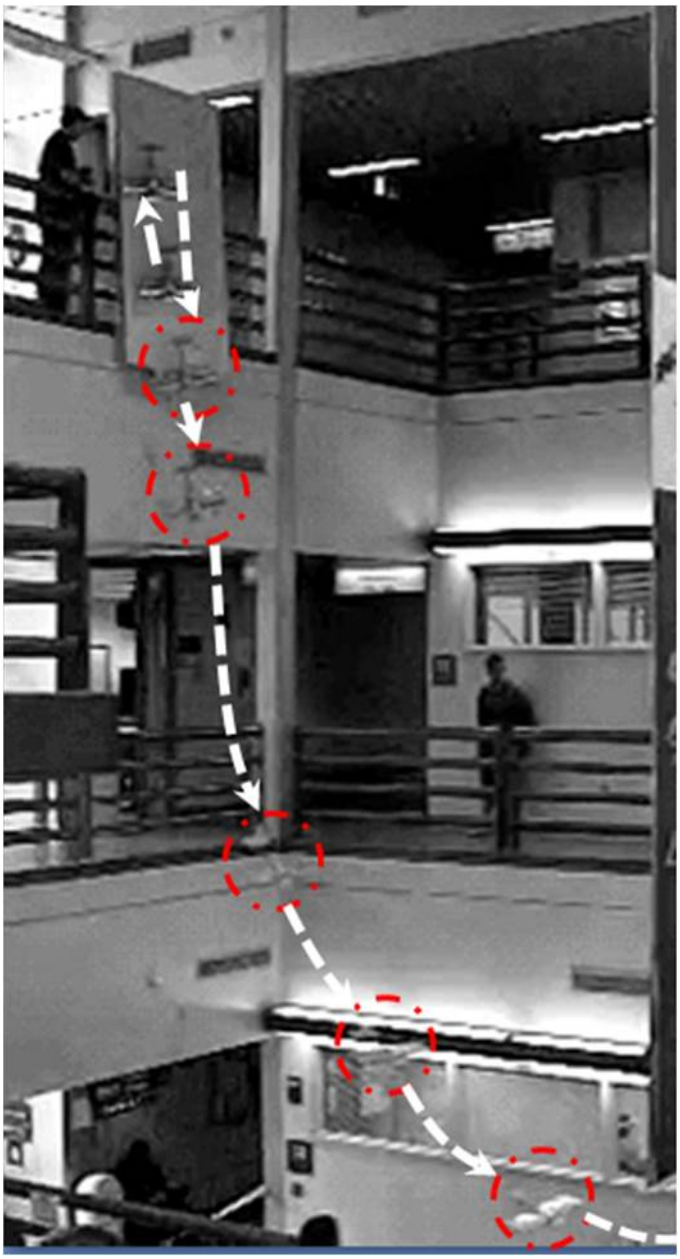


Figure 7.6: Time lapse image of the multi-modal operation of the ICAROS platform. The platform climbs vertically on a prepared climbing surface before detaching and entering a ballistic dive. The platform actuates its rear elevator to transition to a level, horizontal glide path.

maximum glide distance attainable within the test site 80% of the time.

ICAROS has shown the ability to vertically climb prepared surfaces at speeds of $13.5 \pm$

$1.0cms^{-1}$, exceeding vertical climbing speeds of most previous climbing platforms. The initial design criteria developed based on biological analogs for the ICAROS platform suggested a wing loading of $28Nm^{-2}$, a glide ratio of 1.99 ± 0.33 and a glide velocity of $7.11 \pm 1.09ms^{-1}$. The integrated platform's wing loading was $26.5Nm^{-2}$, very close to the biological data, while the glide ratio was approximately 2. The glide velocity of the platform was $5.3 \pm 0.23ms^{-1}$. The glide velocity was less than the targeted glide velocity based on biological analogs. This reduction in velocity is most likely attributable to excess drag on exposed climbing components.

CHAPTER 8

CONCLUSIONS

While animals have shown the ability to adapt between multiple forms of locomotion easily and efficiently, few robots have been able to demonstrate similar flexibility. As with animals, the design flexibility that enables multi-modal operations comes at a price, specifically weight and reduced performance of individual locomotion modalities. The ICAROS platform described in this paper has demonstrated the ability to both climb and glide. While other robotic platforms can climb faster or fly more agilely, none have yet combined these two forms of locomotion into a single platform with performance similar to biological counterparts.

A 2D dynamics simulation representing the mechanical configuration of the proposed robot was developed and run on varying wall inclinations and sprawl angles. This simulation predicted similar results as previous simulation studies on sprawl angle, primarily, a sprawl angle of 30° produces the fastest climbing velocity and the max lateral velocity increases with increasing sprawl angle. This simulation was also utilized to predict the effects of increased mass and gearing differences on the integrated platform.

Initial experimental runs with the developed miniature dynamic climbing platform found correlations in body pitch angle and passive compliance with reliable attachment. A optimal pitch angle for the climber was found empirically. Analysis of ground reaction forces on the platform running steady-state showed a double peak in the fore-aft forces. This pattern was found to be due to the lack of passive compliance in the wrist of the climber. To increase attachment reliability and removed the double peak, a passive compliant wrist with stiffness based on dynamic scaling of the Full-Goldman template was designed and integrated on the climber platform. Following these changes, the climber was found to run at high speed with high reliability.

This platform also represents the smallest dynamic climbing platform built to date. Utilizing the miniature bipedal dynamic climbing platform, a connection between sprawl angle and vertical and lateral velocities has been identified. The independent dynamic climbing platform has demonstrated steady-state climbing at vertical velocities of $37.0 \pm 3.4 cms^{-1}$ making it the second fastest climbing platform, absolute, after Dynoclimber [35]. Initial simulations predicted the fastest velocities at a sprawl of 30° but through the presented experimental analysis, it was found a sprawl angle of 10° enables the fastest vertical climbing velocities. This discrepancy between simulation and experimental results can be attributed to the un-modeled friction in the linear guide rails and blocks as sprawl angle increases.

As predicted through simulation, the climbing velocity of the platform is severely reduced due to increased mass. Even with an effective payload of 150 grams, the final experimental climbing velocity of $0.135 \pm 0.01 ms^{-1}$ exceeds the vertical velocities of most scansorial platforms. In addition, the ICAROS platform is capable of a glide ratio of approximately 2 and a glide velocity of approximately $5.3 ms^{-1}$ on par with climbing and gliding animals.

The developed multi-modal platform has shown the ability to operate in a similar manner to biological analogs using conventional mechanisms and structures in engineering and limited sensing and processing on-board. The platform has set a minimum benchmark for operational capabilities of future platforms. Even utilizing the existing design, it is possible to add increased processing and sensing capabilities which will enable a closer examination of transitioning conditions and mechanisms. The information gained on passive and active transitioning mechanisms integrated into this existing platform will make it possible to reduce minimum drop height in future designs.

Future iterations of the platform will focus on a number of changes for increasing the performance and robustness of the platform. Improved attachment mechanisms will allow the multi-modal platform to climb on a diverse number of surfaces and increase operational capability to exterior environments. Integration of on-board sensing and feedback control will allow for controlled flight and generation of autonomous behavior. Additional limbs will remove the need for passive roll control currently accomplished utilizing wing pontoons. Active launching mechanisms will replace the ballistic dive transition currently utilized by the platform. This will be combined with an analysis of the aerodynamics affecting the platform during transition to reduce the operating height currently necessary for initiating flight. A preliminary CAD model of the proposed future platform utilizing these improvements is shown in Figure 8.1 with an integrated bio-inspired, flexible patagium and controllable, flexible tail.

In the future, integration of smart materials will allow future iterations of the ICAROS platform to utilize actuators and structures for collapsible wings, patagiums, and active flight control. This application of smart materials and adaptive morphology will also reduce mass through more intelligent integration of individual modes of locomotion and is expected to increase the performance, robustness, and versatility of the platform.

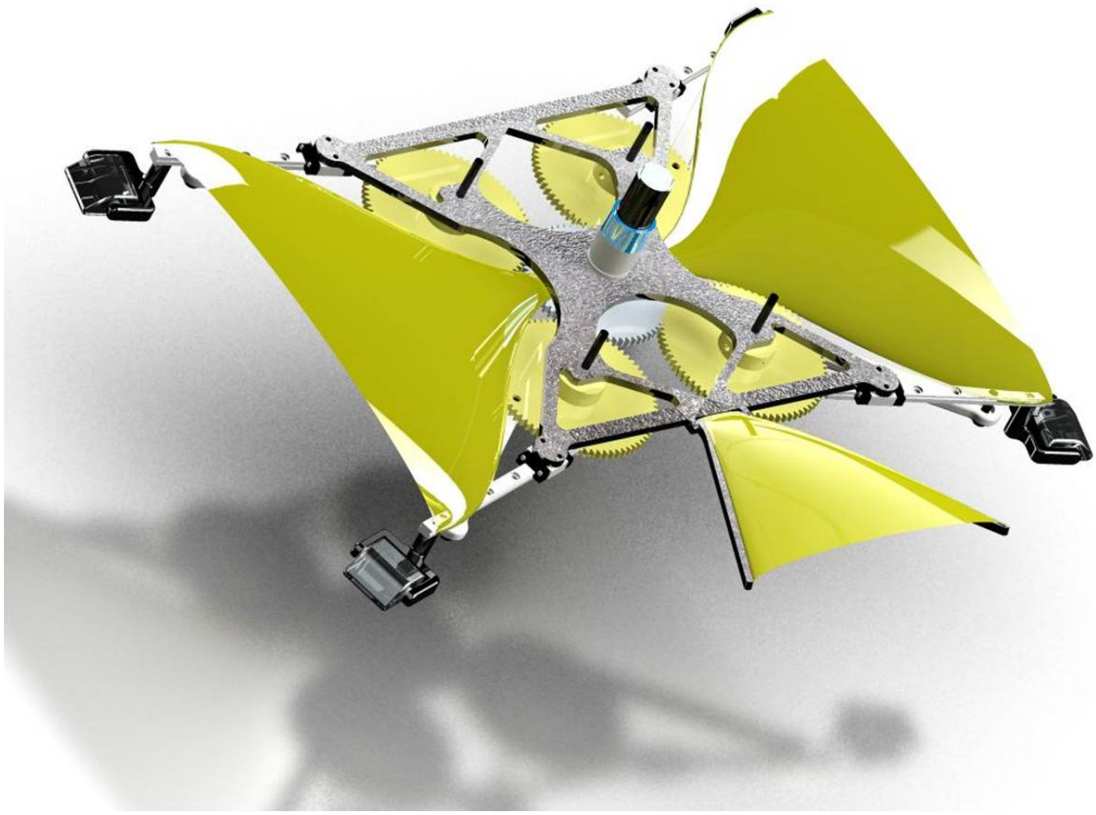


Figure 8.1: Preliminary CAD model of next generation of an Adaptive Robotic Multi-Modal System (ARM^{2}S) that utilizes a bio-inspired, flexible patagium for gliding flight similar to structures seen on flying squirrels and flying dragons. The platform will incorporate specialized, compliant feet for operation on numerous surfaces.

APPENDIX A

WORKING MODEL 2D DYNAMIC SIMULATION SCRIPT

This script creates the physical model in Working Model and sets the physical constraints of the platform (i.e. mass, frequency, torque limits). The simulation has the capability of changing mass and sprawl angle as well as motor transmission constants.

```
Sub Main()
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        ''''''                     SET VARIABLES
           ''''''
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

                WALL_ANGLE = 70*pi/180
                SPRAWL_ANGLE = 0*pi/180
                GRAVITY_CONST = 9.81*sin(WALL_ANGLE)
                Anim_Step = 0.001
                T_EQ=54 * 3.16     'Initial transient time
                FILE_NAME = "H:\BOB\Simulation\Sim 2-14-2012\70\
                   Test2.txt"
                'Test 1 = -10 Deg
                'Test 2 = 0 Deg
                'Test 3 = 10 Deg
                'Test 4 = 20 Deg
                'Test 5 = 30 Deg
                Freq = 4.2
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        ''''''                     SET OPTIONS
           ''''''
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

                BODY_ANCHOR = 0
```

```
        USE_MOTOR = 1  'Defult (0) is Frequency controller
        FOOTSTICK = 1
        METERS_ON = 1
        GRAPHS_ON = 1
        FOOTHOLDTYPE = "pin"
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,               PHYSICAL DEFINITIONS
   ,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,


        Body_Width = 0.0756
        Body_Height = 0.186
        Body_Mass = 0.185
        Body_Moment = 0.00000001006  'Best for frequency
           control is 0.00325
        Coupler_Length = 0.09
        Coupler_Width = 0.005
        Crank_Length = 0.025

        'Set Spring Constants
        SPRING_WRIST = 129
        DAMPER_WRIST = 9
        SPRING_SHOULDER = 100    'The damper for the
           shoulder needs to be 2 orders of magnitude
           less
        SHOULDER_DAMPER = 0.1

        dispTheta=pi/2-(pi/180)*SPRAWL_ANGLE   'Sprawl
           Angle
        ampTheta=(pi/180)*0
        phaseTheta=0
        LEGFREQ=-2*Freq*pi*3.75    'pos/pull    neg/push
        ampR =.025
    dispR=.1
        COEFF_FRICTION = 0.256
        FRICTION = 2*COEFF_FRICTION*((Body_Mass*9.81)*cos
           (WALL_ANGLE))

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,             SET DOCUMENT PROPERTIES
   ,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

```
        Dim Doc as WMDocument
        'Set Doc = WM.New()
        Set Doc = WM.ActiveDocument
        Doc.Reset
        Doc.Selectall
        Doc.Delete
        Set Doc.ShowRulers = True
        Set Doc.ShowGridLines = True
        Doc.UnitSystem = ”si radians”
        Doc.DecimalDigits = 5
        Doc.AutoAnimationStep = False
        Doc.AnimationStep = Anim_Step
        Doc.Gravity = ”linear”
        Doc.LinearGravityConst =GRAVITY_CONST
        Doc.WarnInconsistent = False
        Doc.AutoOverlapError = False
        Doc.OverlapError = 0.1

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,          DEFINE OBJECTS IN SIMULATION
   ,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

        Dim Body as WMBody
        Dim Anchor as WMConstraint
        Dim Pinion as WMBody
        Dim PinionHub as WMConstraint
        Dim Gear(2) as WMBody
        Dim GearHub(2) as WMConstraint
        Dim Shoulder(2) as WMBody
        Dim ShoulderPin(2) as WMConstraint
        Dim Rail(2) as WMBody
        Dim Origin as WMBody
        Dim GuideBlock(2) as WMConstraint
        Dim Coupler(2) as WMBody
        Dim GearCon(2) as WMConstraint
        Dim GearPin(2) as WMConstraint
        Dim CouplerPin(2) as WMConstraint
        Dim Motor as WMConstraint
        Dim Anchors as WMConstraint
        Dim Arm(2) as WMBody
        Dim Arm2(2) as WMBody
        Dim Wrist(2) as WMConstraint
        Dim WristSpring(2) as WMConstraint
```

```
        Dim Foot(2) as WMBody
        Dim AnklePin(2) as WMConstraint
        Dim WristPin(2) as WMConstraint
        Dim Foothold(2) as WMConstraint
        Dim Shoulder2(2) as WMConstraint
        Dim ShoulderDamper(2) as WMConstraint
        Dim Force2 as WMConstraint
        Dim Rope as WMConstraint
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

''''''                  BUILD SIMULATION
   ''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        Set Body = Doc.NewBody("rectangle")
                Body.py.value = 0
                Body.name = "Body"
                Body.height.value = Body_Height
                Body.width.value = Body_Width
                Body.mass.value = Body_Mass
                Body.moment.value = Body_Moment
                Body.PX.Value = 0
                Body.PY.Value = 0

        ANG=Body.pr.value

        Set Pinion = Doc.NewBody("circle")
                Pinion.name = "Pinion"
                Pinion.radius.value = 0.00796145
                Pinion.mass.value = 0.001  ' 1 g
                Pinion.pr.value = ANG
                Pinion.px.value = Body.px.value
                Pinion.py.value = Body.py.value
                Pinion.elasticity.value = 0.0
        'Make this constraint a motor between the body
           and pinion

                Body.py.value = 0
                Body.px.value = 0

        Set PinionHub = Doc.NewConstraint("pin")
                PinionHub.name = "PinionHub"
                Set PinionHub.Point(1).Body = Body
                PinionHub.Point(1).px.value = Body.px.
                    value
```

```
        PinionHub.Point(1).py.value = Body.py.
            value
        Set PinionHub.Point(2).Body = Pinion
        PinionHub.Point(2).px.value = 0
        PinionHub.Point(2).py.value = 0

Set Motor = Doc.NewConstraint("motor")
        Motor.name = "Motor"
        Motor.Motortype = "rotation"
        Set Motor.Point(1).Body = Body
        Set Motor.Point(2).Body = Pinion

Set Force2=Doc.NewConstraint("Force")
        Force2.name="Friction Force"
        Set Force2.Point(1).Body=Body
        Force2.AlwaysActive=TRUE
        Force2.fx.formula=0
        Force2.fy.value=-FRICTION
        Force2.RotatewithBody = false

for ss = 1 to 2

        if ss=1 then
                nn = 1
                mm = -1
        else
                nn = 2
                mm = 1
        end if

        Set Gear(nn) = Doc.NewBody("circle")
                Gear(nn).name = "Gear("+str$(nn)
                    +")"
                Gear(nn).radius.value =
                    Crank_Length
                Gear(nn).mass.value = 0.001 ' 1 g
                'Gear(nn).pr.value = ANG
                Gear(nn).px.value = Pinion.px.
                    value+mm*Pinion.radius.value+
                    mm*Crank_Length
                Gear(nn).py.value = Body.py.value

        Set GearHub(nn) = Doc.NewConstraint("pin
            ")
```

```
        GearHub(nn).name = "GearHub("+
            str$(nn)+")"
        Set GearHub(nn).Point(1).Body =
            Body
        Set GearHub(nn).Point(2).Body =
            Gear(nn)
        GearHub(nn).Point(1).px.value =
            Pinion.px.value+mm*Pinion.
            radius.value+mm*Crank_Length
        GearHub(nn).Point(1).py.value =
            Body.py.value
        'GearHub(nn).Point(2).px.value =
            Body.px.value
        'GearHub(nn).Point(2).py.value =
            Body.py.value

Set Shoulder(nn) = Doc.NewBody("rectangle
    ")
        Shoulder(nn).name = "Shoulder("+
            str$(nn)+")"
        Shoulder(nn).mass.value = 0.001
        Shoulder(nn).width.value = 0.01
        Shoulder(nn).height.value = 0.01
        Shoulder(nn).py.value = Body.py.
            value+0.069
        Shoulder(nn).px.value = Gear(nn).
            px.value+mm*0.012
        Shoulder(nn).pr.value = Body.pr.
            value-mm*SPRAWL_ANGLE

Set ShoulderPin(nn) = Doc.NewConstraint("
    RSpring")
        ShoulderPin(nn).name = "
            ShoulderPin("+str$(nn)+")"
        Set ShoulderPin(nn).Point(1).Body
            = Body
        Set ShoulderPin(nn).Point(2).Body
            = Shoulder(nn)
        ShoulderPin(nn).Point(1).px.value
            = Shoulder(nn).px.value
        ShoulderPin(nn).Point(1).py.value
            = Shoulder(nn).py.value
        ShoulderPin(nn).Point(2).px.value
            = 0
```

```
        ShoulderPin(nn).Point(2).py.value
            = 0
        ShoulderPin(nn).Rotation.value =
            Body.pr.value-mm*SPRAWL_ANGLE
        ShoulderPin(nn).K.Value =
            SPRING_SHOULDER

    Set ShoulderDamper(nn) = Doc.
        NewConstraint("RDamper")
        ShoulderDamper(nn).name = "
            ShoulderDamper("+str$(nn)+")"
        Set ShoulderDamper(nn).Point(1).
            Body = Body
        ShoulderDamper(nn).Point(1).px.
            value = Shoulder(nn).px.value
        ShoulderDamper(nn).Point(1).py.
            value = Shoulder(nn).py.value
        Set ShoulderDamper(nn).Point(2).
            Body = Shoulder(nn)
        ShoulderDamper(nn).Point(2).px.
            value = 0
        ShoulderDamper(nn).Point(2).py.
            value = 0
        ShoulderDamper(nn).Rotation.value
            = Body.pr.value-mm*
            SPRAWL_ANGLE
        ShoulderDamper(nn).K.value =
            SHOULDER_DAMPER

    Set Rail(nn) = Doc.NewBody("rectangle")
        Rail(nn).name = "Rail("+str$(nn)
            +")"
        Rail(nn).mass.value = 0.001
        Rail(nn).width.value = 0.005
        Rail(nn).height.value = 0.085
        Rail(nn).py.value = Shoulder(nn).
            py.value
        Rail(nn).px.value = Shoulder(nn).
            px.value
        Rail(nn).pr.value = Shoulder(nn).
            pr.value

    Set GuideBlock(nn) = Doc.NewConstraint("
        KeyedVslot")
```

```
            Set GuideBlock(nn).Point(1).Body
                = Shoulder(nn)
            Set GuideBlock(nn).Point(2).Body
                = Rail(nn)

Set Coupler(nn) = Doc.NewBody("rectangle
    ")
            Coupler(nn).name = "Coupler("+
                str$(nn)+")"
            Coupler(nn).width.value =
                Coupler_Width
            Coupler(nn).height.value =
                Coupler_Length
            Coupler(nn).mass.value = 0.001
            'Coupler(nn).pr.value = (1*mm
                *45-180)*3.14/180

Set GearCon(nn) = Doc.NewConstraint("gear
    ")
            Set GearCon(nn).Point(1).Body =
                Gear(nn)
            Set GearCon(nn).Point(2).Body =
                Pinion
            GearCon(nn).AutoComputeGearRatio
                = false
            GearCon(nn).GearRatio.value = 1/mm
                *3.75

Body.px.value = 0
Body.py.value = 0
Body.pr.value = 0

Set GearPin(nn) = Doc.NewConstraint("pin
    ")
            GearPin(nn).name = "GearPin("+
                str$(nn)+")"
            Set GearPin(nn).Point(1).Body =
                Gear(nn)
            Set GearPin(nn).Point(2).Body =
                Coupler(nn)
            GearPin(nn).Point(2).px.value =
                0'GearHub(nn).Point(2).px.
                value/mm*Crank_Length
            if nn = 1 then
```

```
                GearPin(nn).Point(2).py.
                    value = mm*Coupler(nn)
                    .height.value/2
        else
                GearPin(nn).Point(2).py.
                    value = -mm*Coupler(nn
                    ).height.value/2
        end if
        GearPin(nn).Point(1).py.value =
            Body.py.value
        GearPin(nn).Point(1).px.value =
            GearHub(nn).Point(2).px.value-
            Crank_Length          'Gear(nn).
            px.value

Body.px.value = 0
Body.py.value = 0
'Gear(nn).pr.value = -15*pi/180

Set CouplerPin(nn) = Doc.NewConstraint("
    pin")
        CouplerPin(nn).name = "CouplerPin
            ("+str$(nn)+")"
        Set CouplerPin(nn).Point(1).Body
            = Coupler(nn)
        Set CouplerPin(nn).Point(2).Body
            = Rail(nn)
        if nn = 1 then
                CouplerPin(nn).Point(1).
                    py.value = -mm*Coupler
                    (nn).height.value/2
        else
                CouplerPin(nn).Point(1).
                    py.value = mm*Coupler(
                    nn).height.value/2
        end if
        CouplerPin(nn).Point(2).py.value
            = Rail(nn).height.value/4

Body.py.value = 0
Body.px.value = 0
'Gear(nn).pr.value = 0

'' Begin making the wrist components ''
Set Arm(nn) = Doc.NewBody("Rectangle")
```

```
        Arm(nn).name="Arm("+str$(nn)+")"
        Arm(nn).pr.value = Body.pr.value+
            Rail(nn).pr.value
        Arm(nn).py.value=Rail(nn).py.
            value+Rail(nn).height.value/2
        Arm(nn).px.value=Rail(nn).px.
            value+(mm)*Rail(nn).width.
            value+0.0025*mm
        Arm(nn).width.value=Rail(nn).
            width.value
        Arm(nn).height.value=0.005
        Arm(nn).mass.value = 0.001

Set Arm2(nn) = Doc.NewBody("Rectangle")
        Arm2(nn).name = "Arm("+str$(nn)
            +")"
        Arm2(nn).pr.value=Body.pr.value+
            Rail(nn).pr.value
        Arm2(nn).py.value=Arm(nn).py.
            value+0.005
        Arm2(nn).px.value=Arm(nn).px.
            value+0.001*mm
        Arm2(nn).width.value=
            Coupler_Width
        Arm2(nn).height.value=0.01
        Arm2(nn).mass.value = 0.001


Set Wrist(nn) = Doc.NewConstraint("
    KeyedVslot")
        Wrist(nn).name = "Wrist("+str$(nn
            )+")"
        Set Wrist(nn).Point(1).Body = Arm
            (nn)
        Set Wrist(nn).Point(2).Body=Arm2(
            nn)

Set WristSpring(nn) = Doc.NewConstraint("
    springdamper")
        WristSpring(nn).name = "Wrist
            Spring("+str$(nn)+")"
        Set WristSpring(nn).Point(1).Body
            = Arm(nn)
        Set WristSpring(nn).Point(2).Body
            = Arm2(nn)
```

```
        WristSpring(nn).K.value =
            SPRING_WRIST
        WristSpring(nn).DamperK.value =
            DAMPER_WRIST
        WristSpring(nn).length.value=0.01

Set Foot(nn) = Doc.NewBody("circle")
        Foot(nn).name = "Foot("+str$(nn)
            +")"
        Foot(nn).radius.value = 0.0035
        Foot(nn).mass.value = 0.001
        Foot(nn).px.value = Arm2(nn).px.
            value+mm*0.001
        Foot(nn).py.value = Arm2(nn).py.
            value+Arm2(nn).height.value/2

Set WristPin(nn) = Doc.NewConstraint("
    squarepin")
        Set WristPin(nn).Point(1).Body =
            Rail(nn)
        Set WristPin(nn).Point(2).Body =
            Arm(nn)
        WristPin(nn).Point(1).px.value =
            0'Rail(nn).px.value+(mm)*Rail(
            nn).width.value+0.0025*mm 'Arm
            (nn).px.value
        WristPin(nn).Point(1).py.value =
            Rail(nn).height.value/2
        WristPin(nn).Point(2).px.value =
            0'Rail(nn).px.value+(mm)*Rail(
            nn).width.value+0.0025*mm      '
            mm*Arm(nn).px.value
        WristPin(nn).Point(2).py.value =
            0'-Rail(nn).height.value/2

Body.py.value = 0
Body.px.value = 0

Set AnklePin(nn) = Doc.NewConstraint("
    squarepin")
        Set AnklePin(nn).Point(1).Body =
            Arm2(nn)
        Set AnklePin(nn).Point(2).Body =
            Foot(nn)
```

```
                AnklePin(nn).Point(1).px.value =
                    0
                AnklePin(nn).Point(1).py.value =
                    Arm2(nn).height.value/2
                AnklePin(nn).Point(2).px.value =
                    0
                AnklePin(nn).Point(2).py.value =
                    0

            Body.py.value = 0
            Body.px.value = 0
            Body.pr.value = 0

            Set Foothold(nn) = Doc.NewConstraint(
                FOOTHOLDTYPE)
                Foothold(nn).name="Foothold("+
                    str$(nn)+")"
                Foothold(nn).AlwaysActive = False
                Foothold(nn).ActiveWhen.Formula=0
                Set Foothold(nn).Point(1).Body=
                    Foot(nn)
                Foothold(nn).point(1).px.formula
                    ="Foot(nn).px.value"
                Foothold(nn).point(1).py.formula
                    ="Foot(nn).py.value"
                Foothold(nn).point(2).px.formula
                    ="body["+str$(Foot(nn).ID)+"].
                    p.x"
                Foothold(nn).point(2).py.formula
                    ="body["+str$(Foot(nn).ID)+"].
                    p.y"

next ss

'Set Rope = Doc.NewConstraint(

V_rated = 6.0
V_act = 10.1
Vm = V_act/V_rated
W_des = LEGFREQ
G = 14
W_nl = Vm*10600 / G
T_c = Vm*(3.2/1000)*G
T_s = Vm*(11.2/1000)*G
```

```
        if USE_MOTOR = 1 then
                Motor.Motortype = "Velocity"
                Motor.Field.Formula = "if(abs(
                    constraintforce("+str$(Motor.ID)+").r)
                     < abs("+str$(T_s)+"),"+str$(W_des)
                    +",("+str$(T_s)+"-(constraintforce("+
                    str$(Motor.ID)+").r))*"+str$(W_nl)
                    +"/"+str$(T_s)+")"
                'Motor.Motortype = "Torque"
                'Motor.Field.Formula = "if(abs(
                    constraintforce("+str$(Motor.ID)+").r)
                     < abs("+str$(T_s)+"), -("+str$(T_s)
                    +"-("+str$(W_des)+"*"+str$(T_s)+")/"+
                    str$(W_nl)+"),-("+str$(T_c)+"))"
        else
                Motor.Field.Formula=str$(LEGFREQ)+" * t"
        end if

for ss=1 to 2
        if ss=1 then
                nn=1
                mm=-1
        else
                nn=2
                mm=1
        end if
        if FOOTSTICK=1 then
                if nn<2 then
                        CLIMBPULL1$="if((mod(body["+str$(
                            Gear(nn).ID)+"].p.r-body["+
                            str$(Body.ID)+"].p.r+(90*pi
                            /180), 2*pi)<pi+0.1),1,0)"
                else
                        CLIMBPULL1$="if((mod(-body["+str$
                            (Gear(nn).ID)+"].p.r-body["+
                            str$(Body.ID)+"].p.r+(90*pi
                            /180), 2*pi)>pi+0.1),1,0)"

                end if

                if nn=1 then
                        Foothold(nn).ActiveWhen.Formula =
                             CLIMBPULL1$
                end if
                if nn=2 then
```

```
                    Foothold(nn).ActiveWhen.Formula =
                        CLIMBPULL1$
                end if
        end if
next ss
if METERS_ON=1 then

                if GRAPHS_ON=1 then
                        GRAPH_FORMAT$="graph"
                else
                        GRAPH_FORMAT$="meter"
                end if

                Dim Meter0 as WMOutput
                Set Meter0=Doc.NewOutput()
                        Meter0.name="Time & Velocity"
                        Meter0.Format="meter"
                        Meter0.width=200
                        Meter0.x=0
                        Meter0.Y=0
                        Meter0.Column(1).Cell.Formula="
                            Time"
                        Meter0.Column(2).Cell.Formula="
                            body["+str$(body.ID)+"].p.y"
                        Meter0.Column(2).Label="Position
                            Y"
                        Meter0.Column(3).Cell.Formula="
                            body["+str$(body.ID)+"].p.x"
                        Meter0.Column(3).Label="Position
                            X"
                        'Meter0.Column(4).Cell.Formula="
                            body["+str$(Body.ID)+"].p.y/t"
                        'Meter0.column(4).Label="Average
                            Vy"

                Dim Meter1 as WMOutput
                Set Meter1=Doc.NewOutput()
                        Meter1.name="Velocity"
                        Meter1.Format="meter"
                        Meter1.width=200
                        Meter1.x=250
                        Meter1.Y=0
                        Meter1.column(1).Cell.Formula="
                            body["+str$(Body.ID)+"].v.y"
                        Meter1.column(1).Label="Inst. Vy"
```

```
                        Meter1.column(2).Cell.Formula=”
                           body[”+str$(Body.ID)+”].v.x”
                        Meter1.column(2).Label=”Inst. Vx”

                Dim Meter2 as WMOutput
                Set Meter2=Doc.NewOutput()
                        Meter2.name=”Torque”
                        Meter2.Format=”meter”
                        Meter2.width=200
                        Meter2.x=0
                        Meter2.Y=300
                        Meter2.column(1).Cell.Formula=”
                           constraintforce(”+str$(Motor.
                           ID)+”).r”
                        Meter2.column(1).Label=”Torque”
                        Meter2.column(2).Cell.Formula=”
                           constraint[”+str$(Motor.ID)
                           +”].dv.r”
                        Meter2.column(2).Label=”Motor
                           Speed”

end if



,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
' Set up collision rules                '
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
        if BODY_ANCHOR=1 then
                Set Anchors= Doc.NewConstraint(”
                   KeyedHslot”)
                Set Anchors.Point(2).Body=Body
                Anchors.ActiveWhen.Formula=”t<”+str$(T_EQ
                   )
        end if
        Doc.SelectAll False     'Make nothing collide so
           that robot can't collide with itself
        Doc.SelectAll
        Doc.Collide False
        Doc.SelectAll False
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

,,,,,,          OUTPUT SIM RESULTS TO FILE
   ,,,,,,
```

```
          ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

                    RUN_FRAMES=4000
                    'Doc.Run RUN_FRAMES
                    'Doc.Reset
                    Doc.ExportStartFrame = 0
                    Doc.ExportStopFrame = RUN_FRAMES
                    Doc.ExportMeterData FILE_NAME
                    Doc.Reset
                    ''next vv

End Sub
```

APPENDIX B

MATLAB ANALYSIS SCRIPT

This MATLAB script takes motion tracking data, current sensor data, and force data into a single program and analyzes each over the same period of time, taken from the motion tracking data set. Plots comparing different data sets are output.

Contents


```
function main()

    clear all; clc; close all;
```

Options

```
%Operation Options
Run_Motion = 0;                 % 1 to run the motion capture
    analysis
Run_Current = 0;                % 1 to run the current sensor
    analysis
```

```
Run_Force = 1;                % 1 to run the force sensor
    analysis
%Plot Options
opt.gridPlotForce = 0;        % 1 to plot each run's force
    individually
opt.gridPlotCurrent = 0;      % 1 to plot each run's
    current individually
opt.gridPlotVelocity = 0;     % 1 to plot each run's
    vertical and lateral velocity individually

opt.avePlotForce = 1;
opt.avePlotCurrent = 0;       % 1 to plot average current
    for each sprawl
opt.avePlotVelocity = 0;      % 1 to plot average
    velocities for each sprawl
```

Set folders and files

```
ForceData = {'..\90 Wall\0 Sprawl\Force\',...
             '..\90 Wall\10 Sprawl\Force\'};%,...
%               '..\50 Wall\0 Sprawl\Force\',...
%               '..\50 Wall\10 Sprawl\Force\',...
%               '..\50 Wall\20 Sprawl\Force\',...
%               '..\50 Wall\30 Sprawl\Force\'};
CurrentData = {'..\90 Wall\10 Sprawl\Current\'};
MotionData = {'..\90 Wall\10 Sprawl\Edited Video\'};

%NOTE: Runs 4 and 10 force data look bad, should re-run for
    90-0
```

Initialize Variables

```
params.fps = 300;
params.numPoints = 2;
params.numRuns = 2;
params.index = 0;
params.mass = 0.192;
params.voltage = 10.1;
params.g = 9.81;
params.strideStart = 2;
params.strieEnd = 3;
params.filter = 15;
params.distm = 90*10^-3;
params.ampFilter = 120;
params.numFolders = length(ForceData);
params.stepNum = 1;
```

Call Functions

```
    for i = 1:params.numFolders
        for j = 1:params.numRuns
            if Run_Motion == 1
                data(i,j).motion = getMotion(params,char(
                    MotionData(i)),i,j);
            end
            if Run_Current == 1
                data(i,j).current = getCurrent(params,char(
                    CurrentData(i)),data(i,j).motion.vertVelocity,
                    i,j);
            end
            if Run_Force == 1
                data(i,j).force = getForce(params,char(ForceData(
                    i)),i,j);
            end
            j
        end
    end
    for p = 1:params.numFolders
        if Run_Motion == 1
            stat(p).motion = motionStat(params,data,p);
        end
        if Run_Current == 1
            stat(p).current = currentStat(params,data,p);
        end
        if Run_Force == 1
            stat(p).force = forceStat(params,data,p);
        end
    end
    dataOptions(opt,params,data,stat);

end
```

Load the motion capture data and Translate

This function loads the motion capture data from video(X).mat and translates the pixels into normal units using the variable params.distm. The velocities are then calculated and all position and velocity data returned. Function returns: [structure for position, velocity, and averages]

```
function [exp] = getMotion(params,folder,i,j)
    run = load(strcat(folder,'video',num2str(j),'.mat'));
    result = run.result;
    index = params.index;
```

```
for i=1:length(result)-1
    if(result(i,4) == 2 && index == 0)
        index=i-1;
    end
    x(i-index,result(i,4)) = result(i,1);
    y(i-index,result(i,4)) = result(i,2);
    t(i-index,result(i,4)) = result(i,3)*(30/params.fps);
end
x1=x~=0;
y1=y~=0;
check = x1(:,1).*x1(:,2).*y1(:,1).*y1(:,2);
count = 0;
for i=1:length(check)
    if(check(i) == 0)
        x(i-count,:) = [];
        y(i-count,:) = [];
        t(i-count,:) = [];
        count=count+1;
    end
end
num2 = size(x,1);
for i =1:num2
    xDist(i) = x(1,1)-x(1,2);
    yDist(i) = y(1,1)-y(1,2);
end
distP = mean(sqrt(xDist.^2+yDist.^2));
factor = params.distm/distP;
x=x*factor;
y=y*factor;
for i=1:num2-1
    for j=1:params.numPoints
        xVel(i,j) = (x(i+1,j)-x(i,j))/(t(i+1,j)-t(i,j));
        yVel(i,j) = (y(i+1,j)-y(i,j))/(t(i+1,j)-t(i,j));
        aveX(i) = mean(x(i,:));
        aveY(i) = mean(y(i,:));
    end
end
exp.t = t;
exp.aveX = aveX;
exp.aveY = aveY;
for a = 1:size(xVel,1)
    exp.aveXVel(a) = (xVel(a,1)+xVel(a,2))/2;
    exp.aveYVel(a) = (yVel(a,1)+yVel(a,2))/2;
end
f = params.filter; a = 1; b = ones(1,f)*(1/f);
```

```
    locs = stride(params,exp);
    exp.aveXVel = filter(b,a,exp.aveXVel);
    exp.aveYVel = filter(b,a,exp.aveYVel);

    exp.aveXVel = exp.aveXVel(locs(2):locs(3));
    exp.aveYVel = exp.aveYVel(locs(2):locs(3));
    exp.vertVelocity = (exp.aveY(end)-exp.aveY(1))/exp.t(end);
    exp.latVelocity = max(abs(exp.aveXVel));
    exp.t = exp.t(locs(2):locs(3)) - exp.t(locs(2));
end
```

Load current data and calculate Power and Specific Resistance

This function loads the current sensor data log(X).txt and uses the matrix array.mat to cut off non-steady state running. The file array.mat is derived from the frames analyzed in the initial processing of the raw video motion capture data. Function returns: [struct for current, power, and specific resistance]

```
function [exp] = getCurrent(params,folder,vel,i,j)
    loc = zeros(10,2);
    array = load(strcat(folder,'array.mat'));
    loc = array.array;
    current = load(strcat(folder,'log',num2str(j),'.txt'));
    exp.amp = current(loc(j,1):loc(j,2))/512;
    exp.power = exp.amp*params.voltage;
    exp.SR = exp.power/(params.mass*params.g*vel);
end
```

Load the Force Data and crop to first step on plate

This function loads the force data using 'textread' and assigns the data to variables. The variables are offset by the baseline log and converted to SI units. A 3rd order Butterworth filter is applied creating a low pass filter for anything under 50 Hz. The F_y peak is identified and the data shortened to only cover one step. Function returns: [struct of Fx,Fy,Fz and force peak information]

```
function [exp] = getForce(params,folder,i,j)
    file_name_baseline =  strcat(folder,'logBase.log');
    [Fx_baseline,Fy_baseline,Fz_baseline,Mx_baseline,My_baseline,
       Mz_baseline] = textread(file_name_baseline,'%f %f %f %f %f
        %f','delimiter',',');
    file_name = strcat(folder,'log',num2str(j),'.log');
    [Fx,Fy,Fz,Mx,My,Mz] = textread(file_name,'%f %f %f %f %f %f
       ','delimiter',',');
```

```
    sample_rate = 10000; sample_interval = 1/sample_rate;
        num_samples = length(Fx);
    time = (0:num_samples-1)*sample_interval; time = time';
    Fx_offset = mean(Fx_baseline); Fy_offset = mean(Fy_baseline);
    Fz_offset = mean(Fz_baseline); Mx_offset = mean(Mx_baseline);
    My_offset = mean(My_baseline); Mz_offset = mean(Mz_baseline);
    Fx_calibrated = Fx - Fx_offset; Fy_calibrated = Fy -
        Fy_offset; % [lb]
    Fz_calibrated = Fz - Fz_offset; Mx_calibrated = Mx -
        Mx_offset;
    My_calibrated = My - My_offset; Mz_calibrated = Mz -
        Mz_offset;
    exp.orig.Fx_calibrated = Fx_calibrated * 1/2.20462262 * 9.81;
        % [N]
    exp.orig.Fy_calibrated = Fy_calibrated * 1/2.20462262 * 9.81;
        % [N]
    exp.orig.Fz_calibrated = Fz_calibrated * 1/2.20462262 * 9.81;
        % [N]
    Fx_calibrated = exp.orig.Fx_calibrated;
    Fy_calibrated = exp.orig.Fy_calibrated;
    Fz_calibrated = exp.orig.Fz_calibrated;
    %Filter data and assign to struct for return to main
    [z,p]=butter(3,0.02,'low');
    Fy_calibrated = filtfilt(z,p,Fy_calibrated);
    Fx_calibrated = filtfilt(z,p,Fx_calibrated);
    Fz_calibrated = filtfilt(z,p,Fz_calibrated);
  % data.time = time;
    [pks,locs] = locatePeaks(abs(Fx_calibrated),abs(Fy_calibrated
        ),abs(Fz_calibrated));
    exp.orig.Fy = -exp.orig.Fy_calibrated(locs.Fy(1,params.
        stepNum)-200:locs.Fy(1,params.stepNum)+800);
    exp.orig.Fx = -exp.orig.Fx_calibrated(locs.Fy(1,params.
        stepNum)-200:locs.Fy(1,params.stepNum)+800);
    exp.orig.Fz = exp.orig.Fz_calibrated(locs.Fy(1,params.stepNum
        )-200:locs.Fy(1,params.stepNum)+800);
%Cropped Data
    exp.Fy_calibrated = -Fy_calibrated(locs.Fy(1,params.stepNum)
        -200:locs.Fy(1,params.stepNum)+800);
    exp.Fx_calibrated = -Fx_calibrated(locs.Fy(1,params.stepNum)
        -200:locs.Fy(1,params.stepNum)+800);
    exp.Fz_calibrated = Fz_calibrated(locs.Fy(1,params.stepNum)
        -200:locs.Fy(1,params.stepNum)+800);
    num_samples = (locs.Fy(1,1)+800)-(locs.Fy(1,params.stepNum)
        -200);
    time = (0:num_samples)*sample_interval;
```

```
    exp.time = time';
    exp.peaks.Fx = pks.Fx; exp.peaks.Fy = pks.Fy; exp.peaks.Fz =
        pks.Fz;
end
```

Identify the strides in the motion capture data

This function filters the lateral velocity data from the exp information passed from the main function by a fth order filter and than identifies the peaks using the MATLAB function findpeaks. The locations of the peaks are returned to the main function through the variable locs. Function returns: [locations of the peaks for the motion capture x vel]

```
function [locs] = stride(params,exp)
    f = params.filter;      a = 1;      b = ones(1,f)*(1/f);
    exp.aveXVel = filter(b,a,exp.aveXVel);
    [pks,locs] = findpeaks(exp.aveXVel,'MINPEAKDISTANCE',45);
end
```

Find the peaks of all the ground reaction forces

This function is designed to find the peak value and location of the peak ground reaction force forall the directions analyzed and returns them to the function Filtration Function returns : [magnitude peaks,locations of peaks]

```
function [pks,locs] = locatePeaks(Fx,Fy,Fz)
    [pks.Fx,locs.Fx] = findpeaks(Fx,'minpeakheight',1);
    [pks.Fy,locs.Fy] = findpeaks(Fy,'minpeakheight',2);
    [pks.Fz,locs.Fz] = findpeaks(Fz,'minpeakheight',1);
end
```

Calculate Statistics for Force Data

This function is designed to create an array of numRuns rows and length of the vectors into a matrix. The function then calculates the average at each data point in the matrix and the standard deviation. Function returns: [struct ave for mean and std]

```
function [ave] = forceStat(params,data,p)
    for i = 1:params.numRuns
        for j = 1:length(data(p,1).force.Fx_calibrated)
            Fx(i,j) = data(p,i).force.Fx_calibrated(j);
            Fy(i,j) = data(p,i).force.Fy_calibrated(j);
            Fz(i,j) = data(p,i).force.Fz_calibrated(j);
        end
    end
    for k = 1:length(Fx)%params.numRuns%length(Fx)
```

```
        ave.Fx(k) = mean(Fx(:,k));
        ave.FxSTD(k) = std(Fx(:,k));
        ave.Fy(k) = mean(Fy(:,k));
        ave.FySTD(k) = std(Fy(:,k));
        ave.Fz(k) = mean(Fz(:,k));
        ave.FzSTD(k) = std(Fz(:,k));
        ave.time(k) = (k-1)/10000;
    end
%     ave.tFx = mean(ave.Fx);
%     ave.stdFx = std(ave.Fx);
%     ave.tFy = mean(ave.Fy);
%     ave.stdFy = std(ave.Fy);
%     ave.tFz = mean(ave.Fz);
%     ave.stdFz = std(ave.Fz);
end

function [ave] = motionStat(params,data,i)
    %i = 1;
    for j = 1:params.numRuns
        yVelTotal(j) = data(i,j).motion.vertVelocity;
        for k = 1:length(data(i,j).motion.aveYVel)
            yVel(j,k) = data(i,j).motion.aveYVel(k);
            xVel(j,k) = data(i,j).motion.aveXVel(k);
        end
    end
    ave.yVel = mean(yVelTotal);
    ave.yVelSTD = std(yVelTotal);
    %ave.xVel = max
    for p = 1:length(yVel)
        ave.aveYVel(p) = mean(yVel(:,p));
        ave.stdYVel(p) = std(yVel(:,p));
        ave.aveXVel(p) = mean(xVel(:,p));
        ave.stdXVel(p) = std(xVel(:,p));
        ave.time(p) = (p-1)/params.fps;
    end
end
```

Statistics operation for the current sensor analysis

This function Function returns: [struct for averages and std calculated]

```
function [ave] = currentStat(params,data,i)
    for j = 1:params.numRuns
        for k = 1:length(data(i,j).current.amp)
        amp(j,k) = data(i,j).current.amp(k);
```

```
            power(j,k) = data(i,j).current.power(k);
            SR(j,k) = data(i,j).current.SR(k);
            end
        end
        for l = 1:params.numRuns
            ave.amp(l) = mean(amp(l,:));
            ave.ampSTD(l) = std(amp(l,:));
            ave.power(l) = mean(power(l,:));
            ave.powerSTD(l) = std(power(l,:));
            ave.SR(l) = mean(SR(l,:));
            ave.SRSTD(l) = std(SR(l,:));
        end
        ave.tAmp = mean(ave.amp);
        ave.stdAmp = std(ave.amp);
        ave.tPower = mean(ave.power);
        ave.stdPower = std(ave.power);
        ave.tSR = mean(ave.SR);
        ave.stdSR = std(ave.SR);
    end
```

Interpret the initial options and process data as instructed

This function looks at the struct 'opt' initially defined at the head of the main function and processes the figures as needed Function returns: nothing

```
function dataOptions(opt,params,data,stat)
    if opt.gridPlotForce == 1
        gridPlotForce(params,data,stat);
    elseif opt.gridPlotCurrent == 1
        gridPlotCurrent(params,data,stat);
    elseif opt.gridPlotVelocity == 1
        gridPlotVelocity(params,data,stat);
    elseif opt.avePlotForce == 1
        avePlotForce(params,data,stat);
    elseif opt.avePlotCurrent == 1
        avePlotCurrent(params,data,stat);
    elseif opt.avePlotVelocity == 1
        avePlotVelocity(params,data,stat);
    end
end
```

Function to plot averages of data to Force Plots

```
function avePlotForce(params,data,stat);
```

```
    t = [0 stat(1).force.time(end)]; z = [0 0];
    color = ['r' 'g'];
    for i = 1:params.numFolders
        figure('Color',[1 1 1])
        plot(stat(i).force.time,stat(i).force.Fx,'-b','LineWidth
            ',3)
        hold on
        plot(stat(i).force.time,stat(i).force.Fy,'--g','LineWidth
            ',3)
        plot(stat(i).force.time,stat(i).force.Fz,'-.r','LineWidth
            ',3)
        plot(t,z,'-k','LineWidth',3)
        xlabel('Time, s','fontsize',24,'FontName','Calibri')
        ylabel('Force, N','fontsize',24,'FontName','Calibri')
        title('Averaged Ground Reaction Forces','fontsize',28,'
           FontName','Calibri')
        for k = 1:length(stat(i).force.time)
            if mod(k,20) == 0
                errorbar(stat(i).force.time(k),stat(i).force.Fx(k
                    ),stat(i).force.FxSTD(k),'b')
                hold on
                errorbar(stat(i).force.time(k),stat(i).force.Fy(k
                    ),stat(i).force.FySTD(k),'g')
                errorbar(stat(i).force.time(k),stat(i).force.Fz(k
                    ),stat(i).force.FzSTD(k),'r')
            end
        end
        legend('F_{x}','F_{y}','F_{z}');
    end
    tilefigs;
end
```

Function to plot all force data to individual figures

This function takes the struct 'data' and creates i+j number of figures for each data run on the force plate Function returns: [number of figures created]

```
function gridPlotForce(params,data,stat)
    for i = 1:params.numFolders
        for j = 1:params.numRuns
            figure
            t = [0 data(i,j).force.time(end)]; z = [0 0];
            plot(data(i,j).force.time,data(i,j).force.
                Fx_calibrated,'.-b','LineWidth',3)
            hold on
```

```
            plot(data(i,j).force.time,data(i,j).force.
                Fy_calibrated,'--g','LineWidth',3)
            plot(data(i,j).force.time,data(i,j).force.
                Fz_calibrated,'-r','LineWidth',3)
            plot(t,z,'-k','LineWidth',3)
            xlabel('Time, s'); ylabel('Force, N')
            title('Ground Reaction Forces')
        end
    end
    tilefigs;
end
```

Plot all Velocities with an overlay of the average velocity

```
function avePlotVelocity(params,data,stat)
    color = ['k' 'b' 'r' 'm' 'y' 'g' 'c' 'b' 'r' 'm'];
    figure('Color',[1 1 1])
    for i = 1:params.numFolders
        for j = 1:params.numRuns
            t = [0 data(i,j).motion.t(end)]; z = [0 0];

            subplot(2,1,1)
                plot(data(i,j).motion.t,data(i,j).motion.aveYVel,
                    color(j),'LineWidth',1)
                hold on
                xlabel('Time, s','fontsize',24,'FontName','
                    Calibri'); ylabel('Velocity, ms^{-1}','
                    fontsize',24,'FontName','Calibri')
                title('Vertical velocity over one stride','
                    fontsize',24,'FontName','Calibri')
            subplot(2,1,2)
                plot(data(i,j).motion.t,data(i,j).motion.aveXVel,
                    color(j),'LineWidth',1)
                hold on
                xlabel('Time, s','fontsize',24,'FontName','
                    Calibri'); ylabel('Velocity, ms^{-1}','
                    fontsize',24,'FontName','Calibri')
                title('Lateral velocity over one stride','
                    fontsize',24,'FontName','Calibri')
        end
        subplot(2,1,1)
            plot(stat(i).motion.time(1:length(data(i,j).motion.t)
                ),stat(i).motion.aveYVel(1:length(data(i,j).motion
                .t)),'-k','LineWidth',3)
            hold on
```

```
            plot(t,z,'-k','LineWidth',2)
        subplot(2,1,2)
            plot(stat(i).motion.time(1:length(data(i,j).motion.t)
                ),stat(i).motion.aveXVel(1:length(data(i,j).motion
                .t)),'-k','LineWidth',3)
            hold on
            plot(t,z,'-k','LineWidth',2)
            legend('Run 1','Run 2','Run 3','Run 4','Run 5','Run
                6','Run 7','Run 8','Run 9','Run 10');
        for k = 1:length(stat(i).motion.time)
            subplot(2,1,1)
            if mod(k,5) == 0
                errorbar(stat(i).motion.time(k),stat(i).motion.
                    aveYVel(k),stat(i).motion.stdYVel(k))
                hold on
            end
            subplot(2,1,2)
            if mod(k,5) == 0
                errorbar(stat(i).motion.time(k),stat(i).motion.
                    aveXVel(k),stat(i).motion.stdXVel(k))
                hold on
            end
        end
    end
    tilefigs;

end
```

APPENDIX C

PERMISSIONS FOR THE REPRODUCTION OF IMAGES

Title:	Dynamics of rapid vertical climbing in cockroaches reveals a template
Author:	Goldman, D.; Chen, T.; Dudek, D.; Full, R.;
Publication:	The Journal of Experimental Biology
Publisher:	The Company of Biologists Limited
Date:	2006
Title:	Dynamics of geckos running vertically
Author:	Autumn, K.; Hsieh, S.; Dudek, D.; Chen, J.; Chitaphan, C.
Publication:	The Journal of Experimental Biology
Publisher:	The Company of Biologists Limited
Date:	2006

The Journal of Experimental Biology
The Company of Biologists Limited
Cambridge, CB4 0DL, UK
jeb.biologists.org

Permission for the reproduction of Figure 3(C) and Figure 7 from [28] and Figure 3 from [7] is granted with no charge for reproduction in the Masters Thesis by James Dickson. Permission granted via on-line communication with Sue Chamberlain at the Journal of Experimental Biologists.

The acknowledgement should state "reproduced / adapted with permission" and give the source journal name - the acknowledgement should either provide full citation details or refer to the relevant citation in the article list - the full citation details should include authors, journal, year, volume, and page citation.

Where appearing online, or in other electronic media, a link should be provided to the original artical (e.g. via DOI).

Title:	Gait regulation and feedback on a robotic climbing hexapod.
Author:	Haynes, G.C.; Rizzi, A.A.;
Publication:	Proceedings of Robotics: Science and Systems II
Publisher:	The MIT Press
Date:	2006

Robotics: Science and Systems II
The MIT Press
Cambridge, MA 02142 USA
mitpress.mit.edu

The non-exclusive permission for the reproduction of figure from [30] in Robotics: Science and Systems II to appear in the Masters Thesis by James Dickson is granted. Permission granted via on-line communication with Pamela Quick, Permissions Manager at The MIT Press.

Please credit the reprinted figure to the chapter authors, Robotics: Science and Systems II, EDITED BY Gaurav S. Sukhatme, Stefan Schaal, Wolfram Burgard, and Dieter Fox, published The MIT Press.

Title:	Design, implementation, and evaluation of an under-actuated miniature biped climbing robot
Author:	Minor, M.; Dulimarta, H.; Danghi, G.; Mukherjee, R.; Lal Tummala, R.; Aslam, D.;
Publication:	Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on
Publisher:	IEEE
Date:	2000

International Conference on Intelligent Robots and Systems
IEEE Xplore Digital Library
ieeexplore.ieee.org

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, this may be used as a permission grant. Permission granted for the reproduction of figure from [40]. Permission granted via RightsLink online system for Copyright Clearance Center from the IEEE Xplore site.

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:
1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line ©2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line ©[Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Title:	Unsupervised Learning of Terrain Apperance for Automated Coral Reef Exploration
Author:	Giguere, P.; Dudek, G.; Prahacs, C.; Plamondon, N.; Turgeon, K.
Publication:	Computer and Robot Vision, 200. CRV '09. Canadian Conference on
Publisher:	IEEE
Date:	25-27 May 2009

Canadian Conference on Computer and Robot Vision
IEEE Xplore Digital Library
ieeexplore.ieee.org

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, this may be used as a permission grant. Permission granted for the reproduction of figure from [40]. Permission granted via RightsLink online system for Copyright Clearance Center from the IEEE Xplore site.

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:
1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line ©2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line ©[Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Title:	Performance of different foot designs for a water running robot
Author:	Floyd, S.; Adilak, S.; Ramirez,S.;
Publication:	Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on
Publisher:	IEEE
Date:	19-23 May 2008

IEEE International Conference on Robotics and Automation (ICRA 2008)
IEEE Xplore Digital Library
ieeexplore.ieee.org

Title:	A sensor platform capable of aerial and terrestrial locomotion
Author:	Boria, F.J.; Bachmann, R.J.; Ifju, P.G.
Publication:	Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ Intenational Conference on
Publisher:	IEEE
Date:	2-6 August 2005

IEEE International Conference on Robotics and Automation (ICRA 2008)
IEEE Xplore Digital Library
ieeexplore.ieee.org

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, this may be used as a permission grant. Permission granted for the reproduction of figure from [40]. Permission granted via RightsLink online system for Copyright Clearance Center from the IEEE Xplore site.

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:
1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line ©2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line ©[Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

License Number:	2877830462453
License Date:	Mar 28,2012
Title:	Landing and Perching on Vertical Surfaces with Microspines for Small Unmanned Air Vehicles
Licensed Author:	Alexis Lussier Desbiens
Publication:	Journal of Intelligent and Robotic Systems
Publisher:	Springer
Licensed Content Date:	Jan 1, 2009
Type of Use:	Thesis / Dissertation
Portion:	Figures
Title of your thesis / dissertation:	Development of a Dynamic Bipedal Climbing and Multi-Modal Robotic Platform
Expected Completion Date:	Apr 2012

reproduction of the licensed material: "Springer and the original publisher /journal title, volume, year of publication, page, chapter/article title, name(s) of author(s), figure number(s), original copyright notice) is given to the publication in which the material was originally published, by adding; with kind permission from Springer Science and Business Media"

License Number:	2881481161685
License Date:	April 3,2012
Title:	Towards a Self-Deploying and Gliding Robot
Licensed Author:	Mirko Kovac
Publication:	Flying Insects and Robots
Publisher:	Springer
Licensed Content Date:	Oct 23, 2009
Type of Use:	Thesis / Dissertation
Portion:	Figures
Title of your thesis / dissertation:	Development of a Dynamic Bipedal Climbing and Multi-Modal Robotic Platform
Expected Completion Date:	Apr 2012

volume, year of publication, page, chapter/article title, name(s) of author(s), figure number(s), original copyright notice) is given to the publication in which the material was originally published, by adding; with kind permission from Springer Science and Business Media"

BIBLIOGRAPHY

[1] *Aerodynamics for Engineers*. Prentice Hall, 1998. 7.2

[2] *Principles of Animal Locomotion*. Princeton University Press, 2003. 3.1

[3] *Flying Insects and Robots*, chapter Towards a Self-Deploying and Gliding Robot. Springer, 2009. (document), 2.5

[4] Y. Asari, H. Yanagawa, and T. Oshida. Glaucomys, petaurista, flying squirrels, gliding, winglet, wrist anatomy. *Mammal Study*, 2007. 2.1.2

[5] A. Asbeck, S. Kim, M. Cutkosky, W. Provancher, and M. Lanzetta. Scaling hard vertical surfaces with compliant microspine arrays. *The International Journal of Robotics Research*, 2006. 2.2, 2.2, 2.3, 3.1, 4

[6] K. Autumn, M. Buehler, M. Cutkosky, R. Fearing, R. Full, D. Goldman, R. Groff, W. Provancher, A. Rizzi, U. Saranli, A. Saunders, and D. Koditschek. Robotics in scansorial environments. In *In Proceedins of SPIE: Unmanned Ground Vehicle Technology VII*. 2.1.1, 2.2, 2.3

[7] K. Autumn, S. Hsieh, D. Dudek, J. Chen, and C. Chitaphan. Dynamics of geckos running vertically. *The Journal of Experimental Biology*, 2006. (document), 2.1.1, 2.1.1, 2.1, 2.2, 2.2, C

[8] K. Autumn and A. Peattie. Mechanisms of adhesion in geckos. *Integrative & Comparative Biology*, 2002. 2.1.1

[9] S. Baek, K. Ma, and R. Fearing. Efficient resonant drive of flapping-wing robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. (document), 3.2, 3.3

[10] F. Boria, R. Bachmann, P. Ifju, R. Quinn, R. Vaidyanathan, C. Perry, and J. Wagener. A sensor platform capable of aerial and terrestrial locomotion. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. (document), 2.3, 2.3, 2.6

[11] T. Bretl, S. Rock, J.C. Latombe, B. Kennedy, and H. Aghazarian. Free-climbing with a multi-use robot. In *In Proceedings of the International Symposium on Experimental Robots*. 2.2

[12] G. Byrnes, N. Lim, and A. Spence. Take-off and landing kinetics of a free-ranging gliding mammal, the malayan colugo (galeopterus variegatus). In *Proceedings of the Royal Society B*. 2.1.2

[13] 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*, 1977. 2.1.1

[14] J. Clark, D. Goldman, T. Chen, R. Full, and D. Koditschek. Toward vertical dynamic climbing. In *Proceedings of the 9th International Conference on Climbing and Walking Robots*. 3.1, 3.1

[15] J. Clark, D. Goldman, P. Lin, G. Lynch, T. Chen, H. Komsuoglu, R. Full, and D. Koditschek. Design of a bio-inspired dynamical vertical climbing robot. In *Proceedings of Robotics: Science and Systems*. (document), 2.3, 2.2, 3.3.2, 3.5, 3.3.2, 4, 4.1, 6.3, 6.5, 7.1

[16] Z. Dai, S. Gorb, and U. Schwarz. Roughness-dependent friction force of the tarsal claw system in the beetle pachnoda marginata (coleoptera, scarabaeidae). *The Journal of Experimental Biology*, 2002. 2.2

[17] K. Daltorio, S. Gorb, A. Peressadko, A. Horchler, R. Ritzmann, and R. Quinn. A robot that climbs walls using micro-structured polymer feet. In *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*. 2.2

[18] A. Degani, A. Shapiro, H. Choset, and M.T. Mason. A dynamic single actuator vertical climbing robot. In *In Proceedings of the International Conference on Intelligent Robots and Systems*. 2.2

[19] A. Desbiens, A. Asbeck, and M. Cutkosky. Hybrid aerial and scansorial robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2.3, 2.3

[20] A. Desbiens, A. Asbeck, and M. Cutkosky. Landing, perching and taking off from vertical surfaces. *International Journal of Robotics Research*, 2011. (document), 2.7

[21] A. Desbiens and M. Cutkosky. Landing and perching on vertical surfaces with microspines for small unmanned air vehicles. *Journal of Intelligent and Robotic Systems*, 2009. 2.2, 2.3

[22] J. Dickson, C. Kulinka, M. Martin, J. Yeol, and J. Clark. Development and integration of adaptive robotic multi-modal system (arm^2s). In *Proceedings of AIAA 49th Annual Aerospace Sciences Meeting*. 6.1, 6.1

[23] S. Floyd, S. Adilak, and S. Ramirez. Performance of different foot designs for a water running robot. In *In Proceedings of the International Conference on Robotics and Automation*. (document), 2.5

[24] R. Full and D. Koditschek. Templates and anchors: Neuromechanical hypotheses of legged locomotion on land. *The Journal of Experimental Biology*, 1999. 2.1.1

[25] R. Galvao, E. Israeli, A. Song, X. Tian, K. Bishop, S. Swartz, and K. Breuer. The aerodynamics of compliant membrane wings modeled on mammalian flight mechanics. *American Institute of Aeronautics and Astronautics Fluid Dynamics*, 2006. 2.1.2

[26] C. Georgiades, A. German, A. Hogue, H. Liu, C. Prahacs, A. Ripsman, R. Sim, L. Torres, P. Zhang, M. Buehler, G. Dudek, M. Jenkin, and E. Milios. Aqua: an aquatic walking robot. In *In Proceedings of the International Conference on Intelligent Robots and Systems*. 2.3

[27] P. Giguere, G. Dudek, C. Prahacs, N. Plamondon, and K. Turgeon. Unsupervised learning of terrain appearances for automated coral reef exploration. In *In Proceedings of the Sixth Canadian Conference on Computer and Robot Vision*. (document), 2.5

[28] D. Goldman, T. Chen, D. Dudek, and R. Full. Dynamics of rapid vertical climbing in cockroaches reveals a template. *The Journal of Experimental Biology*, 2006. (document), 2.1.1, 2.1.1, 2.1, 2.1.1, 2.2, 2.2, 2.3, 3.1, 3.1, 4.1, C

[29] G. Haynes, A. Khripin, G. Lynch, J. Amory, A. Saunders, A. Rizzi, and D. Koditschek. Rapid pole climbing with a quadrapedal robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2.2, 2.2, 2.3

[30] G.C. Haynes and A. A. Rizzi. Gait regulation and feedback on a robotic climbing hexapod. In *In Proceedings of Robotics: Science and Systems*. (document), 2.4, 2.5, C

[31] S. Kim, A. Asbeck, W. Provancher, and M.R. Cutkosky. Spinybotii: Climbing hard walls with compliant microspines. In *In Proceedings of the Intenartional Conference on Advanced Robotics*. 2.2, 2.2

[32] V. Koli, C. Bhatnagar, and D. Mali. Gliding behaviour of indian giant flying squirrel petaurista philippensis. *Current Science*, 2011. 2.1

[33] M. Kovac, J. Zufferey, and D. Floreano. Towards the self deploying microglider, a biomimetic jumping and gliding robot. In *In Proceedings of the 4th International Symposium on Adaptive Motion of Animals and Machines*. 2.3

[34] G. Lynch. *Dynamic Vertical Climbing: Bioinspiraton, Design, and Analysis*. PhD thesis, University of Pennsylvania, 2011. (document), 3.6

[35] G. Lynch, J. Clark, and D. Koditschek. A self-exciting controller for high-speed vertical running. In *Proceedings of the International Conference on Intelligent Robots and Systems*. 2.2, 2.2, 4, 5, 8

[36] G. Lynch, L. Rome, and D. Koditschek. Sprawl angle in simplified models of vertical climbing: Implications for robots and roaches. *Applied Bionics and Biomechanics*, 2011. 3.3.1, 3.3.2

[37] J. McGuire. Allometric prediction of locomotor performance: An example from southeast asian flying lizards. *The American Naturalist*, 2003. 2.1.2

[38] J. McGuire and R. Dudley. The biology of gliding in flying lizards (genus draco) and their fossil and extant analogs. In *The Biomechanics and Behavior of Gliding Flight*. 2.1.2

[39] J. McGuire and R. Dudley. The cost of living large: Comparative gliding performance in flying lizards (agamidae: Draco). *The American Naturalist*, 2005. 2.1.2, 2.1

[40] M. Minor, H. Dulimarta, G. Danghi, G. Mukherjee, R. Lal Tummala, and D. Aslam. Design, implementation, and evaluation of an under-actuated miniature biped climbing robot. In *In Proceedings of IEEE/RSH International Conference on Intelligent Robots and Systems*. (document), 2.2, 2.4, C, C, C, C

[41] U. Norberg. Evolution of vertebrate flight: An aerodynamic model for the transition from gliding to active flight. *The American Naturalist*, 1985. 1

[42] H.S. Park and M. Sitti. Compliant footpad design analysis for a bio-inspired quadruped amphibious robot. In *In Proceedings of the International Conference on Intelligent Robots and Systems*. 2.3

[43] H. Prahlad, R. Pelrine, S. Stanford, J. Marlow, and R. Kornbluh. Electroadhesive robots-wall climbing robots enabled by a novel, robust and electrically controllable adhesion technology. *IEEE International Conference on Robotics and Automation*, pages 3028–3033, May 2008. 2.2

[44] W. Provancher, S. Jensen-Segal, and M. Fehlberg. Rocr: An energy-efficient dynamic wall-climbing robot. *IEEE/ASME Transactions on Mechatronics*, 2010. 2.2

[45] A. Russell and L. Dijkstra. Patagial morpology of draco volans (reptilia: Agamidae) and the origin of glissant locomotion in flying dragons. *Journal of Zoology*, 2001. 2.1.2

[46] D. Santos, B. Heyneman, S. Kim, N. Esparza, and M. Cutkosky. Gecko-inspired climbing behaviors on vertical and overhanding surfaces. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2.2

[47] J. Scheibe, W. Smith, J. Bassham, and D. Magness. Locomotor performance and cost of transport in the northern flying squirrel glaucomys sabrinus. *Acta Theriologica*, 2006. 2.1

[48] J. Socha, K. Miklasz, F. Jafari, and P. Vlachos. Non-equilibrium trajectory dynamics and the kinematics of gliding in a flying snake. *Bioinspiration and Biomimetics*, 2010. 2.1.2

[49] J. Socha, T. O'Dempsey, and M. LaBarbera. A 3-d kinematic analysis of gliding in a flying snake, chrysopelea paradisi. *The Journal of Experimental Biology*, 2005. 2.1

[50] M. Spenko, M. Cutkosky, C. Majidi, R. Fearing, R. Groff, and K. Autumn. Foot design and integration for bioinspired climbing robots. In *In Proceedings of SPIE: Unmanned Systems Technology VIII*. 2.2

[51] M. Spenko, G. Haynes, J. Saunders, M. Cutkosky, A. Rizzi, J. Full, and D. Koditschek. Biologically inspired climbing with a hexapedal robot. *Journal of Field Robotics*, 2008. 2.2, 2.2, 2.2, 2.2, 2.3, 2.3

[52] R. Thorington, K. Darrow, and C. G. Anderson. Wing tip anatomy and aerodynamics in flying squirrels. *Journal of Mammalogy*, 1998. 2.1.2

BIOGRAPHICAL SKETCH

Figure C.1: James Dickson, M.S. '12, Department of Mechanical Engineering, Florida State University holding the Okaloosa Sword presented to the Department of Mechanical Engineering for winning the US Air Force Campus Challenge IV. James' Masters Thesis work focused on the development of a prototype of the Adaptive Robotic Multi-Modal Systems presented as part of the winning proposal submitted to the United States Air Force Research Lab (AFRL).

James Dickson was born on December 12th, 1986 in St. Petersburg, Florida. He received a Bachelor of Science in Mechanical Engineering & Anthropology, Summa Cum Laude, from Florida State University in Spring of 2010 and a Master of Science in Mechanical Engineering from Florida State University in Spring of 2012. His undergraduate academic focus in Anthropology was Underwater Archaeology in which he participated in a Florida State University field school in Apalachee Bay. James went on to assist in publishing a field report on the study, Apalachee Bay Maritime Archaeological Research Project Report of Operations May 15th - June 9th, in Fall of 2006. In Mechanical Engineering, James focused

on biological inspired robotics. During his undergraduate academic career, James served as president of the Engineering Honor Society, Tau Beta Pi Florida Eta chapter, and reinstated and served as the president of the International Mechanical Engineering Society, Pi Tau Sigma Alpha Iota Chapter. James also served as student advisor and inaugural student president of the liberal arts honor society Phi Beta Kappa, Alpha of Florida Chapter. In June of 2010, James was featured as a Student Star on the Florida State University campus portal for his undergraduate research in biomimetic robotics and senior design project on Lunar Regolith excavation utilizing a telerobotic excavating platform.

During his graduate studies, James remained active as a graduate student advisor to Tau Beta Pi, Pi Tau Sigma, and Phi Beta Kappa assisting in the operations and planning of the local chapters. James' Masters Thesis research, presented in this thesis, focused on the development of a first generation Adaptive Robotic Multi-Modal System as part of the United States Air Force Campus Challenge Problem Solving Competition in miniature self-deploying systems in cluttered/confined environments. In April, 2012 the Florida State University Department of Mechanical Engineering was awarded the Okaloosa Sword for winning the Campus Challenge Competition.