

LOCOMOTION CONTROL EXPERIMENTS IN COCKROACH ROBOT
WITH ARTIFICIAL MUSCLES

by
Jongung Choi

Submitted in partial fulfillment of the requirements
For the degree of Doctor of Philosophy

Thesis Advisor: Dr. Roger D. Quinn

Department of Mechanical and Aerospace Engineering
CASE WESTERN RESERVE UNIVERSITY

August, 2005

CASE WESTERN RESERVE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

We hereby approve the dissertation of

Jongung Choi

candidate for the Doctor of Philosophy degree *.

Committee Chair: _____

Dr. Roger D. Quinn
Dissertation Advisor
Professor,
Department of Mechanical and Aerospace Engineering

Committee: _____

Dr. Joseph M. Mansour
Professor,
Department of Mechanical and Aerospace Engineering

Committee: _____

Dr. Roy E. Ritzmann
Professor,
Department of Biology

Committee: _____

Dr. Robert F. Kirsch
Associate Professor,
Department of Biomedical Engineering

August, 2005

*We also certify that written approval has been obtained for any
proprietary material contained therein.

Waiver of Reproduction Rights

To HyeYoung Jeon

Have I not commanded you? Be strong and
courageous. Do not be terrified: do not be
discouraged,
for the LORD your God will be with you
wherever you go.
Joshua 1:9

Table of Contents

Table of Contents	vi
List of Tables	ix
List of Figures	x
Acknowledgements	xvii
Abstract.....	xix
Chapter 1	1
Introduction	1
1.1 Cockroach-Inspired Robots	2
1.2 Dissertation Outline	4
References.....	7
Chapter 2	8
Background	8
2.1 Insect-inspired robots.....	8
References.....	14
Chapter 3	16
Leg Control	16
3.1 Workspace.....	16
3.1.1 Workspace of Front Leg	19
3.1.2 Workspace of Middle Leg	23
3.1.3 Workspace of Rear Leg	26
3.2 Inverse Kinematics.....	29
3.2.1 Modified Moore-Penrose Method.....	29
3.2.2 Inverse Kinematics Solution	32
3.3 Neural Network.....	36
3.4 Training of Neural Network.....	39
3.5 Validating Neural Networks	42
3.5.1 Design of the Desired Foot Path	43
3.5.2 Comparison of Desired and Actual Paths using Neural Networks ...	45
3.5.3 Discussion	48
References.....	49
Chapter 4	50
Coordination of Legs	50
4.1 Pre-Planned Tripod Gait	50
4.2 Cruse Control for Gait Generation.....	52
References.....	57
Chapter 5	59
Dynamic Simulation of Robot V	59
5.1 Mechanics of Robot V	60
5.2 Braided Pneumatic Actuator	63
5.2.1 McKibben vs. Festo Actuator	63
5.2.2 Testing of Festo Actuator Properties	65
5.2.3 Modeling of Festo Actuators	69
5.3 Valve System of Robot V	72

5.3.1 Inlet and Outlet Valves	73
5.3.2 Modeling of Valve	75
5.4 Applied Joint Torque	76
5.5 Dynamic Simulation of Robot V	78
5.5.1 Flow Chart of Simulation.....	79
5.5.2 File System of Simulation.....	82
References.....	86
Chapter 6	88
Metrics	88
6.1 Joint Angle Position Error	88
6.2 Foot Position	91
6.3 Velocity.....	93
6.4 Body motions during walking.....	93
References.....	93
Chapter 7	94
Simulation Results	94
7.1 Air and Ground Walking with Pre-Planned Gait.....	94
7.1.1 Joint Angles for Air and Ground Walking with Pre-Planned Gait ...	94
7.1.2 Foot Paths and Body Bounce using Pre-Planned Gait.....	98
7.2 Air and Ground Walking with Cruse Gait Control.....	101
7.2.1 Joint Angles for Air and Ground Walking using Cruse Controller	101
7.2.2 Foot Path and Body Bounce using Cruse Controller	105
7.3 Conclusions.....	107
References.....	108
Chapter 8	110
Implementation into Robot V Hardware.....	110
8.1 Robot V Sensors	110
8.1.1 Joint Angle Sensor	111
8.1.2 Pressure sensor.....	112
8.2 Robot V Hardware Setup.....	112
8.3 Initial Implementation of Leg Control.....	114
8.3.1 Initial implementation of Leg Control	114
8.4 Initial Implementation of Gait Control	115
8.4.1 Air-walking and Supported walking	116
8.5 Robot V Problems and Solutions.....	119
8.5.1 Parallel Joints in Front Legs	119
8.5.2 Non-symmetric Joint ROM and Sensor Tuning	123
8.5.3 Actuator Buckling Problem	127
8.6 Conclusions.....	130
Reference	131
Chapter 9	133
Comparisons with Cockroach and Robot III	133
9.1 High Speed Video Analysis of Cockroach	133
9.2 Cockroach vs. Design of Robot III	135
9.3 Cockroach vs. Design of Robot V	136
9.4 Comparison with Robot III	137

9.4.1	Beta and Alpha Axes Differ in Robot III and Robot V	138
9.4.2	Comparison of Forward Swing	140
9.4.3	Comparison of ROM of Coxa Segments	141
9.5	Comparison with Digitized Cockroach	143
9.5.1	Joint Synchronization between Digitized Cockroach and Robot V	144
9.5.2	Comparison of Joint Angles	146
9.5.3	Consequence of Outstretched Coxa Segment	155
9.6	Discussion: Can We Apply Biomechanics to Robot Directly?	157
	References	158
Chapter 10	159
Lessons Learned and Conclusions	159
10.1	Careful Choice of Sensors Saves Time	159
10.2	Deep Insight into Animal Design is Essential in Bio-Inspired Design	160
10.2.1	Local View of Joint Angle	160
10.2.2	Global View of Body Posture	162
10.3	Mechanical Design Differences	164
10.3.1	Inclined Coxa Segment	164
10.3.2	CF Joint	164
10.3.3	Beta Joint at Front Leg	165
10.4	Are the Actuators Strong Enough for Standing and Walking?	166
10.5	Improvement of Gait Controller	167
10.6	Foot Slipping	167
10.7	Conclusions	168
	References	169
Appendix	170
A.	Revised Cruse Controller	170
Bibliography	184

List of Tables

Table 1 : Segment lengths of right front leg in inches.....	21
Table 2 : Range of motion of right front leg in degrees.....	21
Table 3 : Segment lengths of right middle leg in inches.....	24
Table 4 : Joint ranges of motion of right middle leg in degrees.	24
Table 5 : Segment lengths of right rear leg in inches.	28
Table 6 : Joint ranges of motion of right rear leg in degrees.	28
Table 7 : Scaling Factors for Cruse Controller	54
Table 8 : Mass and Inertia Values of Robot V.....	62
Table 9 : File system of simulation.....	85
Table 10 : Minimum and maximum range of motion of all joints. *less range of motion than desired.....	124
Table 11 : Joint calibration coefficient after tuning.....	126
Table 12 : Range of Motion after Sensor Tuning	127
Table 13 : Revised ROM in degrees in Robot V angle definition convention	162

List of Figures

Figure 1 : Robot V. Robot V has 24 joints. There are 5, 4, and 3 joints in the front, middle and rear legs, respectively. The design of Robot is inspired by the cockroach called <i>Blaberus discoidalis</i> shown on the right.	7
Figure 2 : Protobot [5]. Protobot is modeled after <i>Periplaneta americana</i> cockroach. It has 18 DOF. It uses the double acting cylinder actuator.....	9
Figure 3 : TUM walking robot [6]. Each leg has 3 joints which are driven by 3 DC motors.....	10
Figure 4 : AirInsect.....	11
Figure 5 : Robot I and II. (a) Robot I and (b) Robot II.....	12
Figure 6 : Robot III.....	13
Figure 7 : Robot V.....	14
Figure 8 : Inertial coordinate system and direction of axis of rotations.	19
Figure 9 : Fully assembled right front leg.....	21
Figure 10 : X vs. Y plot of workspace of right front leg in inches.	22
Figure 11 : X vs. Z plot of workspace of right front leg in inches.....	23
Figure 12 : Y vs. Z plot of workspace of right front leg in inches.....	23
Figure 13 : Right middle leg.....	24
Figure 14 : X vs. Y plot of workspace of right middle leg in inches.....	25
Figure 15 : X vs. Z plot of workspace of right middle leg in inches.	25
Figure 16 : Y vs. Z plot of workspace of right middle leg in inches.	26
Figure 17 : Right rear leg.....	27
Figure 18 : X vs. Y plot of workspace of right rear leg in inches.....	28
Figure 19 : X vs. Z plot of workspace of right rear leg in inches.	29
Figure 20 : Y vs. Z plot of workspace of right rear leg in inches.	29
Figure 21 : Cubical points in the range of motion of right front leg.....	34

Figure 22 : Reachable points of right front leg	34
Figure 23 : Reachable points of right middle leg.....	35
Figure 24 : Reachable points of right rear leg.....	35
Figure 25 : Neural network for finding joint angles given foot position. The input is the desired foot position and output is the set of joint angles which correspond to desired foot position.	38
Figure 26 : Procedure to validate the neural network	43
Figure 27 : Foot path for right front leg.....	44
Figure 28 : Foot path for right middle leg.....	44
Figure 29 : Foot path for the right rear leg.....	45
Figure 30 : Desired vs. actual foot path of right front leg and position errors along x, y and z axis vs. time.	46
Figure 31 : Desired vs. actual foot path of right middle leg and foot position errors along x, y and z axis vs. time	47
Figure 32 : Desired vs. actual foot path of right rear leg and foot position errors along x, y and z axis vs. time	48
Figure 33 : Tripod gait footfalls produced by a fixed gait generator. RF, RM, RR = right front, middle and rear, respectively. LF, LM, LR = left front, middle and rear, respectively. The bold lines represent swing phase.	51
Figure 34 : Desired Paths for Tripod Walking. (a), (b) and (c) show the desired path for front leg, middle and rear legs, respectively. The foot position is expressed with respect to the intersection point of Gamma and Beta joint axes for each leg.	52
Figure 35 : Three of the Cruse mechanisms. Sending Leg Position versus Time is plotted with Mechanism 1, 2 and 5. AEP is the anterior extreme position and PEP is the posterior extreme position.....	56
Figure 36 : Cruse gait network using mechanisms (1, 2 and 5). L1, L2, L3 = Left front, middle and rear leg, respectively. R1, R2, R3 = right front, middle and rear leg, respectively.	56
Figure 37 : Generated Gait Pattern by Cruse Controller.....	57
Figure 38 : Automatically Generated Foot Paths for Six Legs in inches.....	57

Figure 39 : Leg configuration. (a), (b) and (c) show the front, middle and rear legs respectively.	60
Figure 40 : McKibben Actuator.....	63
Figure 41 : Festo actuator. (a) original Festo air muscle end. (b) modified end.....	64
Figure 42 : Force vs. h (contraction ratio). $h = (l_0 - l) / l_0$ where l_0 is the original actuator length. l is the length after contraction. The diameter of the actuator is 20 mm.	65
Figure 43 : Actuator Test Rig. The setup is for measurement of the pressure and length of Festo actuators.	66
Figure 44 : Length Calibration. The relationship between length and sampled potentiometer data is shown.....	67
Figure 45 : Pressure Calibration. The relationship between pressure and sampled pressure data is shown.....	67
Figure 46 : Festo Experimental Results: Length vs. Time for different pressures	68
Figure 47 : Geometry of Festo Actuator. The braided fiber turns n times.....	70
Figure 48 : Virtual Work principle applied to McKibben Actuator.	71
Figure 49 : Valve configuration for Robot V. The inlet valve is connected to a pressurized air supply and the outlet valve is connected to the atmosphere.	73
Figure 50 : Schematic of a Matrix Series 850 Valve.....	74
Figure 51 : The Schematic of a Matrix Series 750 valve.....	75
Figure 52 : Antagonistic actuators. Flexor and extensor tensile actuators drive each joint in the robot.....	77
Figure 53 : Dynamic Simulation of Robot V. Yobotics Simulation Set is used for the simulation of Robot V.....	79
Figure 54 : Flow chart of simulation.....	81
Figure 55 : Detailed algorithm of simulation.....	84
Figure 56 : Desired vs. actual joint angles of Gamma joint on right front leg.	89
Figure 57 : Errors between desired and actual joint angles	90
Figure 58 : Absolute errors between desired and actual joint angles	90

Figure 59 : Comparison of average joint angle error with and without tensioning reflex.	91
Figure 60 : Desired vs. actual foot path of right front leg with and without tensioning reflex.....	92
Figure 61 : Averages and ranges of foot position error without and with tensioning reflex.....	92
Figure 62 : Desired vs. actual joint angles and foot path on front right leg with simple gait during air walking. Figure 56 (a), (b), (c), (d) and (e) show the gamma, beta, alpha, cf and ft joints respectively. Figure 56 (f) shows the desired and actual joint angles.	96
Figure 63 : Desired vs. actual joint angles on right front leg with simple gait during ground walking. From the left top, the gamma, beta, alpha, cf and ft joint are shown. The red and blue lines represent the desired and actual joint angles.....	97
Figure 64 : Desired vs. actual foot paths for air and ground walking with pre-planned gait of right front, middle and rear legs. (a), (c) and (e) show the right front, middle and rear legs for air walking. (b), (d) and (f) show the right front, middle and rear legs for ground walking.....	100
Figure 65 : Body bounce of Robot V with pre-planned tripod gait during ground walking.....	101
Figure 66 : Desired vs. actual joint angles on right front leg with Cruse Controller during air walking. From the left top, the gamma, beta, alpha, cf and ft joint are shown. The red and blue lines represent the desired and actual joint angles	103
Figure 67 : Desired vs. Actual Joint Angles on Front Right Leg with Cruse Controller for simulated ground walking. From the left top, the gamma, beta, alpha, cf and ft joint angles are shown. The red and blue lines represent the desired and actual joint angles, respectively.	104
Figure 68 : Desired vs. Actual Foot Paths in right front, middle and rear legs for Air and Ground Walking with Cruse Controller. (a), (c) and (e) are for air walking. (b), (d) and (f) are for ground walking.....	106
Figure 69 : Body Bounce of Robot V using Cruse Controller.....	107
Figure 70 : Spectra Symbol flexible joint angle sensor is mounted on a rubber strip, which is then attached to each side of a joint.....	111
Figure 71 : Hardware setup for Robot V.	113

Figure 72 : Initial implementation of individual leg control for right middle leg in air walking. The Beta, Alpha, CF and FT joint angles are shown as well as the foot path.	115
Figure 73 : Air walking test of Robot V	117
Figure 74 : Joint angle motions vs. time for left front leg during air walking.....	117
Figure 75 : Supported walking test of Robot V.	118
Figure 76 : Joint angle motions vs. time for left front leg during supported walking.....	119
Figure 77 : Limited range of motion in serial front leg design in air walking.....	121
Figure 78 : Desired and actual foot paths of left and right front legs in air walking.....	121
Figure 79 : Parallel and serial joints designs in right Front Legs. (a) parallel gamma and beta joint designs. (b) revised serial gamma and beta joint design.	122
Figure 80 : Desired vs. Actual Joint Angle motions for air walking after design revision. (a) joint angles in left front leg. (b) joint angles in right front leg.....	122
Figure 81 : Joint angle measurement for Robot V	124
Figure 82 : Festo actuator Buckling Problem.	128
Figure 83 : FT joint angle motion vs. time in the right front leg. The red, blue and green lines represent the desired joint angles, joint angles without reflex and joint angle with reflex, respectively.	129
Figure 84 : Average Joint Angle Errors in Right Front Leg with vs. without Tensioning Reflex	129
Figure 85 : Foot trajectory of Right Front Leg. (Left) without tensioning reflex (right) with Tensioning Reflex.....	130
Figure 86 : Average and Range of foot Position Errors in Right Front Leg with vs. without Tensioning Reflex.....	130
Figure 87 : Design process from the Cockroach to Robot III and Robot V. A cockroach is video taped and positions are digitized. The joint angles motions are found and these joint angles are used for the simulated cockroach. Robot III can use the joint angles from the digitization due to its physical similarity. The joint angles of Robot V are different from those of the cockroach.....	134

Figure 88 : Digitized cockroach vs. Basic Design of Robot III.....	136
Figure 89 : Cockroach vs. basic design of Robot V	137
Figure 90 : Right Front Legs of Robot III and Robot V. (a) initial configuration of Robot III. (b) initial configuration of Robot V. (c) side view of Robot III and (d) side view of Robot V.	139
Figure 91 : Different Beta rotations between Robot III and Robot V	140
Figure 92 : Foot Position of Robot III and Robot V when Beta = 45° and Alpha =45°	141
Figure 93 : Physical Difference between Robot III and Robot V. (a) bottom view of Robot III. (b) bottom view of Robot V. In (b), the red lines show a design more similar to Robot III.	143
Figure 94 : Joint Angle Synchronization between Cockroach and Robot V. (a) joint angle measurement of cockroach. (b) joint angle measurement of Robot V. (c) pre-inclination of Gamma joint of Robot V.....	145
Figure 95 : Right Front Gamma Joint.	147
Figure 96 : Right Front Beta Joint.	147
Figure 97 : Right Front Alpha Joint.....	148
Figure 98 : Right Front CF Joint.....	148
Figure 99 : Right Front FT Joint.....	149
Figure 100 : Right Middle Gamma Joint	150
Figure 101 : Right Middle Beta Joint.	150
Figure 102 : Right Middle Alpha Joint	151
Figure 103 : Right Middle CF Joint.....	151
Figure 104 : Right Middle FT Joint.....	152
Figure 105 : Right Rear Beta Joint.	153
Figure 106 : Right Rear CF Joint.....	154
Figure 107 : Right Rear FT Joint.....	154
Figure 108 : Gamma Rotation Test. All joint commands are same through all tests except Gamma joint. Gamma joint is rotated inside or outside.	156

Figure 109 : Total Joint Torque Change corresponding to Gamma Joint Change. The Gamma joints are increased by 0.2.....	156
Figure 110 : Postural Strategy for Forward Swing. (a) cockroach's right front leg inside the ROMs of each leg. (b) Robot V's right front leg inside the ROMs of each leg.	163
Figure 111 : Beta Joint of Right Front Leg. The Beta joint with the small moment arm can not produce enough joint torque.....	166

Acknowledgements

This dissertation is dedicated to my family. My wife, HyeYoung always supported and encouraged me. I could not be here without her help and support. My parents, JinKyu Choi and Oksoon Park have supported me by prayer as well as financial support. When I became a father of two kids, I knew their burden as parents. I can not reward for them. I want to thank my two daughters, Dain and Jeongin. Frankly speaking, sometimes they bothered my work and I spent a lot of time to take care of them. But they give a lot of joy and encouragement.

It is time to thank to Dr. Roger Quinn. I do not know how I can express my gratuity. He is so kind and generous to me. Always he is waiting for me and listen my opinion. I wish to thank him for his patience, encouragement as well as financial support. I am an international student. That means my speeches and writings are not good to understand, but always he has tried to understand and listened to me. I will miss him.

I am extremely thankful to my thesis committee for their tolerance, patience, helpfulness, and kindness: Dr. Roy Ritzmann, Dr. Joseph Mansour and Dr. Robert Kirsch.

I appreciate my friends: Dr. Sangeun Choi and Won Joo. They gave me advices in terms of technical issues and regular life. We shared a lot of communications. I really appreciate them.

I wish to thank all of BioRobotics Lab guys. They are always kind and help me. I remember former alumni: Gaberial Nelson (he encouraged me and prayed for me), Satiya, Matt Birch, Robb and Tom. The following people were directly involved in Robot

V: Daniel Kingsley (design of Robot V), Brandon Rutter (low level control). I can not forget their efforts and times we spent for Robot V. I really appreciate.

Let's talk about current Biorobotics gangs: Rich Bachmann, Terence Wei, Andy Horchler, Brandon Rutter, Adam Rutkowski, Alex Boxerbaum, Bill Lewinger, Nicole Kern, Kurt Aschenbeck, Kati Daltorio and Brian Dodgson. They are all funny and kind. I really appreciate all of them. In Biorobotics lab, I really enjoy the American life and share the joyful moments. I really appreciate all of them. I will miss them and remember all of moments with them.

It was a long way to get Ph.D degree. But I really enjoy the painful time because everybody around me is always kind and generous for me. My God prepared everything for me. On my blackboard there are three words, "God with me". Always it encouraged me and always it became true. I believe it works in my future.

Locomotion Control Experiments in Cockroach Robot with Artificial Muscles

Abstract

by

Jongung Choi

This dissertation describes experimental efforts to improve the control and mechanical designs of a biologically-inspired hexapod robot. Robot V is modeled after the *Blaberus discoidalis* cockroach. It uses Festo pneumatic muscle actuators with two-way solenoid valves activated by Pulse-Width-Modulation. Robot V is capable of rudimentary walking without sensors, but walking with style requires proprioceptors to measure joint angles and load. Control circuits are described in this thesis that coordinate the robot's joints and legs using sensor data.

The Modified Moore Penrose method is used to solve the inverse kinematics problem for each of the robot's legs at a number of foot positions within the legs' workspaces. These solutions are used to train neural networks that then are used to solve the inverse kinematics problems on line as the robot moves. A Cruise controller is used to coordinate the legs into insect gaits. These controllers are tested in a dynamic simulation that models the robot's dynamics, actuators and valves. The simulated Robot V walks successfully.

The control strategies were then implemented and tested in Robot V. The robot was shown to move its legs in insect gaits while it was supported in the air such that its

feet could not touch the ground. Load feedback must be implemented before it will walk well on the ground. The robot's design was compared to Robot III and a number of problems with Robot V's design were discovered during experimentation.

Chapter 1

Introduction

The BioRobotics Lab at Case Western Reserve University has been developing insect-inspired hexapod robots for 15 years. Robot I and Robot II have six legs and use electric motors. Robot III, Robot IV and Robot V have cockroach inspired designs and are driven by pneumatic actuators. Robot III uses standard air cylinders. It can lift a payload equal to its own weight and move its legs in cockroach walking patterns. Because of its valve architecture air could not be trapped in its cylinders, which prevented it from having passive stiffness and it can not walk well. Robot IV used McKibben actuators which are also called air muscles or braided pneumatic actuators and double the number of valves. It showed promise, but its actuators failed in fatigue. The current robot called Robot V uses air muscles sold by Festo Inc. which are more robust.

The final goal of this project is to make a hexapod robot walk, run, turn and climb obstacles with agility. To do so, we are taking inspiration from cockroaches. Biologists have researched cockroach neuromechanics for decades to determine how it can move with such remarkable agility. The goal of this research is to implement that knowledge in robot designs. Daniel Kingsley [1] designed and constructed Robot V based upon cockroach kinematics and mechanics. In this dissertation, I describe a dynamic simulation of Robot V, implement insect-inspired control strategies in Robot V. and measure the robot's performance.

1.1 Cockroach-Inspired Robots

The design of Robot V is modeled after *Blaberus discoidalis* cockroach. Why was the cockroach chosen for inspiration? In nature, the cockroach has a lot of advantages. It can move rapidly over unstructured terrain and it can even climb vertical walls using its sticky pads, claws and spines. A robot that could perform these tasks would be very useful for numerous exploration, defense and humanitarian missions.

The Ritzmann Lab has been studying cockroach locomotion. They have been developing an understanding of cockroach circuits via intercellular and extracellular recordings. They also observe cockroach behavior using high speed video and digitize the results. In conjunction with the Quinn Lab joint motions have been extracted for running and climbing behaviors and dynamic simulations of cockroach behavior have been developed. These studies led to the joint degrees of freedom and ranges of motion used by cockroach legs while performing these behaviors.

Robot III was designed by Richard Bachmann [2] based upon cockroach kinematics extracted by Nelson from the Ritzmann Lab's data [3]. It has relatively small front legs with 5 degrees of freedom (DOF), larger middle legs with 4 DOF, and powerful rear legs with 3 DOF. Its cockroach design enables it to perform cockroach behaviors such as reaching in front of its body with its front legs, pitching its body up in preparation of obstacle climbing with its middle legs, and accelerating itself forward with its rear legs.

Daniel Kingsley [1] designed Robot V with the joint degrees of freedom and ranges of motion of Robot III. In addition to these cockroach features it also has an exoskeleton design and uses air muscles to activate its joints. Furthermore its valves are located in the rear of the robot to place the center of mass of the robot above the body-coxa joint of the rear legs – the same center of mass location observed in the animal. The overall weight of Robot V is 33 lbs, its length is 40 inches and its overall height is 18 inches when it is standing.

Each joint is operated by antagonistic pairs of a type of McKibben artificial muscle consisting of Festo, Inc. air muscle tubes, with their large, heavy aluminum ends replaced with smaller plastic ones [1]. Each actuator is operated by a pair of valves which are driven by pulse-width-modulation (PWM). One side of each inlet valve is connected to supply pressure and the other is connected to an inlet port of a Festo actuator. One side of each outlet valve is connected to the actuator and the other is connected to the atmosphere. The PWM method enables analog actuators to be controlled via digital commands. The duty cycle of PWM varies from 0 % to 100 %. In proportion to the duty cycle, the valve opens and closes. A 100% duty cycle means the valve is open (ON) during one PWM period. A 0 % duty cycle means the valve is closed (OFF) during a PWM period. A 50 % duty cycle means that in the first half period, the valve is open (ON) and in the next half period, the valve is closed (OFF). The mass flow of air into an actuator depends on the duty cycle. A large PWM duty cycle increases the air mass flow which causes pressure inside the actuator to increase. When the inlet valve command is ON and the outlet valve command is OFF, actuator pressure increases. On the contrary, when the inlet valve command is OFF and the outlet valve command is ON, actuator

pressure declines and tends toward atmospheric. Similarly, there are two cases in which the two commands have same values, i.e. ON or OFF. If both commands are OFF, pressurized air is trapped inside the actuator. On the other hand, the control scheme prevents the case in which both commands are ON because that would cause the supply pressure to leak directly to the atmosphere.

Robot V has proprioceptive sensors that measure its joint angle positions and actuator pressures. It has 24 joint DOF and each joint is driven by two air muscle groups. These sensor measurements are input to Robot V's off board control system via a computer board with 96 input channels. 48 channels are assigned to joint angle sensors and 48 channels are assigned to actuator pressure sensors. All sensory inputs are gathered and the controller calculates the proper duty cycle commands for each valve. These commands are sent to inlet and outlet valves through 96 output channels in the control computer. The controller software will be explained later.

1.2 Dissertation Outline

In this dissertation, the two main issues addressed are modeling and control of Robot V. A dynamic simulation of Robot V is described, which includes models of its mechanics and actuators. A control system for the robot is also developed and evaluated.

In chapter 2, previous biologically inspired robots are reviewed. Biorobots can be classified based upon the level of biological inspiration used in their design. Some of the robots in this chapter are directly inspired by biology – their design or control system is similar to a particular animal. The difficulty in doing this is that robot technology is

different and perhaps inferior to animal technology. Other biorobots are only loosely inspired by animal mechanisms. Engineers have recognized the technological gaps and differences between animal and robot components and materials and because of these differences have purposely not used direct inspiration from particular animals. Instead, they abstract basic locomotion principles from animals and implement them with available technology. However, animal mobility is superior to existing robot mobility and direct inspiration is a worthwhile research focus for two reasons: First, to develop more capable robots and second to use these robots as a model to better understand the animal.

In chapter 3, I will describe the workspace of each leg. It is essential to know this when later assigning motion paths for each leg. The desired foot path is not feasible when it is out of the range of the workspace of the leg. To find the desired joint angles which correspond to a desired path, we have to solve the inverse kinematics problem. The Modified Moore-Penrose method [4] is described and a method to solve the inverse kinematics problem is shown. Calculation of joint angles on line takes time and imposes a burden on the control computer. To reduce this burden, a trained neural network which produces joint angles is implemented.

In chapter 4, the Cruse [5] gait controller is introduced. Cruse control coordinates the legs into insect gaits. It does not determine foot or joint paths, but only switches legs to and from stance and swing.

In chapter 5, a dynamic simulation of Robot V is described. The model includes the mechanics of the robot, air muscles, and valves. Modeling of Festo actuators is not a trivial problem because the actuator force depends on pressure, length and air flow. Furthermore, this actuator is stiff when it is stretched and it buckles when it is

compressed. These nonlinearities in stiffness are difficult to model. The simulation is in Java language. The Yobotics simulation package provides libraries for the construction of the robot mechanics, integration and a well-managed window environment.

Chapter 6 describes performance metrics that I use to measure the quality of the various control systems. These metrics are used to measure the performance of the robot in simulation and in hardware experiments.

In chapter 7, the simulation results are shown. Simple gait control is applied in simulation for air walking and ground walking before it is applied to the robot. The desired and actual joint angles and paths are shown.

Before implementing feedback control schemes into the robot, joint angle and load sensors are needed. Chapter 8 describes the joint angle sensors and pressure sensors and the results of implementation of different control schemes are shown. Results are shown for individual leg control, for air walking, partially supported walking, and ground walking. After the experiments are performed and unexpected problems are reviewed. The solutions of these problems are discussed.

In chapter 9, Robot V is compared with a modeled cockroach and Robot III. The designs of Robot III and Robot V are compared. The different standing postures and forward swing strategy due to the different designs of Robot III and Robot V are described. The joint angles of Robot V are compared with those of the modeled cockroach.

In chapter 10, lessons learned through the experiments and conclusions are described. Future work is also suggested.

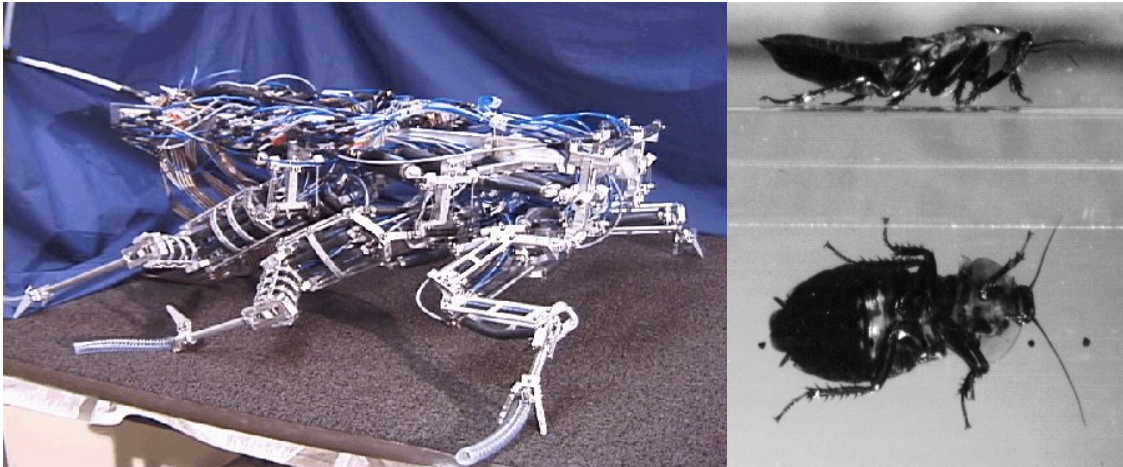


Figure 1 : Robot V. Robot V has 24 joints. There are 5, 4, and 3 joints in the front, middle and rear legs, respectively. The design of Robot is inspired by the cockroach called *Blaberus discoidalis* shown on the right.

References

- 1 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.
- 2 Bachmann, R. J., A Cockroach-Like Hexapod Robot for Running and Climbing. M.S. thesis. 2000.
- 3 Watson, J. T., Ritzmann, R.E. Leg kinematics and muscle activity during treadmill running in the cockroach, *Blaberus discoidalis* : I. Slow running. *J. Comp. Physiol., A* 182: 11-22, 1998.
- 4 Mussa-Ivaldi, F.A., Hogan, N. Integrable Solutions of Kinematic Redundancy via Impedance Control. *Int. J. of Robotics Research*, Vol. 10, No. 5, pp. 481-491, October, 1991.
- 5 Cruse, H. What mechanisms coordinate leg movement in walking arthropods?, *Trends in Neural Science*, Vol. 13, pp. 15-21, 1990.

Chapter 2

Background

For several decades researchers have been trying to make ground robots with mobility that rivals that of animals. They have been unsuccessful. Because animals have greater mobility than existing ground vehicles, biologically inspired approaches have been pursued for the past 15 years. Some of those robots are reviewed in this chapter. This review includes some examples of insect-inspired robots, but does not include robots that are based on more abstracted biological principles such as PROLERO [1], RHex [2], Whegs [3] and Sprawlita [4] because these robots are not comparable to Robot V. They do not have multi-segmented legs or active controls that permits them to move their joints independently in an insect like manner. Robots such as Robot V promise much greater mobility on unstructured terrain because of their multi-segmented legs. However, much more research is necessary to improve their designs and control strategies before they are fieldable robots.

2.1 Insect-inspired robots

The following is a brief and non inclusive list of insect-inspired robots.

Protobot - Protobot[5] is pneumatically actuated and its geometry is designed after the American cockroach, *Periplaneta americana*. Its body measures 58 cm × 14 cm × 23 cm length, width, and height. The robot weighs approximately 11kg. It does not enjoy the kinematic advantages of the cockroach. Protobot has only 3 DOF in each leg. The simplified configuration is convenient for design and control, but the kinematic redundancy of cockroach legs provides them with superior functionality. Protobot uses off-the-shelf double acting cylinders with three-way valves. Protobot's valves are actuated with pulse frequency modulation (PFM) with 10 ms pulses. With the simplified leg design and PFM valve control, Protobot walked successfully.



Figure 2 : Protobot [5]. Protobot is modeled after *Periplaneta americana* cockroach. It has 18 DOF. It uses the double acting cylinder actuator.

TUM – Pfeiffer et al. [6] at the Institute of Mechanics at the Technical University in Munich developed the six-legged TUM walking robot. TUM is modeled after the

walking stick insect, *Carausius morosus*. Like Protobot, each leg has 3 DOF. TUM has no central supervision. The single leg controller (SLC) produces individual leg paths. Even if a leg slips and detects an obstacle, the SLC can produce the adjusted motion which is one of swing, retract, re-swing, stance and protract. During the swing phase, the SLC detects overload in the leg joints. Whenever the strain gauge signal exceeds the threshold value the avoidance mechanism is activated. On the other hand, for global leg coordination only the AEP and PEP threshold values are changed during walking. The coordination of each leg is determined by the sum of the influences exchanged among the legs.

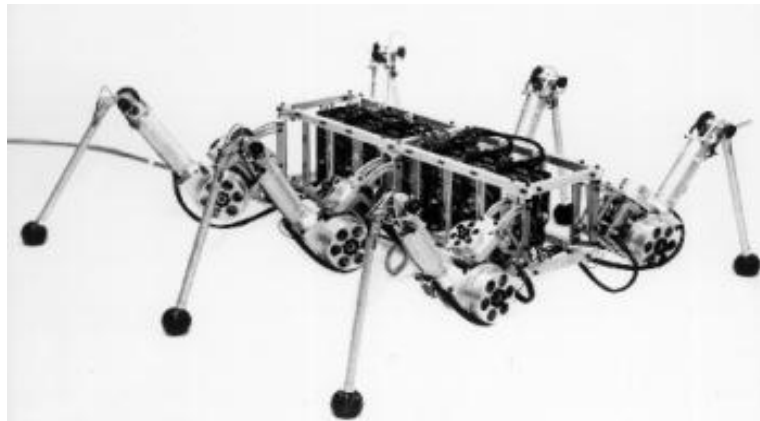


Figure 3 : TUM walking robot [6]. Each leg has 3 joints which are driven by 3 DC motors.

AirInsect – T. Kercher et al. [7] developed this hexapod inspired by the stick insect, *Carausius morosus*. Pneumatic actuators made by FESTO Inc. are used. Each leg has 3 DOF and the robot is lightweight because pneumatic muscle actuators have high power to weight ratios.

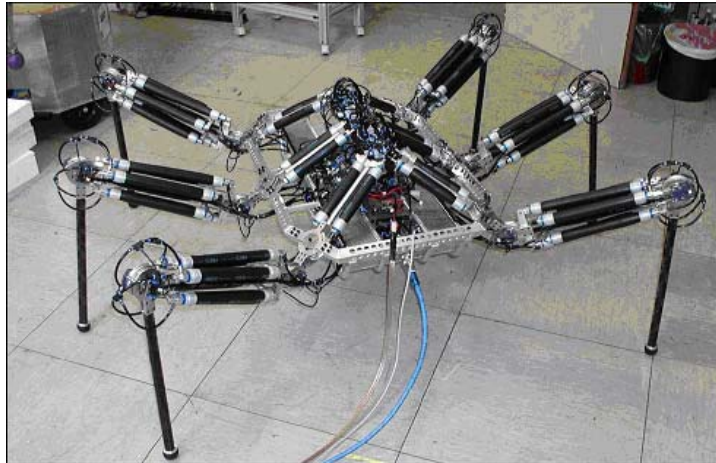


Figure 4 : AirInsect

Robot I, II – These were the first insect-inspired robots developed at Case Western Reserve University [8], [9]. They are inspired by the stick insect and have off-board energy and control systems. Figure 5 (a) shows Robot I and Figure 5 (b) shows Robot II. Robot I is driven by 12 electric motors – one motor drives each of its joints. Each leg has one linear joint and one rotational joint. Robot I walked in a continuum of insect gaits using the stick insect leg coordination mechanisms reviewed by Cruse (1990) [10] and detailed by Dean [11]. Each leg on Robot II has three rotational joints and it is driven by 18 electric motors. It has an insect-like sprawled posture. Using its distributed control system and leg reflexes, Robot II can walk in a continuum of insect gaits using Cruse control, walk omni-directionally, and move over irregular terrain. For example, in Figure 5 (b), when a leg on Robot II encounters a gap in the slatted substrate, it searches for a foothold.

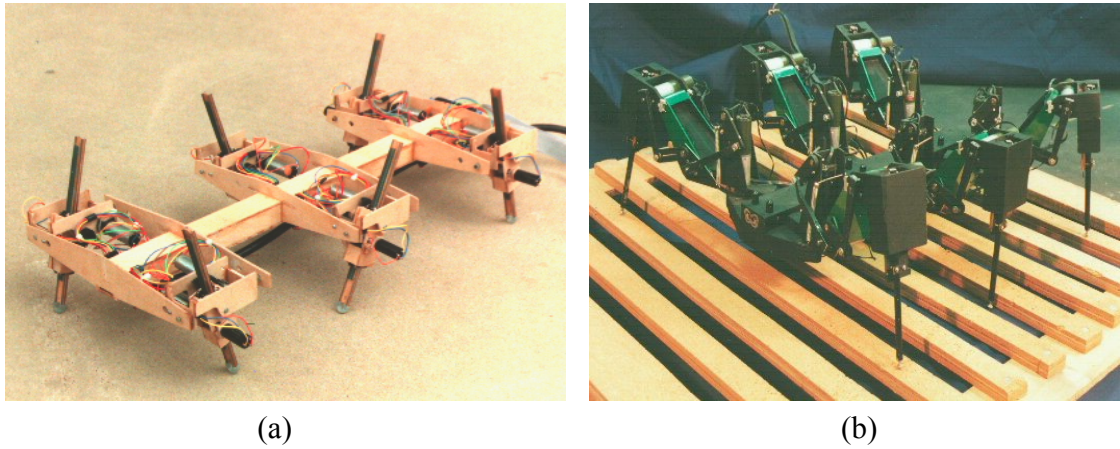


Figure 5 : Robot I and II. (a) Robot I and (b) Robot II.

Robot III – This is the first Case robot modeled after *Blaberus discoidalis* cockroach [12]. It has 24 joint degrees of freedoms (DOF). There are 5, 4 and 3 DOF at the front, middle and rear legs, respectively, to accurately model the motions of cockroach legs. Air cylinders are used as actuators because of their higher power to weight ratio as compared to electric motors. Joint angles are measured using potentiometers at each joint. A total of 48 solenoid valves, controlled by Pulse Width Modulation (PWM), supply air to the actuators. The pulse width determines the opening duration of a valve. Using its powerful actuators and an efficient posture control scheme, Robot III can lift a payload of 30 lb which is equal to its own body weight and withstand larger perturbations while it is standing. The front and middle legs are kinematically redundant, which means there are an infinite number of solutions for joint angles that will place their feet in a particular location. These leg designs have the advantage increasing the dexterity of the legs such that they can better perform their locomotion functions. A neural network was trained to perform the inverse kinematics solutions for these legs

using a biologically-inspired strategy to produce the training data. Robot III can move its legs smoothly and in a coordinated tripod gait while suspended in air. In spite of its successful posture control and smooth leg motions, Robot III can not walk well. There are two reasons why Robot III can not walk successfully. The actuation system is not fast enough to overcome disturbances that occur in normal swing-stance cycles because it is limited by the speed of moving air. The joints need passive stiffness to complement the control system. However, the three way valves can not trap air in the air cylinders and provide the joints with passive stiffness.

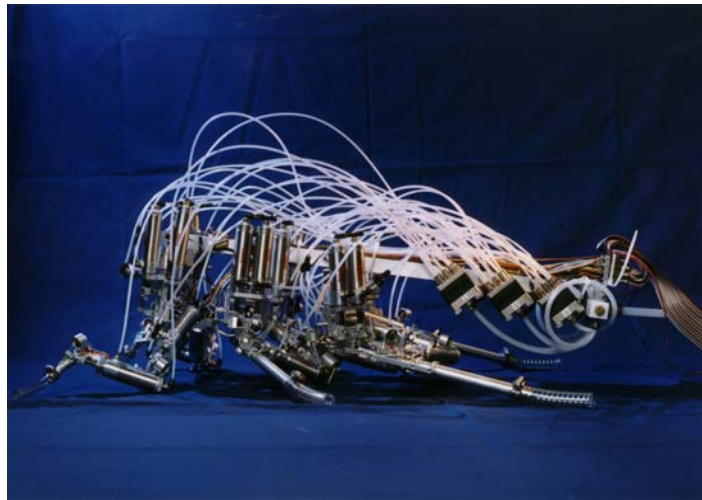


Figure 6 : Robot III

Robot V - Robot V is modeled after *Blaberus discoidalis* cockroach. Daniel Kingsley [13] designed the robot. It has 6 legs and has 5, 4 and 3 degree of freedom at front, middle and rear leg, respectively. The Festo pneumatic muscle actuator is used because of its high power to weight ratio and because it can provide the joints with passive compliance. Robot V can walk without joint angle sensors. The 96 solenoid valves controlled by Pulse Width Modulation (PWM) input or output air to or from the

actuators. The two-way valve system enables air to be trapped in the actuators. For closed loop control, flexible joint angle sensors and pressure sensors were mounted and calibrated.

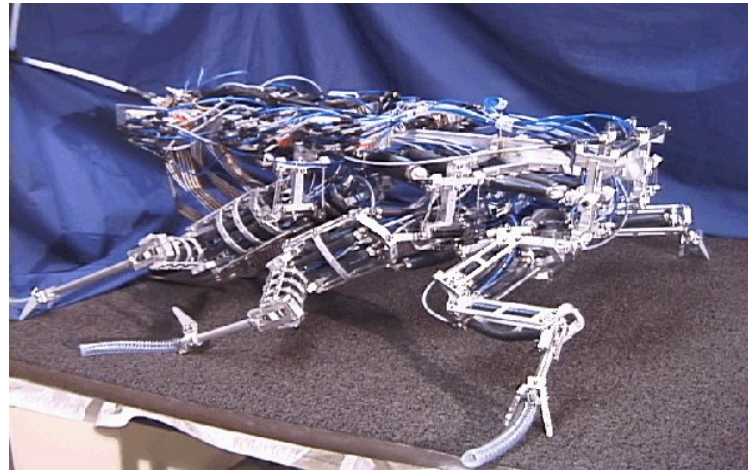


Figure 7 : Robot V

References

- 1 Martin-Alvarez, A., De Peuter, W., Hillebrand, J., Putz, P., Matthyssen, A. and De Weerd, J. F. Walking robots for planetary exploration missions. *Second World Automation Congress (WAC '96)*, Montpllier, France, May 27-30, 1996.
- 2 Sarnli, U., Buehler, M. and Koditschek, D. E. RHex: a Simple and highly mobile hexapod robot. *The Intenational Journal of Robotics Reserch*, 20 (7) : 616-631, July 2001.
- 3 Quinn, R.D., Nelson, G.M., Ritzmann, R.E., Bachmann, R.J., Kingsley, D.A., Offi, J.T. and Allen, T.J. Parallel Strategies For Implementing Biological Principles Into Mobile Robots, *Int. Journal of Robotics Research*, Vol. 22 (3) pp. 169-186. 2003
- 4 Clark, J., Cham, J., Bailey, S., Froehlich, E., Nahata, P., Full, R. and Cutkosky, M. Biomimetic Design and Fabrication of a Hexapedal Running Robot. *Proceedings of*

- the 2001 IEEE International Conference on Robotics and Automation (ICRA 2001)*, Vol. 4, pp. 3643 – 3649. 2001
- 5 Delcomyn, F., Nelson, M.E. Architectures for a biomimetic hexapod robot. *Robotics and Autonomous Systems*, Vol. 30, pp. 5-15, 2000.
 - 6 Pfeiffer, F., Eltze, J. and Weidemann, J.-J. The TUM-walking machine. *Intelligent Automation and Soft Computing*, Vol. 1, pp. 307-323, TSI Press Series, 1995.
 - 7 Kerscher, T., Albiez, J., Zoellner, J. M. and Dillmann, R. AirInsect – A New Innovative Biological Inspired Six-Legged Walking Machine Driven by Fluidic Muscles, Proceedings of IAS 8, The 8th Conference on Intelligent Autonomous Systems, Amsterdam, The Netherlands, 10-13 March 2004
 - 8 Espenschied, K. S., Quinn, R. D., Chiel, H. J., and Beer, R. D., Leg Coordination Mechanisms in Stick Insect Applied to Hexapod Robot Locomotion, *Adaptive Behavior*, Vol. 1, No. 4, pp. 455-468, 1993.
 - 9 Espenschied, K. S. and Quinn, R. D., Biologically-Inspired Hexapod Robot Design and Simulation, *AIAA Conference on Intelligent Robots in Field, Factory, Service and Space*, Houston, Texas, March 20-24, 1994.
 - 10 Cruse, H. What mechanisms coordinate leg movement in walking arthropods?, *Trends in Neural Science*, Vol. 13, pp. 15-21, 1990.
 - 11 Dean, J., Brunn, D., Bartling, C., Cruse, H., Cymbalyuk, G., Dreifert, M. and Schmitz, J. Control of hexapod walking using artificial and real neurons. *BIONA-report 9*: 17 – 36. 1995
 - 12 Nelson, Gabriel. Learning about control of legged locomotion using a hexapod robot with compliant pneumatic actuators, Ph. D dissertation. May 2002
 - 13 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.

Chapter 3

Leg Control

This chapter describes an inverse kinematics solution for the legs of Robot V. This problem is not straightforward because the front and middle legs are kinematically redundant. The strategy follows that which Nelson used for Robot III [2], but the method of solution is different. First, the workspace for each leg is established using forward kinematics. Next, the Modified Moore-Penrose method is used to solve the inverse kinematics problem for a three-dimensional grid of foot positions lying in the workspace. This method is computationally intensive and is only appropriate for off line solutions. These solutions are used as training data for a neural network that then solves the general inverse kinematics problem for each leg. Before the neural network is applied to the robot, foot paths using the joint angles from the neural network are compared with desired foot paths to check the performance of the neural network.

3.1 Workspace

The goal of the work described in this chapter is to solve the inverse kinematics problems for each of the legs of Robot V. It has 5, 4 and 3 DOF in its front, middle and rear legs, respectively. The front and middle legs are kinematically redundant, which

complicates the problem because there is an infinity of solutions. I find an optimal solution given a biologically inspired cost function. The first step in this process is to determine the workspace for each leg using forward kinematics. Fortunately, because of symmetry of the leg pairs, only 3 solutions must be found. Figure 8 (a) shows the inertial coordinate system. The configuration of legs on the right side is mirrored to the left side. Once the workspaces of the right side legs are determined, the x and z positions can be applied to the left side without sign changes. The y positions are multiplied by -1 to apply to the left side.

In the inertial coordinate system (Figure 15), the direction of the x is in the direction of forward motion of the robot. The direction of y is directed from the right front leg frame to the left front leg frame. The direction of z is upward. This inertial coordinate system is used throughout this dissertation in robot simulations and for the real robot. To find the workspace of each leg, the forward kinematics equation for each leg must be developed. The foot position of each leg is represented by a position vector originating from the intersection of the Gamma and Beta axes (Figure 15). Figure 8 (b) shows the rotational directions of the right front leg. Each front leg has 5 links and 5 joints which are Gamma (rotation about z axis), Beta (rotation about y axis), Alpha (rotation about x axis), Coxa-Femur or CF (rotation about z axis) and Femur-Tibia or FT (rotation about z axis), which rotate with respect to the local coordinate system.

The foot position of each leg can be expressed as

$$\begin{aligned}
 P_{foot} &= C_{01}L_1 + C_{02}L_2 + C_{03}L_3 + \dots + C_{0n}L_n \\
 &= C_{01}(L_1 + C_{12}(L_2 + C_{23}(L_3 + C_{34}(L_4 + \dots + C_{n-2,n-1}(L_{n-1} + C_{n-1,n}L_n))))))
 \end{aligned} \tag{1}$$

Where $C_{01}, C_{02}, C_{03}, C_{04}, \dots, C_{0n}$ are the rotational matrices and $L_1, L_2, L_3, L_4, \dots, L_n$ are the position vectors from the previous joint to the current joint with respect to the current local coordinate system. The advantage of this notation is that once we know the rotation matrix ($C_{i-1,i}$) between the previous coordinate system and the current coordinate system and we know the relative position vector (L_i) from the previous joint to the current joint with respect to the current coordinate system, all joints and foot positions can be expressed using these rotational matrices and the position vectors. All joint position and foot position vectors originate from the intersection point of the Gamma and Beta axes in the inertial coordinate system. The Euler rotational matrices about the x, y and z axes are C_x , C_y and C_z :

$$\begin{aligned}
 C_x &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix}, & C_y &= \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix}, \\
 C_z &= \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}
 \end{aligned} \tag{2}$$

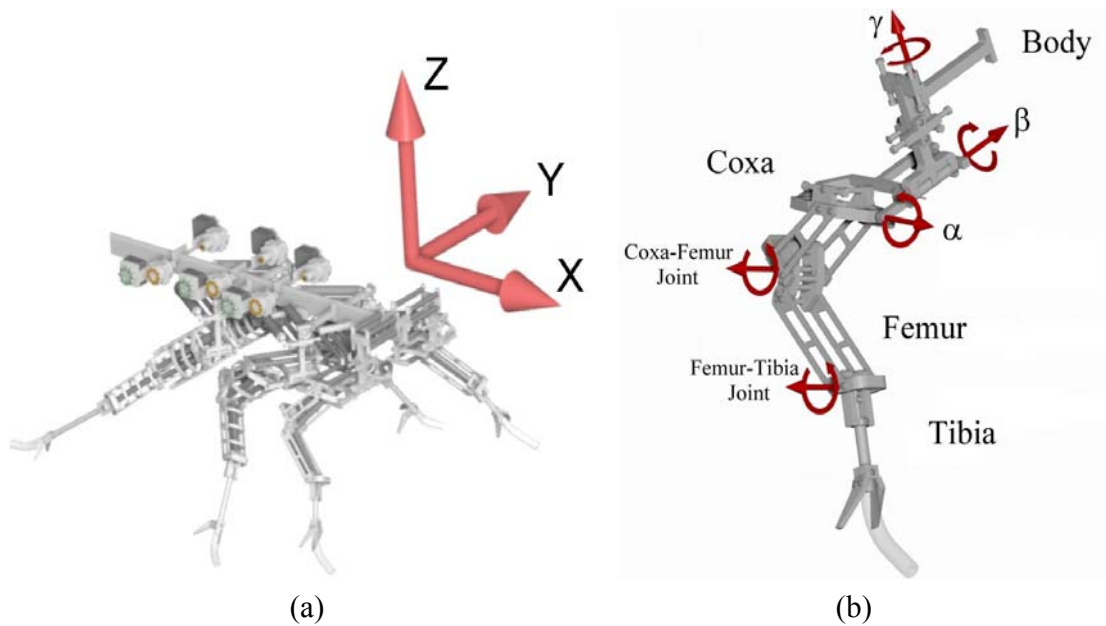


Figure 8 : Inertial coordinate system and direction of axis of rotations.

3.1.1 Workspace of Front Leg

The design of the front leg is complicated because the Gamma rotation axis intersects with the Beta rotation axis. Furthermore, the front leg is designed as a kinematically redundant manipulator for dexterity. Therefore, there are infinite solutions to the inverse kinematics problem for any desired foot position. An alternate solution is to consider the foot position and orientation as desired parameters and then solve for the joint angles. This problem would be over determined rather than the underdetermined problem that I solve here. The reason we design the front leg as a kinematically redundant manipulator is that a function of front leg is to reach onto obstacles, which requires the extra dexterity gained from incorporating extra joints. Figure 9 shows the fully assembled right front leg. The coordinate system for each leg is aligned with the

coordinate system for the body, which is shown in Figure 15. The front leg has 5 segments and 5 joint rotations which are the Gamma, Beta, Alpha, Coxa-Femur (CF) and Femur-Tibia (FT). Table 1 shows the segment lengths of the right front leg. In the initial state (all joints are set to 0 and all leg segment are in the x-y plane except the Gamma segment), Gamma, Beta, Alpha, CF and FT joints rotate about the z, y, x, z and z axis, respectively.

Due to actuator contraction limits, each joint has a reduced range of motion. Table 2 shows the right front leg's maximum range of motion based on the original design of Robot V [1]. However this range of motion was measured when the joints were pushed by hand to their mechanical limits. These extreme positions were not produced by the actuators. The real ranges of motion are less. In this chapter, to present the inverse kinematics solution, the mechanically limited range of motion will be used. This is done because Nelson found that it is important to have inverse kinematics solutions outside of the true range of motion when proportional position control is used [2]. When the solution is implemented, the real, more conservative, ranges of motion will be used to design footpaths.

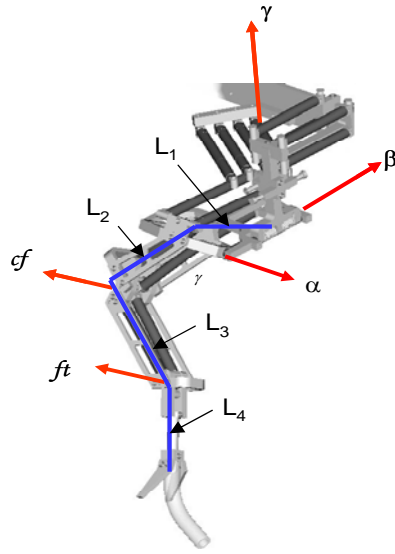


Figure 9 : Fully assembled right front leg.

Segment	Length (inches)
L1	4.452
L2	5.036
L3	5.864
L4	5.80

Table 1 : Segment lengths of right front leg in inches.

Rotation Axis	Range of Motion (degrees)
γ	-20 ~ 20
β	-70 ~ 0
α	45 ~ 65
cf	0 ~ 50
ft	-30 ~ 30

Table 2 : Range of motion of right front leg in degrees.

The forward kinematics for the front leg's foot position can be expressed as

$$\begin{aligned}
 P_{foot} &= C_{01}L_1 + C_{02}L_2 + C_{03}L_3 + C_{04}L_4 \\
 &= C_{01}(L_1 + C_{12}(L_2 + C_{23}(L_3 + C_{34}L_4)))
 \end{aligned} \tag{3}$$

The rotation matrices C_{01} , C_{12} , C_{23} and C_{34} are

$$C_{01} = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix},$$

$$C_{12} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix}, \quad C_{23} = \begin{bmatrix} \cos cf & -\sin cf & 0 \\ \sin cf & \cos cf & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

$$C_{34} = \begin{bmatrix} \cos ft & -\sin ft & 0 \\ \sin ft & \cos ft & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Equation 3 was used to determine the workspace of the right front leg, which is shown in Figure 10, Figure 11 and Figure 12. Sets of joint angles were inserted into this equation to determine the resulting foot position which was then plotted on these figures. This was done for the entire ranges of motion for the joints. Figure 10 shows the top view of the workspace of the right front leg. The range of motion of the foot in the x direction is approximately -8 to 10 inches and the y range of motion is approximately -6 to -14 inches. Figure 11 shows the front right foot can reach from -11.5 to -2 in inches in the z direction.

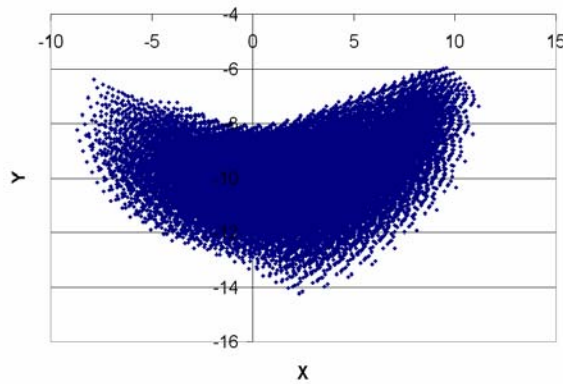


Figure 10 : X vs. Y plot of workspace of right front leg in inches.

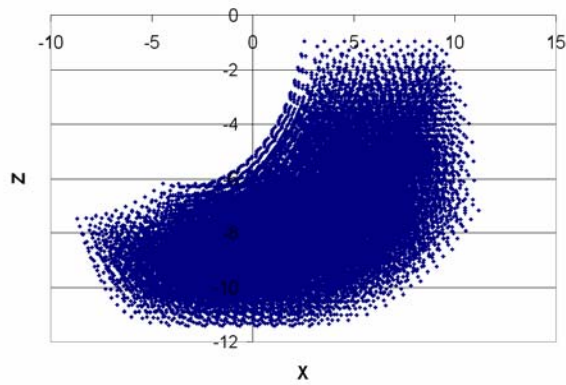


Figure 11 : X vs. Z plot of workspace of right front leg in inches.

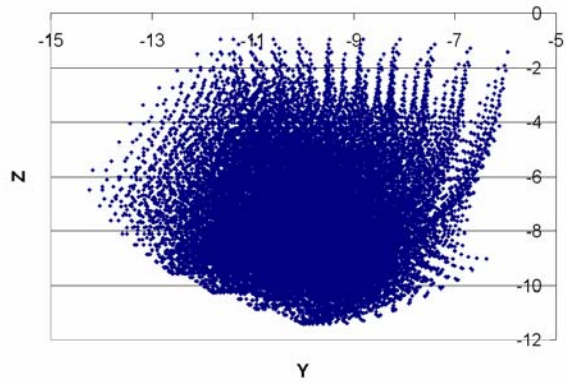


Figure 12 : Y vs. Z plot of workspace of right front leg in inches.

3.1.2 Workspace of Middle Leg

Figure 13 shows the assembled right middle leg. The middle leg has 5 segments and 4 joints which are the Beta, Alpha, CF and FT joints. Table 3 shows the segment lengths of the right middle leg. The rotation of the Gamma joint is fixed at -35 degrees. Table 4 shows the joint ranges of motion of the right middle leg.

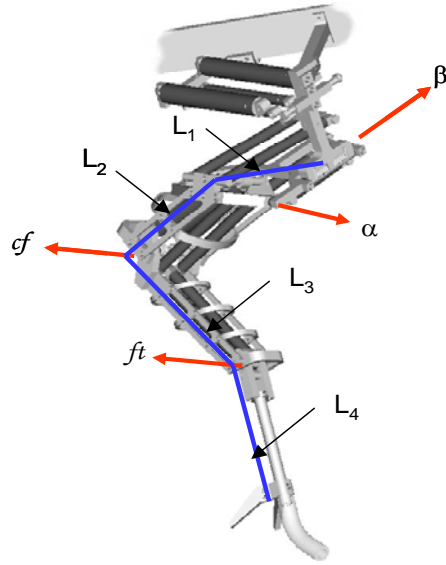


Figure 13 : Right middle leg.

Segment	Length (inches)
L1	4.15
L2	4.334
L3	7.16
L4	6.625

Table 3 : Segment lengths of right middle leg in inches.

Rotation Axis	Range of Motion (degrees)
β	-50 ~ -10
α	30 ~ 60
cf	0 ~ 30
ft	-40 ~ 40

Table 4 : Joint ranges of motion of right middle leg in degrees.

The forward kinematics equation for foot position of the middle leg is the same as equation (1) except that the rotational matrix for the Gamma joint, C_{0i} , is constant, and the leg segment lengths are different.

$$C_{0I} = \begin{bmatrix} \cos 35^\circ & -\sin 35^\circ & 0 \\ \sin 35^\circ & \cos 35^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \quad (5)$$

The forward kinematics equation for the right middle leg was used to determine its workspace. Figure 14, Figure 15 and Figure 16 show the top view, side view and front view of the workspace of the right middle leg, respectively in inches.

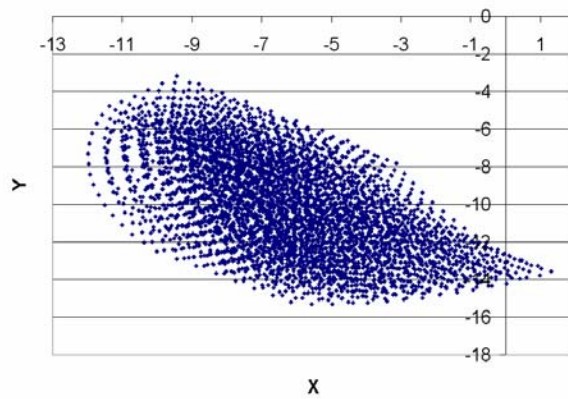


Figure 14 : X vs. Y plot of workspace of right middle leg in inches.

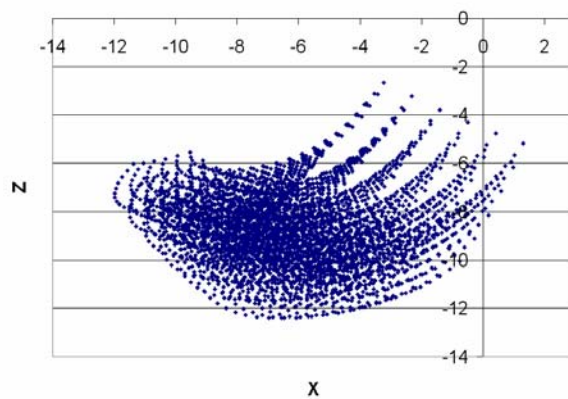


Figure 15 : X vs. Z plot of workspace of right middle leg in inches.

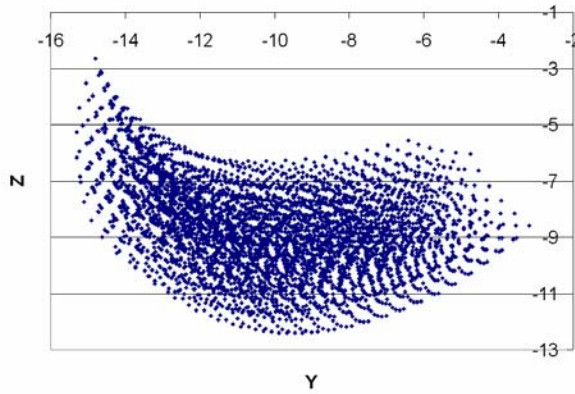


Figure 16 : Y vs. Z plot of workspace of right middle leg in inches.

3.1.3 Workspace of Rear Leg

The rear leg has 4 links and 3 joints which are the Beta, CF and FT joints. Figure 17 shows the rear leg and Table 5 shows the segment lengths of right rear leg. The most complicated part of the rear leg is the second link which is located between β and cf joint. The second link consists of two elements. The position vector L_1 shifts -1.75 along the x axis. But the position vector of L_2 shifts -5.5 along the x axis in the local coordinate system and then rotates about the z axis by 35 degree and rotates about the new y axis by -30 degree in the new coordinate system.

The position vector between β and cf joints, $L_{intermediate}$ can be expressed

$$L_{intermediate} = L_1 + C_{12}L_2 , \quad (6)$$

$$C_{12} = \begin{bmatrix} \cos(35^\circ) & -\sin(35^\circ) & 0 \\ \sin(35^\circ) & \cos(35^\circ) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(-30^\circ) & 0 & \sin(-30^\circ) \\ 0 & 1 & 0 \\ -\sin(-30^\circ) & 0 & \cos(-30^\circ) \end{bmatrix}$$

The forward kinematics equation for the right rear foot can be expressed as

$$\begin{aligned} P_{foot} &= C_{01}L_{intermediate} + C_{03}L_3 + C_{04}L_4 \\ &= C_{01}(L_1 + C_{12}(L_2 + C_{23}(L_3 + C_{34}L_4))) \end{aligned} \quad (7)$$

Where C_{23} is the rotation of the CF joint about the z axis and C_{34} is the rotation of the FT joint about the z axis.

Table 6 shows the joint ranges of motion of the right rear leg. The γ rotation is fixed at -35 degree and then the intermediate link (Link1 + Link2) rotates about the y axis (β rotation). The CF and FT joints rotate about the z axis.

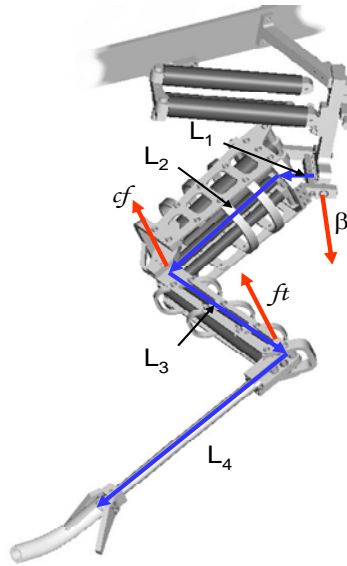


Figure 17 : Right rear leg.

Segment	Length (inches)
L1	1.75
L2	5.5
L3	7.74
L4	9.875

Table 5 : Segment lengths of right rear leg in inches.

Rotation Axis	Range of Motion (degrees)
β	-20 ~ 5
cf	0 ~ 40
ft	-40 ~ 40

Table 6 : Joint ranges of motion of right rear leg in degrees.

Forward kinematics was used to determine the workspace of the foot of the right rear leg. Figure 18, Figure 19 and Figure 20 show the top view, side view and front views of the workspace of the right rear leg, respectively. The rear leg has only 3 DOF and this limits its movements. It is not kinematically redundant, which means the joint angles are unique for any given foot position. At $z = 8$ in Figure 26, the rear leg has a maximum stance length of 4 inches (between -14 and -10). This is a short stride length as a percentage of body length relative to the cockroach.

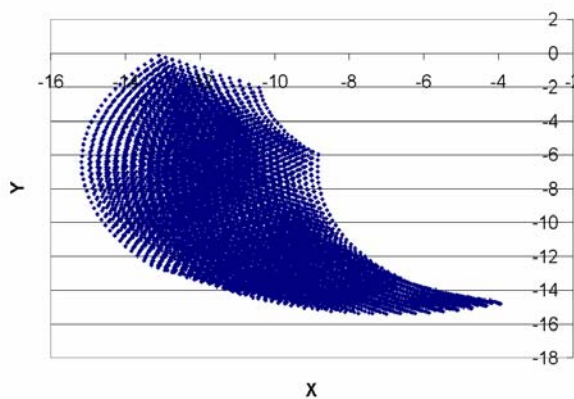


Figure 18 : X vs. Y plot of workspace of right rear leg in inches.

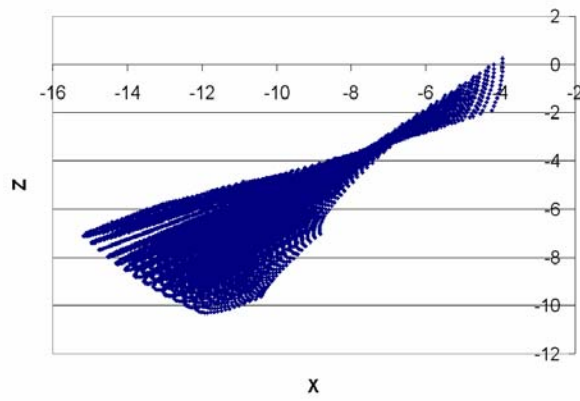


Figure 19 : X vs. Z plot of workspace of right rear leg in inches.

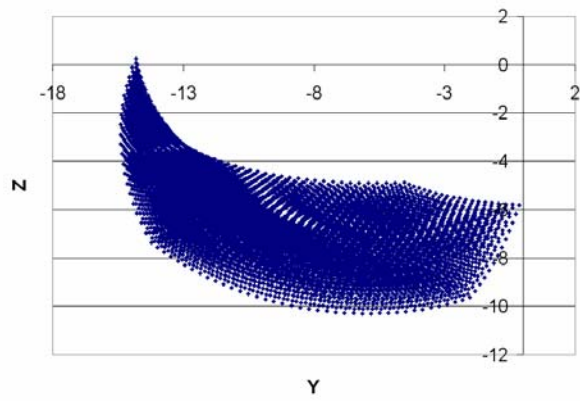


Figure 20 : Y vs. Z plot of workspace of right rear leg in inches.

3.2 Inverse Kinematics

3.2.1 Modified Moore-Penrose Method

When solving the inverse kinematics problem for kinematically redundant legs, we choose to achieve desired Cartesian foot locations by using joint angles that minimize deviations from their midrange positions, which minimize the chances of encountering joint limits. This problem is mathematically equivalent to minimizing the strain energy in conservative elastic elements acting on the joints while driving the foot to desired Cartesian positions [2]. The equilibrium angles for the springs are set to the midrange joint positions. The straightforward constrained optimization is

$$H(q, \lambda) = \frac{1}{2} \Delta q^T K_j \Delta q + \lambda^T (p_{foot/d} - p_{foot/a}) \quad (8)$$

where $\Delta q = (q - q_{mid})$. After differentiating with respect to q and λ and replacing λ with F_{foot} , we get two equations,

$$J^T F_{foot} = K_j \Delta q, \quad \text{where } J = \left[\frac{\partial p_{foot/a}}{\partial q} \right] \quad (9)$$

$$p_{foot/d} = p_{foot/a} \quad (10)$$

Eqn. (9) represents the static equilibrium. Eqn. (10) places the foot position at the desired foot position. When we solve inverse kinematics for a redundant manipulator, Eqn. (9) and Eqn. (10) should be satisfied to minimize the strain energy

In recent decades, many researchers have tried to find robust inverse kinematics solutions for kinematically redundant manipulators. The problem is to reliably choose the best solution among an infinity of solutions. The manipulator kinematics can be described by an N-dimensional vector $q = [q_1, q_2, q_3, \dots, q_N]^T$ and the workspace of the manipulator can be described by an M-dimensional vector $p_{foot} = [p_1, p_2, p_3, \dots, p_M]^T$. If

$M < N$, the manipulator is called kinematically redundant. In forward kinematics, let the workspace of manipulator, p be described as $p_{foot} = p_{foot}(q)$

The differential equation for the kinematics of the manipulator can be expressed as

$$dp = J(q)dq$$

$$\text{where, } J(q) = \frac{\partial p}{\partial q} \quad (11)$$

If we can find the mapping $q = q(p_{foot/d})$, we can move the end-effector to the desired position $p_{foot/d}$ in the joint space. To find the mapping $q = q(p_{foot/d})$, we have to integrate $dq = J^{-1}(q)dp_{foot/d}$. Many researchers have tried to find integrable inverse kinematics solutions. Mussa-Ivaldi and Hogan [3] defined the integrable inverse kinematics function, $J^*(q)$ which is called Modified Moore-Penrose (MP) pseudoinverse.

$$dq = J^*(q)dp_{foot/d} \quad (12)$$

Where J^* is

$$J^* = (K_j - \Gamma)^{-1} J^T (J (K_j - \Gamma)^{-1} J^T)^{-1} \quad (13)$$

$$\Gamma \Rightarrow \Gamma_{i,l} = \sum_{k=1}^M \frac{\partial^2 p_{foot/a,k}}{\partial q_l \partial q_i} F_{foot,k} \quad (14)$$

$$\Gamma_{i,l} = \frac{\partial^2 p_{foot/a,x}}{\partial q_l \partial q_i} F_{foot,x} + \frac{\partial^2 p_{foot/a,y}}{\partial q_l \partial q_i} F_{foot,y} + \frac{\partial^2 p_{foot/a,z}}{\partial q_l \partial q_i} F_{foot,z} \quad (15)$$

New joint angles, q_{new} can be obtained using this pseudo-inverse as follows:

$$q_{new} = q_{old} + J^* dp_{foot/d}, \quad \text{where } J^* \text{ is from Equation (13),} \quad (16)$$

The new foot force, $F_{foot,new}$ is

$$F_{foot,new} = F_{foot,old} + (J (K_j - \Gamma)^{-1} J^T)^{-1} dp_{foot/d} \quad (17)$$

The force in Eqn. (14) is replaced by Eqn. (9).

$$F_{foot} = (J J^T)^{-1} J K_j \Delta q \quad (18)$$

The result is an “optimal” solution that determines joint angles for any particular foot position. If the joints are imagined to have internal elastic springs, this solution minimizes strain energy in those springs. This is equivalent to the solution that places the joint angles as close as possible to their mid-range positions [2]. Animals are thought to solve this problem in this way [4].

3.2.2 Inverse Kinematics Solution

While the robot is walking, the inverse kinematics problem must be solved to find the desired joint angles to place the foot at the desired location at every time step. The calculation of inverse kinematics as described in the previous section is computationally intensive. So, to lighten the online computational load, we will use the MP method to find solutions which can be used to train a neural network, which can then be used to solve the inverse kinematics problem online. Once the foot position (x, y and z) is input to the neural network, the trained neural network produces the joint angles (Gamma, Beta, Alpha, CF and FT) without explicitly solving the inverse kinematics problem.

For calculating training data for the neural network, a cubic mesh of foot positions is chosen inside the workspace. The MP method is used to solve off line for joint angles

for each point in this mesh. The workspace for the right front leg is $-12 \leq x \leq 2$, $-16 \leq y \leq -4$ and $-12 \leq z \leq -6$. Figure 21 shows the mesh of foot positions used for the right front leg. Points having the same y values are represented by the same color. By using Eqn. (12)~Eqn.(18), I found reachable joint angle solutions for each point of the mesh. Because some points produce singular solutions, the robot foot can not reach those points. Figure 22 shows reachable solution points for the right front leg. When I designed the desired foot path, I chose a path with a constant y value. I chose the plane where the y value is constant and the foot has a maximum range of motion. According to Figure 22 when $y=10$, the reachable points widely range in x and z . In section 4.5, I will describe the foot path design further.

For the right middle leg, the cubic mesh points are placed in the region defined by $-12 \leq x \leq 2$, $-16 \leq y \leq -4$ and $-12 \leq z \leq -6$. Figure 23 shows the reachable solution points of the right middle leg. The swing height of the middle legs does not need to be as high as the front leg. So, the plane where $y=12$ was chosen as the desired path plane for the middle leg.

For the right rear leg, the cubic mesh is located in the region defined by its workspace $-15 < x < -4$, $-13 < y < -4$ and $-10 < z < -4$. Figure 24 shows reachable solution points of the right rear leg. If we design the desired path on a y -plane which is less than 10, it will cause the body to roll about the z axis. So to prevent body rolling, the desired path of the rear leg should be on the $y=10$ or larger plane.

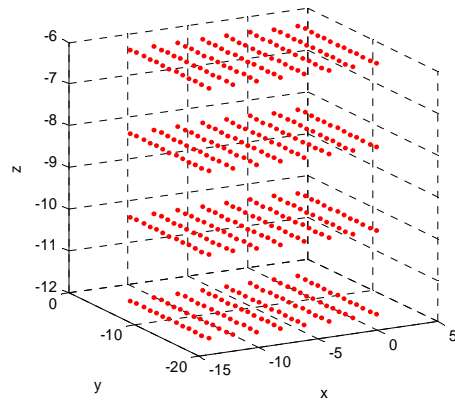


Figure 21 : Cubical points in the range of motion of right front leg.

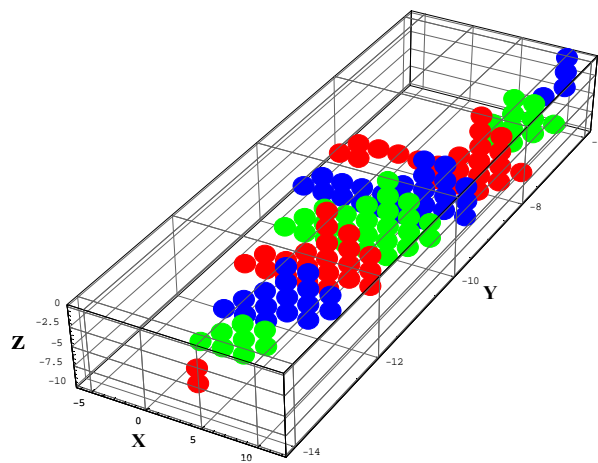


Figure 22 : Reachable points of right front leg.

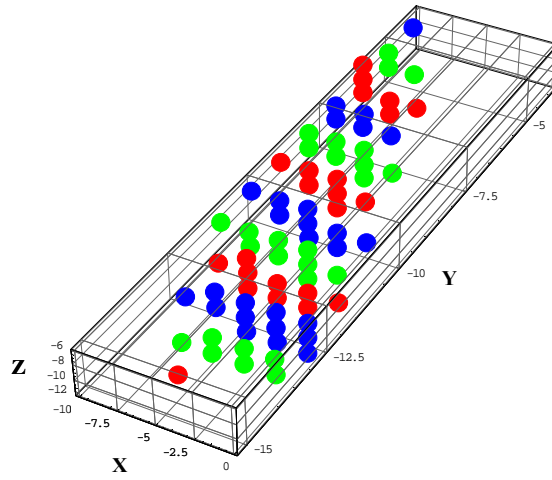


Figure 23 : Reachable points of right middle leg.

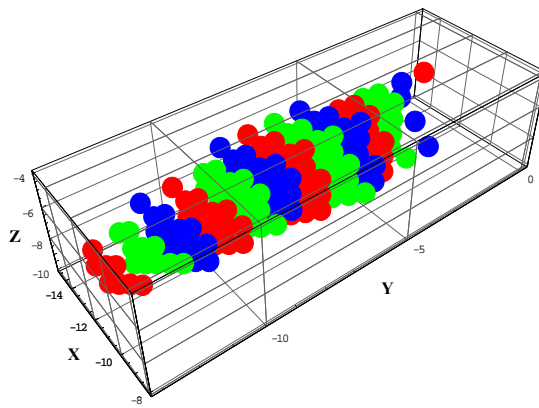


Figure 24 : Reachable points of right rear leg.

3.3 Neural Network

The previous sections described how I solved the inverse kinematics problem for a mesh of points in the workspaces of each of the legs. In this section these solutions will be used as training data for a neural network that will solve the inverse kinematics problem online for any given foot position.

To find desired joint angles without explicitly solving inverse kinematics online, solutions could be tabulated and referenced when needed. This method would require an extensive table or a method of interpolating the tabulated data. I chose to use a neural network instead of a look up table. A well trained neural network essentially stores the tabulated values (training data) and interpolates solutions not tabulated. It is a good function generator and a convenient way to rapidly find solutions. However, the neural network does not produce the exact inverse kinematics solution. It produces approximate joint angles which are corresponding to the desired foot position. For an industrial robot, a precise path or precise commanded joint angles are necessary for a precision job such as welding. But the compliant joints of Robot V compensates for joint command errors. Therefore, a neural network solution was expected to be sufficient and computationally attractive.

As described in the previous section, the Modified Moore-Penrose method by Mussa-Ivaldi and Hogan [5] is executed for a three dimensional Cartesian grid of desired foot position inside the workspace of the leg with the initial condition set to $q(0) = q_{mid_point}$. The joint angles corresponding to each foot position in the grid are

determined. These foot positions and joint angles are then used as training data for a neural network.

The neural network consists of three layers: input, hidden and output. For the front leg, the inputs are x, y, z foot position, the hidden layer consists of 15 neurons and the outputs are the five joint angles ($\alpha, \beta, \gamma, cf$ and ft).

The output of the hidden layer, $y_j(t)$ is

$$y_j(t) = \text{sigmoid}\left(\sum_i w_{j,i}(t)x_i(t) + b_j\right),$$

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (19)$$

where $i = 1, 2, 3$ and $j = 1, 2, 3, \dots, 15$

where $x_i(t)$ is the foot position, i.e. $\{x, y, z\}^T$. The $w_{j,i}(t)$ is the weight value and $b_j(t)$ is the bias. And the weights and bias between the input and the hidden layer can be expressed in matrix form as follows:

$$w_{j,i} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ \vdots & \vdots & \vdots \\ w_{14,1} & w_{14,2} & w_{14,3} \\ w_{15,1} & w_{15,2} & w_{15,3} \end{bmatrix}_{[15 \times 3]}, \quad b_j = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{14} \\ b_{15} \end{bmatrix}_{[15 \times 1]} \quad (20)$$

The outputs of the neural network are the joint angles $z_k(t)$ which can be expressed as

$$z_k(t) = \sum_j w_{k,j}(t)y_j(t) + b_k$$

where $j = 1, 2, 3, \dots, 15$ and $k = 1, 2, \dots, 5$ (21)

The matrix of weights and bias between the hidden layer and output layer is expressed as follows

$$w_{k,j} = \begin{bmatrix} w_{1,1} & w_{1,2} & & w_{1,14} & w_{1,15} \\ w_{2,1} & w_{2,2} & & w_{2,14} & w_{2,15} \\ w_{3,1} & w_{3,2} & \cdots & w_{3,14} & w_{3,15} \\ w_{4,1} & w_{4,2} & & w_{4,14} & w_{4,15} \\ w_{5,1} & w_{5,2} & & w_{4,14} & w_{5,15} \end{bmatrix}_{[5 \times 15]}, \quad b_k = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}_{[5 \times 1]} \quad (22)$$

The neural networks for the middle and rear legs are the same except that their hidden and output layers have fewer neurons. The middle leg hidden layer has 12 neurons and the rear leg hidden layer has 9 neurons. Figure 25 shows the structure of the neural networks for the front, middle and rear legs.

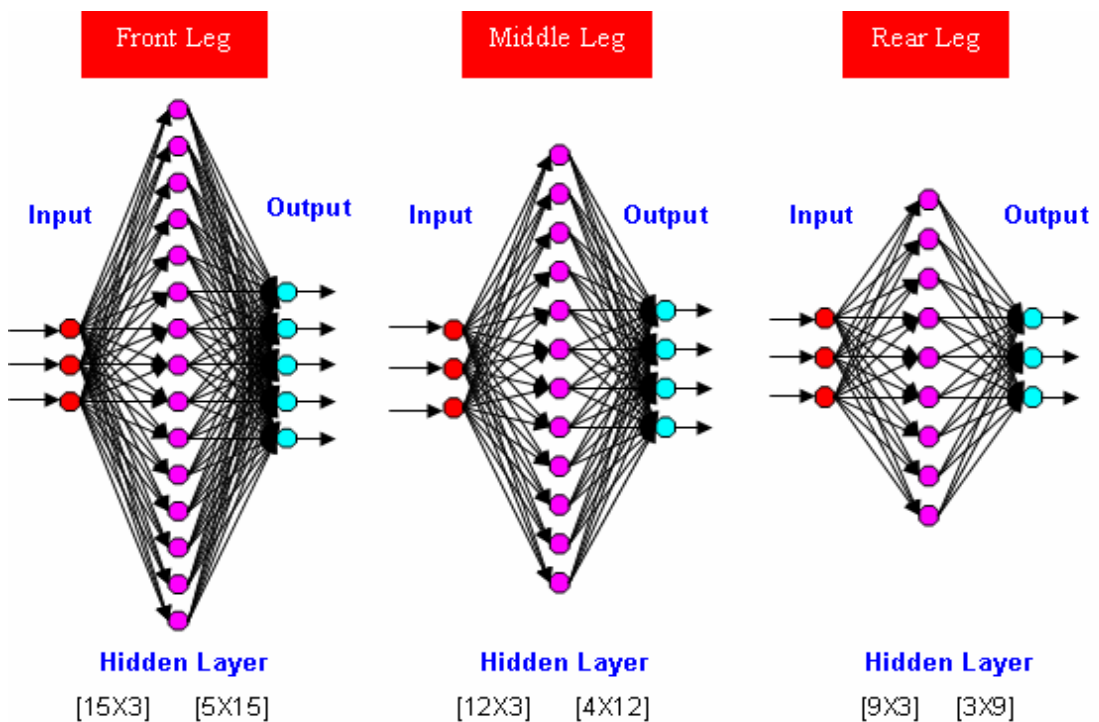


Figure 25 : Neural network for finding joint angles given foot position. The input is the desired foot position and output is the set of joint angles which correspond to desired foot position.

3.4 Training of Neural Network

The goal is for the neural networks to produce appropriate joint angles given foot position input. The neural network is trained with a set of foot positions and the joint angles which correspond to those foot positions. I used MATLAB to train the neural networks. The desired foot position and corresponding joint angles which are found by MMP method are used as the inputs and the targets. The single layer feed forward neural network is trained by back propagation. Weights and biases are found using the NNWFF() function which creates a feed-forward back propagation network. The weight and bias matrices between the input and hidden layer for the front leg are

$$w_{j,i} = \begin{bmatrix} 0.0191531248 & 0.1812573022 & 0.1506997222 \\ 0.4213675109 & -0.3433307116 & 0.5740981579 \\ -0.1738398424 & -0.0677412272 & 0.1369817528 \\ 39.0959498202 & 17.8065524723 & 7.7078348285 \\ -0.1918165108 & 0.0397124721 & -0.0566994067 \\ 0.5471941240 & -0.1090232565 & 0.2010124364 \\ 0.4971912413 & -0.2530875373 & 0.3903729057 \\ 0.2255384656 & 0.0076796918 & -0.3700381389 \\ 0.0735741751 & 0.0691201937 & 0.1190972380 \\ -0.1916797962 & 0.4434734761 & -0.1347711324 \\ -0.0183602394 & -0.1589576868 & 0.2172305175 \\ -26.5450746185 & -18.7458982112 & 0.3548482683 \\ 13.4794550337 & -21.0725582829 & 14.4638206143 \\ -0.2781166490 & -0.0945572834 & -0.1380472145 \\ 5.4146424615 & -11.2951004198 & 21.6380655619 \end{bmatrix}, \quad b_j = \begin{bmatrix} 3.6420985186 \\ 2.4233002818 \\ -0.2092866132 \\ 45.4958986433 \\ 2.0253665709 \\ 0.7208354224 \\ 1.2726963759 \\ -5.4433904141 \\ 1.2660252420 \\ 4.0806947844 \\ 1.2582814890 \\ 7.4616541092 \\ -7.9701114227 \\ -3.2523418259 \\ 37.7036480996 \end{bmatrix} \quad (23)$$

The weight and bias matrices between the hidden and output layers for the front leg are

$$\begin{aligned}
w_{k,j}^T &= \begin{bmatrix} 4.4281563986 & 128.3937259875 & -18.9536660821 & 148.6509683125 & -284.5157981737 \\ -32.5143963756 & 108.8269269710 & 131.4617695691 & 0.0568326467 & -70.0557488265 \\ -0.4943619299 & -149.6373805495 & -38.9226918286 & -101.7412637777 & 12.5021477334 \\ 0.0222170338 & 0.2194262816 & 0.5895882410 & 0.1699702762 & -0.5772397643 \\ 2.9555747505 & 155.9116952748 & -21.5989269524 & 219.1270602047 & 45.6661239775 \\ -6.1040517786 & 89.8595283870 & 106.0161835042 & -17.6664759408 & -41.9998241663 \\ 28.3083213133 & -168.3181752531 & -195.5001115079 & 16.5950805758 & 97.5868228945 \\ -19.1622274634 & 67.3210358318 & -14.4927674786 & 26.9825442452 & 46.7869905196 \\ 16.4285987006 & -71.7601280738 & 63.7396703150 & 69.0866337423 & 353.5832876960 \\ 1.0742988830 & 39.4860514457 & 0.2031571784 & 16.1706820777 & 14.5368384837 \\ -52.1822003913 & 173.8784512835 & -94.8403510826 & 156.7077361155 & 42.2495258279 \\ 0.0483859479 & -0.2836864563 & -0.2478133488 & -0.5797355512 & 0.4645658697 \\ 17.6387598033 & -17.8775836094 & 86.5902287117 & -9.8545504427 & -61.8087772334 \\ -20.4220974828 & 126.3349317186 & 80.6815840457 & 9.6914668441 & 33.6602536433 \\ -0.1758034916 & 0.2351785452 & 0.0425342598 & 0.2056197575 & -0.1088589600 \end{bmatrix} \\
b_k &= \begin{bmatrix} 77.3920495351 \\ -355.5388987601 \\ -53.7128924430 \\ -376.5502822318 \\ 7.3027727072 \end{bmatrix}
\end{aligned} \tag{24}$$

We can find joint angles which correspond to a desired foot position using Eqn.(19) - (22) using the weight and bias matrices from training. Similarly, the weight and bias matrices were found for the middle and rear legs.

The weight and bias matrices between the input and hidden layer for the middle leg are

$$\begin{aligned}
w_{j,i} &= \begin{bmatrix} 0.1084754428 & -0.0963714563 & 0.0661810834 \\ -0.1482020493 & 0.5429028680 & -0.2114539833 \\ 0.1300903024 & 0.5659834031 & -0.2591534069 \\ -7.8920235453 & 0.9689307473 & -7.3475869357 \\ 0.2413001255 & -0.2398468051 & -0.0547363075 \\ 0.1319309304 & -0.3110746156 & -0.1528990671 \\ -0.1847760281 & 0.2436848518 & 0.0786250850 \\ 0.1175365769 & -0.1270690793 & 0.0282677223 \\ 0.0496455521 & 0.0800307093 & -0.1465387549 \\ 0.2812474518 & 0.2659073943 & 0.0105529495 \\ 3.2095032933 & 10.1019141514 & 0.6313279241 \\ -0.0960443066 & 0.0170826326 & -0.1328746266 \end{bmatrix}, \quad b_j = \begin{bmatrix} 0.0287240145 \\ 10.7142633976 \\ 1.0679969510 \\ 17.0607444122 \\ -2.2357155119 \\ -5.0646394410 \\ 2.9980986204 \\ -0.8700297396 \\ 1.1066587111 \\ 3.0365482685 \\ 24.0218024777 \\ -2.2630152590 \end{bmatrix}
\end{aligned} \tag{25}$$

The weight and bias matrices between the hidden and output layers for middle leg are

$$w_{k,j}^T = \begin{bmatrix} 678.1624865189 & 204.5652078164 & 1510.1034433687 & 761.3529854976 \\ -124.2603430918 & 788.1843416385 & -31.7756254461 & 214.1246113164 \\ 12.9463930988 & 18.7036744192 & 13.3626850298 & 9.0419458943 \\ 91.2322927904 & 249.3214318117 & 145.9655123471 & 366.9446868692 \\ 23.9076702654 & -429.4625624010 & -151.1366900338 & -459.0412141673 \\ -74.6721527518 & -467.1055738632 & -233.5633920066 & -328.9513950864 \\ 20.5302042636 & -1187.2120820537 & -556.6729617009 & -1116.7753283034 \\ 667.4782145157 & -637.2108077168 & -1526.5202508860 & -742.3439209984 \\ 67.3144837982 & -97.1936703344 & 158.9776028329 & -258.0606068824 \\ 18.6765720804 & -36.7154417062 & -51.9954502307 & 26.4882957392 \\ 3.4477134346 & -11.5254983421 & 8.4491876094 & -5.6609207935 \\ -104.9227007384 & 15.8441913139 & 118.6796024558 & 549.4758663793 \end{bmatrix} \quad (26)$$

$$b_k = \begin{bmatrix} 101.8293586338 \\ 239.9834917204 \\ 141.2292617188 \\ 378.3440690353 \end{bmatrix}$$

And the weight and bias matrices between the input and hidden layer for the rear leg are

$$w_{j,i} = \begin{bmatrix} -0.7298375537 & 0.3755709901 & -0.2746582843 \\ 0.1590709807 & 0.0362103116 & -0.2920021082 \\ -0.0817101441 & 0.2125335927 & -0.5394249404 \\ 0.0147228838 & -0.1880105601 & 0.0362337944 \\ 0.0973918831 & -0.1252106407 & 0.0634471429 \\ 0.1768067678 & -0.0918212691 & 0.1235758050 \\ 0.1749210757 & 0.3881618052 & -0.0055489079 \\ 0.2549806405 & -0.2651504029 & 0.3507232485 \\ 0.1110419310 & -0.0107431873 & -0.1027811612 \end{bmatrix}, \quad b_j = \begin{bmatrix} 1.9238098461 \\ -3.9909807801 \\ 2.1318744053 \\ -0.7003214852 \\ 1.4822221395 \\ 3.6504025226 \\ 8.5709893712 \\ -1.3932788056 \\ -1.3085335316 \end{bmatrix} \quad (27)$$

The weight and bias matrices between the hidden and output layers for rear leg are

$$w_{k,j}^T = \begin{bmatrix} 413.1239250826 & -96.2482724426 & 223.6683103497 \\ 390.4367534226 & -210.0175329013 & 180.7365068827 \\ -808.2997067257 & 33.9631128615 & -246.7014857910 \\ -20.8442768821 & -383.1375147083 & 235.3313570872 \\ 33.8420987186 & 809.5920245056 & -181.2504049857 \\ 21.5595426088 & -257.6290895860 & -67.7473174191 \\ 2.1879169906 & 89.0354647803 & -138.1499095891 \\ -337.6137039453 & -1.5234851755 & -61.7340232559 \\ -457.4830638232 & -163.8331583948 & -54.2086517721 \end{bmatrix} \quad (28)$$

$$b_k = \begin{bmatrix} 413.5613584671 \\ -116.7609384072 \\ 192.2137959537 \end{bmatrix}$$

The trained neural network produces a mapping between desired foot position and joint angles. The resulting joint angles should reliably produce the desired foot position. Before applying the neural networks to the robot I tested them and those results are described in the next section.

3.5 Validating Neural Networks

Before applying the trained neural networks to solve the inverse kinematics problem for the robot's legs, I tested their performance. To check the reliability of the neural networks, a simple desired path is designed. Simple paths are designed in the workspace of each leg. The paths are discretized into a set of desired foot positions(x, y and z), which are then plugged into the trained neural networks. The output joint angles are plugged into the forward kinematics equations to determine the resulting foot positions. These foot positions are compared with the desired (input) foot positions. Figure 25 illustrates the procedure used to check the reliability of neural networks.

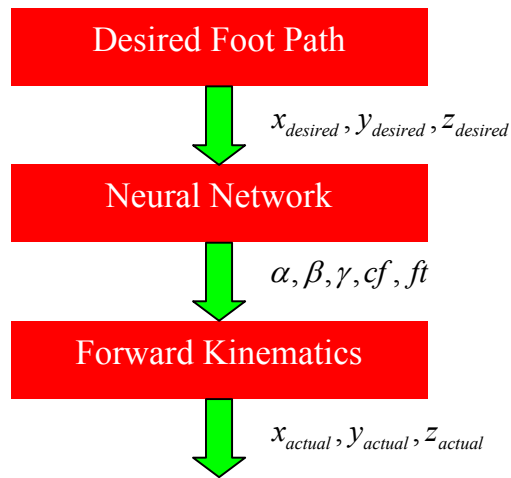


Figure 26 : Procedure to validate the neural network

3.5.1 Design of the Desired Foot Path

Simple foot paths are designed for the front, middle and rear legs to be used as test cases to verify that the trained neural networks successfully solve the inverse kinematics problems. Figure 27 shows a foot path for the front leg. The front leg steps forward in front of the body, which is the case for cockroach walking. During swing, the path follows the parabolic equation $z = -(1/4) * x^2 + 2.0 * x - 8.8125$. The desired path is designed in the plane where $y = -10$ (in the case of right front leg). The z distance from the center of mass (COM) to the foot position of front, middle and rear legs during stance are -12.5 inches. The length from the COM to the BC joint of the front leg is 3.6875 inches in the z direction.

Figure 28 shows the foot path for the middle leg. The foot moves behind the leg's BC joint. The x position ranges from -1 to -9 inches. High stepping in the swing phase is

not necessary for the middle leg. During swing, its path follows the parabolic equation $z = -(1/8)*x^2 - 5./4.*x - 73/8$ and the y position of leg is constant at $y=-12$ inch.

Figure 29 shows the foot path for the rear leg. The main function of the rear leg is to push the robot forward. The x position ranges from -10 to -14. The parabolic equation for the rear leg's swing is $z = -(1/2)*x^2 - 12*x - 78$ and the rear leg is moving in the plane defined by $y=-10$ inch. Due to the limited workspace of the rear leg, the x range of the rear leg is shorter than that of middle and rear leg.

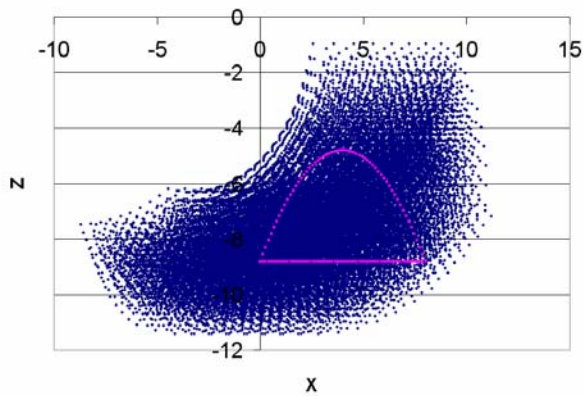


Figure 27 : Foot path for right front leg

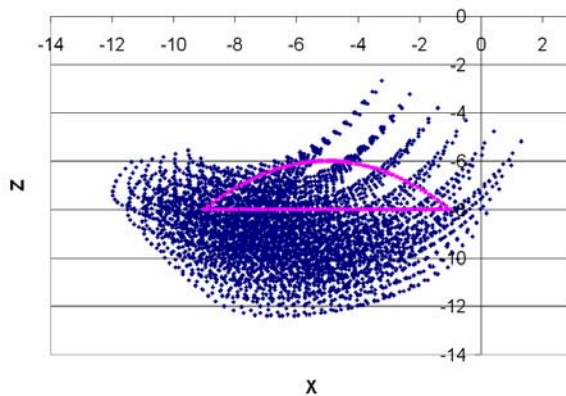


Figure 28 : Foot path for right middle leg

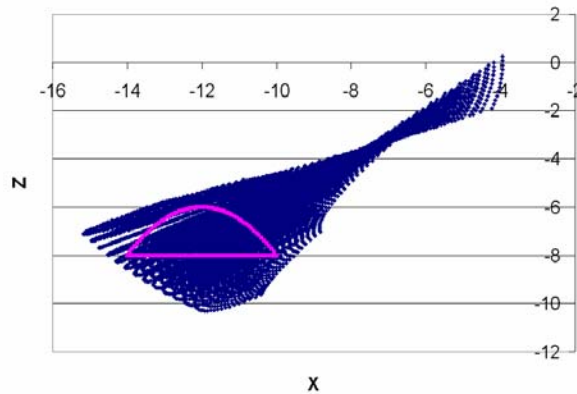


Figure 29 : Foot path for the right rear leg.

3.5.2 Comparison of Desired and Actual Paths using Neural Networks

Figure 30 shows a comparison between the desired path and the actual path for right front leg. The position errors of x, y and z directions for right front leg are shown in Figure 30. The position errors are between $-0.2 \sim 0.2$ inches. Figure 31 shows a comparison between the desired path and the actual path for the right middle leg. In Figure 31, the position errors in x, y and z directions for right middle leg are shown. The range of errors is from -0.05 to 0.05 inches. Figure 32 shows comparisons between the desired path and the actual path for the right rear leg. The position error in x, y and z directions are shown. The position error in the x direction for rear leg is the biggest, but its range is between -0.05 and 0.1 . The actual path follows the desired path except at the extreme positions. The neural network produces a good mapping between foot position and joint angles.

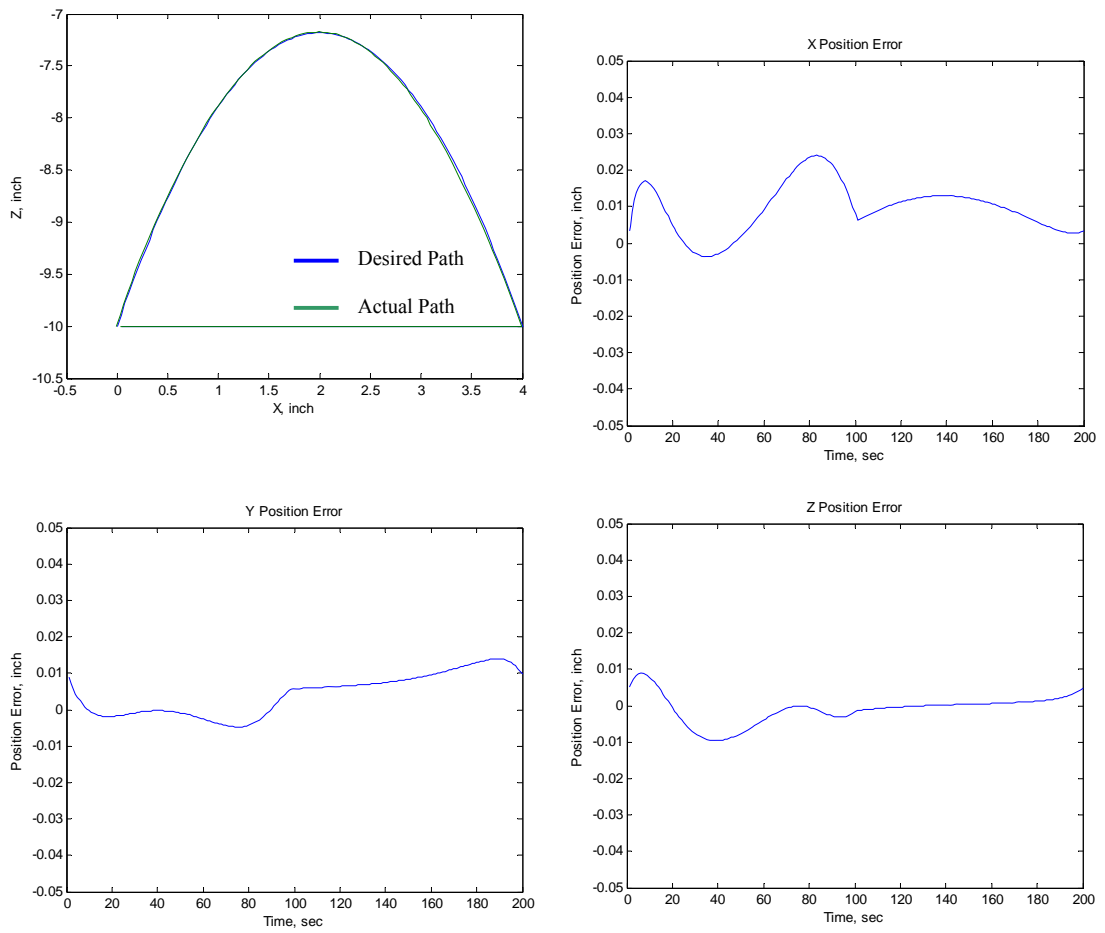


Figure 30 : Desired vs. actual foot path of right front leg and position errors along x, y and z axis vs. time.

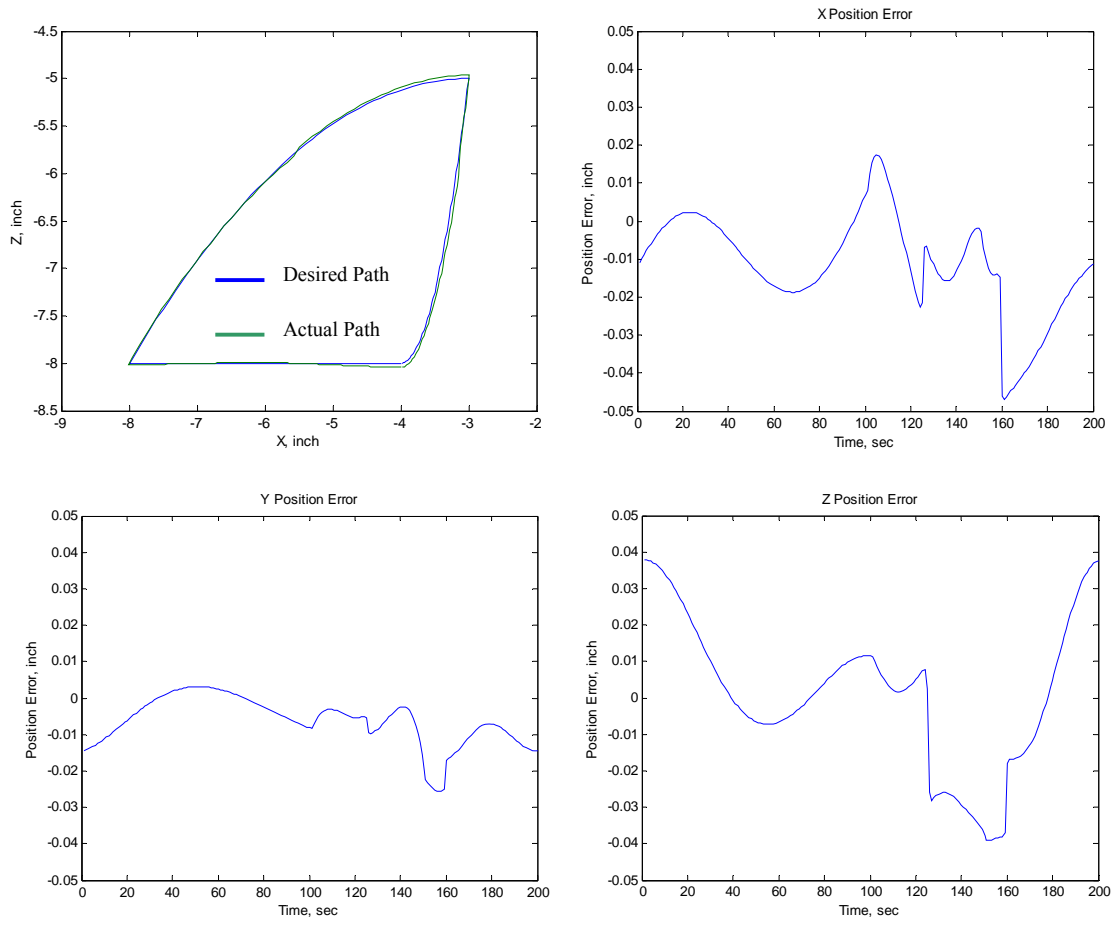


Figure 31 : Desired vs. actual foot path of right middle leg and foot position errors along x, y and z axis vs. time

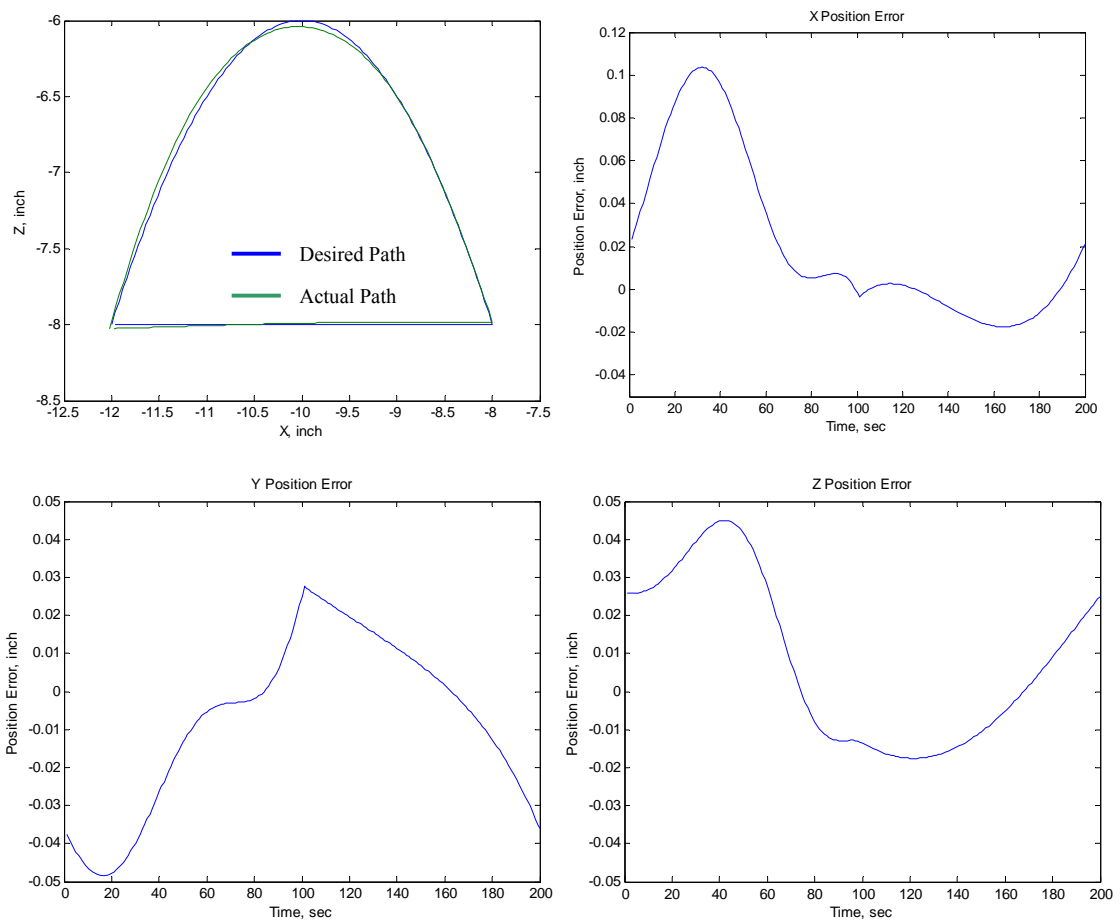


Figure 32 : Desired vs. actual foot path of right rear leg and foot position errors along x, y and z axis vs. time

3.5.3 Discussion

The inverse kinematics problem for each of the legs on Robot V was solved in this chapter. Before finding a mapping between foot position and joint angles, I used

forward kinematics to determine the workspace of each leg because there is no sense in trying to find an inverse kinematic solution for an unreachable foot position.

The Modified Moore-Penrose (MMP) method was introduced to solve the inverse kinematics problem for the legs because the front and middle legs are kinematically redundant. Solving this problem is computationally intensive, so it was performed off line. A three dimensional Cartesian grid was defined in each leg's workspace and the MMP method was used to solve the inverse kinematics problem for each of those points. These solutions were used as training data for neural networks that could then solve the inverse kinematics problems online. The trained neural networks produced accurate mapping between foot positions and joint angles. The neural networks can be used to solve the inverse kinematics problems for Robot V.

References

- 1 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.
- 2 Nelson, Gabriel. Learning about control of legged locomotion using a hexapod robot with compliant pneumatic actuators, Ph. D dissertation. May 2002.
- 3 Mussa-Ivaldi, F.A., Hogan, N. Integrable Solutions of Kinematic Redundancy via Impedance Control. *Int. J. of Robotics Research*, Vol. 10, No. 5, pp. 481-491, October, 1991.
- 4 Mussa-Ivaldi, F. A. and Morasso, P. Patterns of interarticulator phasing and their relation to linguistic structure. *Biological Cybernetics*, Vol. 60, pp. 1-16, 1988.

Chapter 4

Coordination of Legs

The robot controller must coordinate the joints of all the legs to move the robot's body in a desired manner. In the previous chapter, the intra-leg control problem was solved to coordinate the joints of a leg and cause a desired foot motion. A gait controller is described in this chapter to coordinate the legs. The gait controller switches each leg to and from stance and swing phases. A leg cycle consists of the stance phase, in which the leg supports the body, and the swing phase, in which the leg lifts and moves forward to start a new stance phase. Cruse reviewed a network [1] that can generate insect gaits and Dean described the network in more detail [2]. In this chapter, a fixed tripod gait controller and the Cruse network are explained and used to coordinate the legs of Robot V.

4.1 Pre-Planned Tripod Gait

An alternating tripod gait is observed when the cockroach walks or runs rapidly. In this gait, the right front and right rear legs move in synchrony with the left middle leg to form a tripod, which alternates with a tripod formed by the other three legs. These leg movements can be easily programmed in software to form a fixed "pre-planned tripod

gait.” Figure 33 shows a pattern of footfalls for a tripod gait resulting from a pre-planned gait. Each leg completes a cycle every 1.2 sec. The tripods alternate every 0.6 sec. While the right front, right rear and left middle legs support the body, the right middle, left front and left rear legs swing in unison. The pre-planned gait controller successfully produced a tripod gait in which each foot moves through the desired paths depicted in Figure 34 (a), (b) and (c). Even though the stride length in the rear legs is shorter than that in the front and middle legs, the stance and swing of all legs is done in 1.2 sec.

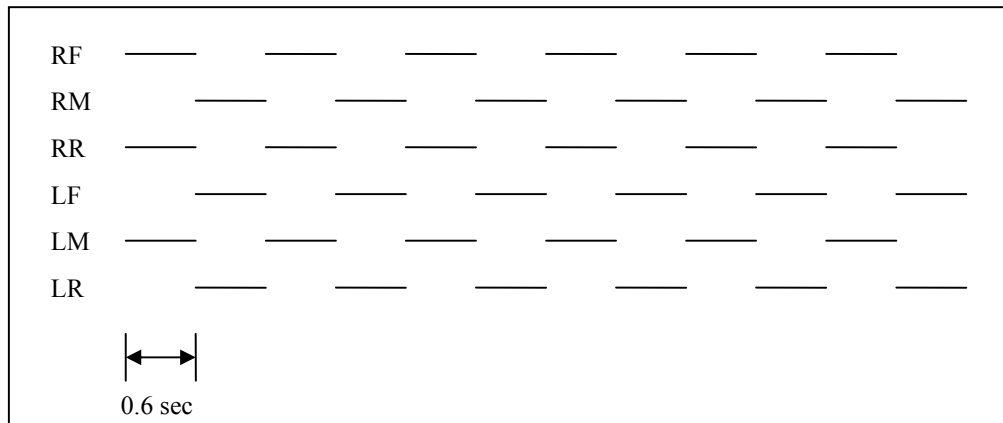


Figure 33 : Tripod gait footfalls produced by a fixed gait generator. RF, RM, RR = right front, middle and rear, respectively. LF, LM, LR = left front, middle and rear, respectively. The bold lines represent swing phase.

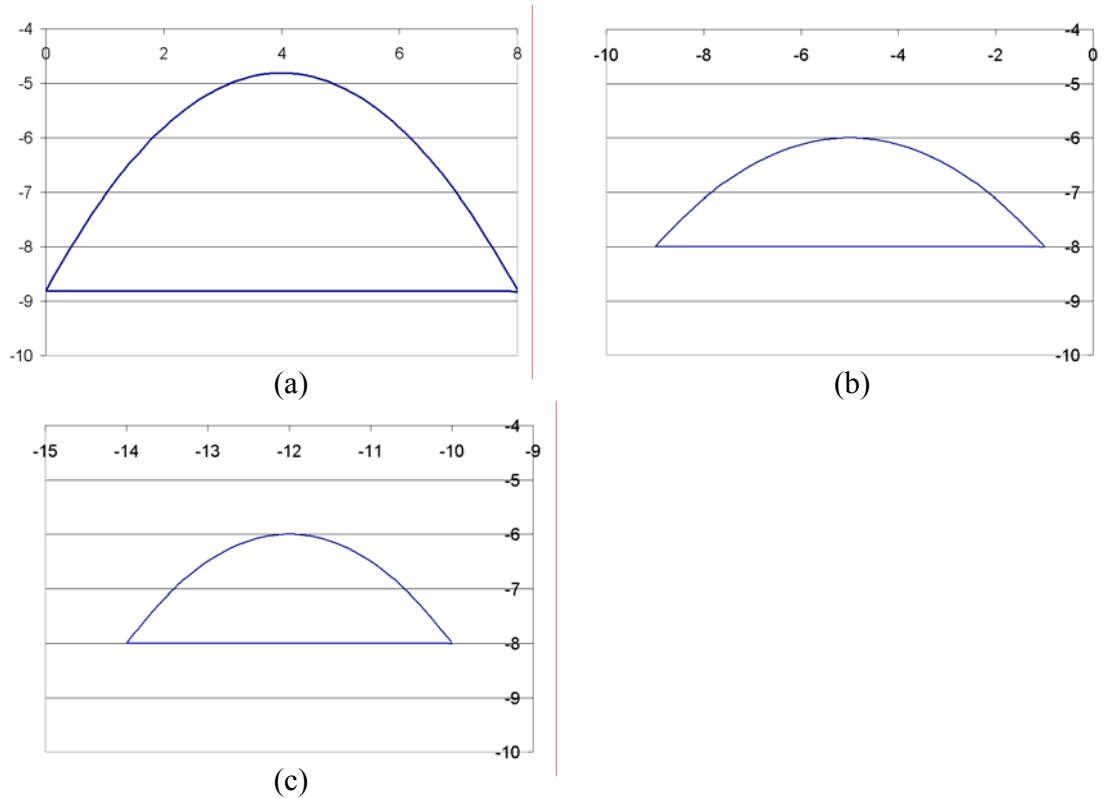


Figure 34 : Desired Paths for Tripod Walking. (a), (b) and (c) show the desired path for front leg, middle and rear legs, respectively. The foot position is expressed with respect to the intersection point of Gamma and Beta joint axes for each leg.

4.2 Cruse Control for Gait Generation

Fixed gaits are not appropriate for irregular terrain or even for acceleration on regular terrain. The gait should adapt to the speed of the robot and its terrain. Dante II [3] used a fixed gait which was partially responsible for it rolling over during its ascent from a volcano. Cruse [1] and Dean [2] published a network gait controller that was based upon the behavior of the stick insect *Carausius morosus*. It has been applied to a number

of robots including Robot I [4], Robot II [5], the Lauron series [6], and the Tarry series [7].

Robot I and Robot II used three of the six mechanisms that Cruse and Dean described in the gait network. A leg moves from its anterior extreme position (AEP) to its posterior extreme positions (PEP) in stance and then swings from the PEP back to the AEP. The AEP position is fixed, but the PEP is adjusted forward or backward by influences sent by adjacent legs. The influences are regulated by the three mechanisms shown in Figure 42. Mechanism 1 is a negative step function which moves the PEP of adjacent legs backward to inhibit them from swinging while the sending leg is in its swing phase. It begins at the start of swing of the sending leg and ends 60ms after the sending leg begins stance. Mechanism 2 is an impulse that shifts the PEP of the receiving leg forward. It encourages receiving legs to initiate swing. Mechanism 2 begins 60ms after the sending leg's stance phase begins and ends 60ms later. Mechanism 5 is a ramp function. During the stance phase of a sending leg, Mechanism 5 shifts the PEP of the receiving legs forward.

Figure 36 shows the Cruse gait network, which shows how the three mechanisms described above influence particular legs. L1, L2 and L3 denote the left front, middle and rear legs, respectively. Similarly, R1, R2 and R3 denote the right front, middle and rear legs, respectively. The arrows represent the direction of the influences from a sending leg to a receiving leg. The ipsilateral (same side of body) legs in swing send Mechanisms 1 and 2 from rear legs forward to middle legs and middle legs forward to front legs. Mechanism 5 is sent rearward by ipsilateral sending legs while they are in stance. The contralateral (opposite side of body) front and middle legs influence each

other with Mechanisms 2 and 5. The rear legs send Mechanisms 1, 2, and 5 to their contralateral mate.

Influences are summed at each leg and the result moves the PEP of that leg forward or rearward at each time step in the control cycle. When a leg in stance reaches its PEP it switches to swing. The stance speed is changed by the user or another part of the control system. However, the swing speed of a leg must be constant and independent of any other control system for this network to successfully produce stable gaits.

Each network connection has a corresponding weight or scaling factor that must be determined to produce a stable gait controller. Through trial and error, the scaling factors shown in Table 4 for all network connections were found.

	IPSILATERAL	CONTRALATERAL
Mechanism 1	1.0325	0.3
Mechanism 2	2.0	0.1
Mechanism 5	7.25	3.08

Table 7 : Scaling Factors for Cruse Controller

Stable tripod gaits are produced using these scaling factors and Figure 37 shows footfalls for one such gait. After a transient, the Cruse controller produces a stable tripod gait. When the commanded speed is changed, the gait also changes.

The Cruse controller switches a leg to and from stance and swing. It does not define the path of the foot in stance or swing. Combining the leg controller described in the previous chapter, particular foot paths within the workspace of the legs, and the Cruse controller can result in the robot moving its legs in insect-like gaits. The Cruse controller determines the AEP and PEP as the extreme x values of foot position for each leg. When

the foot meets the PEP, the foot starts the swing and at the middle of AEP and PEP, the swing path reaches the maximum height.

The elevation of a foot in swing is defined by a parabolic equation. The basic parabolic equation of motion is defined by Kingsley [8]

$$C(X - PEP)(AEP - X) + \text{ground} = Z \quad (29)$$

Where X is the current foot position, Z is the vertical position and the ground means the elevation below the foot position during stance. C ensures that the foot reaches the maximum heights at the middle of AEP and PEP.

$$C\left(\frac{AEP + PEP}{2} - PEP\right)\left(AEP - \frac{AEP + PEP}{2}\right) + \text{ground} = Z_{\max} \quad (30)$$

$$C = 4 \frac{(Z_{\max} - \text{ground})}{(AEP - PEP)^2}$$

So the final parabolic equation for the path is

$$Z = 4 \frac{(Z_{\max} - \text{ground})}{(AEP - PEP)^2} (X - PEP)(AEP - X) + \text{ground} \quad (31)$$

Once the Z_{\max} and ground values are defined, the elevation of foot is decided by Eqn.

(31). Figure 38 shows the automatically generated foot paths for all six legs. Each swing path is expressed with respect to the BC joint.

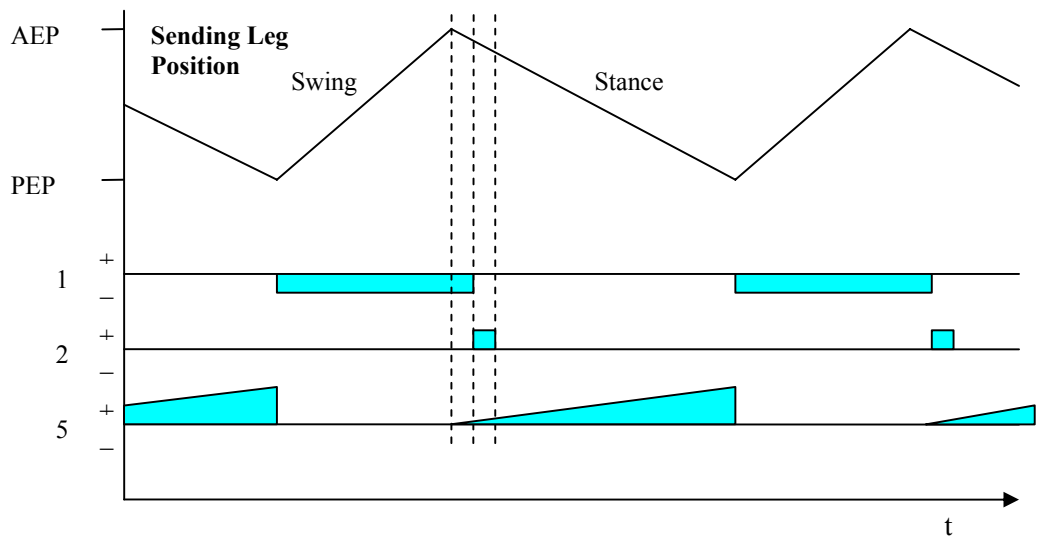


Figure 35 : Three of the Cruse mechanisms. Sending Leg Position versus Time is plotted with Mechanism 1, 2 and 5. AEP is the anterior extreme position and PEP is the posterior extreme position.

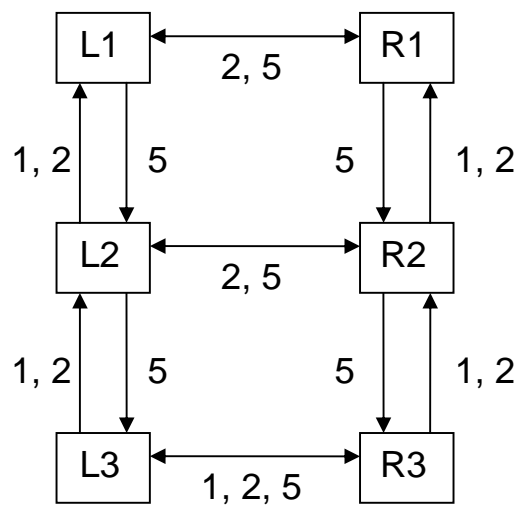


Figure 36 : Cruse gait network using mechanisms (1, 2 and 5). L1, L2, L3 = Left front, middle and rear leg, respectively. R1, R2, R3 = right front, middle and rear leg, respectively.

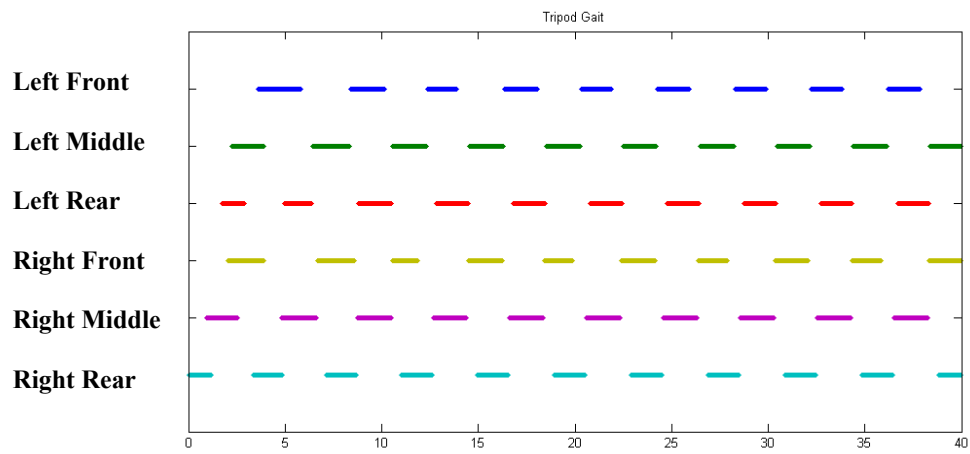


Figure 37 : Generated Gait Pattern by Cruise Controller.

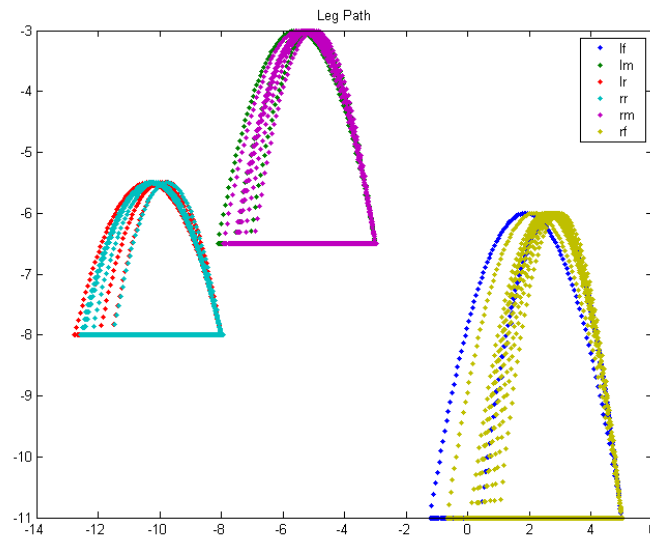


Figure 38 : Automatically Generated Foot Paths for Six Legs in inches

References

- 1 Cruse, H., What Mechanisms Coordinate Leg Movement in Walking Arthropods? *Trends in Neural Science*, 12, pp. 15-21, 1990.
- 2 Dean, J. A model of leg coordination in the stick insect : description of the kinematic model and simulation of normal step pattern. *Bio Cybern.* 1991.
- 3 Wettergreen, D., Thorpe, C. Developing Planning and Reactive Control for a Hexapod Robot. *Proceeding of the 1996 IEEE International Conference on Robotics and Automation (ICRA 1996)*, Vol. 3, pp. 2718-2723. April, 1996.
- 4 Espenschied, K. S., Quinn, R. D., Chiel, H. J. and Beer, R. D. Leg Coordination Mechanisms in Stick Insect Applied to Hexapod Robot Locomotion. *Adaptive Behavior*, Vol. 1, No. 4, pp. 455-468, M.I.T. Press, 1993.
- 5 Beer, R.D., Quinn, R.D., Chiel, H.J., Ritzmann, R.E. Biologically-Inspired Approaches to Robotics. *Communications of the ACM*, Vol. 40, No. 3, March 1997
- 6 Celaya, E., Albarral, J. L. Implementation of a hierarchical walk cotroller for the LAURON III hexapod robot, *Proc. 6th International Conference on Climbing and Walking robots (Clawar 2003)*, pp. 4-9-416. Catania, Italy 2003.
- 7 Frik, M., Guddat, M., Karatas M. and Losch, D. C., A novel approach to autonomous control of walking machines. *Proc. Of the 2nd Int. Conf. on Climbing and Walking Robots (CLAWAR 99)*, Portsmouth, UK. 13-15 September 1999.
- 8 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.

Chapter 5

Dynamic Simulation of Robot V

Control solutions have been described in the previous chapters that were designed to move the feet of Robot V along prescribed trajectories and coordinate its legs into insect gaits. Before these controllers are applied to the robot hardware they should be tested to evaluate their performance. Testing off line reduces the chances that an improperly designed control system will damage the robot. A dynamic simulation of Robot V is developed for this purpose. The model of Robot V is used to test if it can move its feet along the desired foot paths and if it can produce enough actuator force to walk. The simulation has the advantage that whenever a modification to the control scheme or robot hardware is considered the simulated robot can be modified correspondingly and then used to test the changes without changing or damaging the robot. The model robot has the same properties as that of the real Robot V. The robot's body, air muscles, valves and controller are all modeled. The process of modeling a robot's joints and rigid links is well known. However, integrating a robot's mechanical model with models of air muscles and valves has not been done before to our knowledge. This chapter describes such a dynamic model of Robot V.

5.1 Mechanics of Robot V

The kinematic model inputs include leg segment lengths, joint degrees of freedom, and joint ranges of motion. Daniel Kingsley designed and built Robot V [1]. Kingsley supplied me with all of the kinematic data. The front, middle and rear legs are shown in Figure 39,

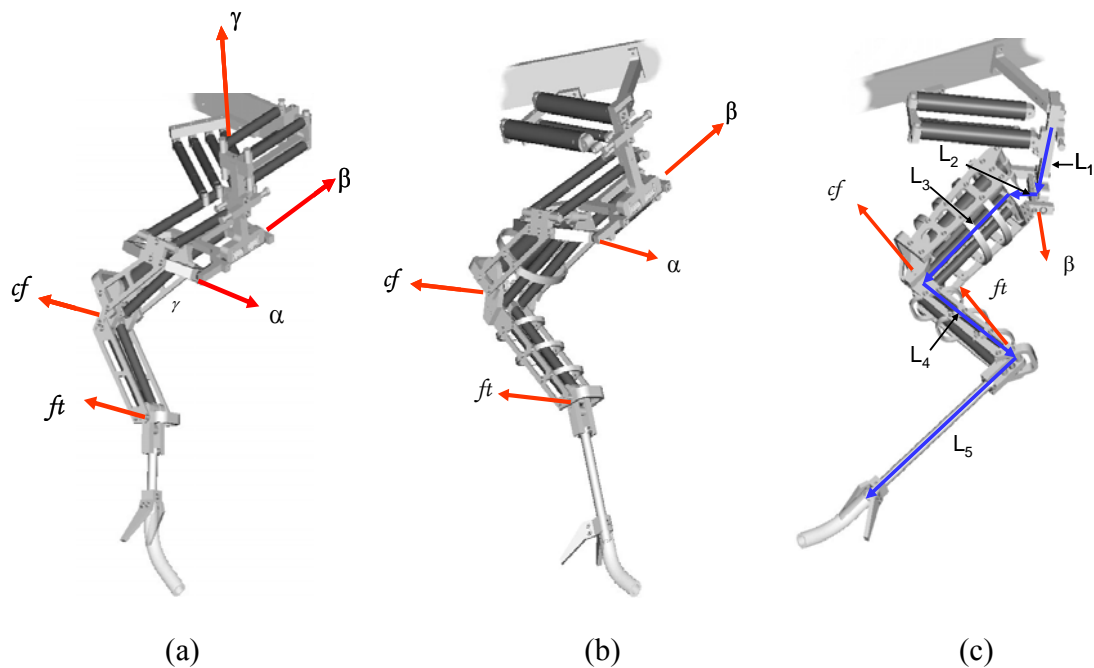

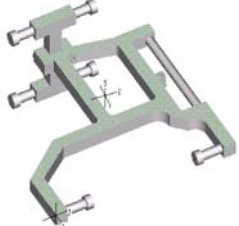

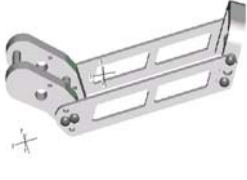



Figure 39 : Leg configuration. (a), (b) and (c) show the front, middle and rear legs respectively.

Kingsley [1] used ProEngineer software to design Robot V and used that software to output mass and inertia properties of all of the leg and body segments. Table 8 shows the mass and inertia values for Robot V. The tabulated mass moment of inertia values are measured at the center of mass of each segment with respect to that segment's coordinate system. When these values are implemented in the simulation, they are converted into the inertial coordinate system. As mentioned in Chapter 4, in the inertial coordinate system

the x axis is pointed in the direction of forward locomotion. The y axis is directed from right to left and the z axis is in the upward direction.

Leg	Joint	Mass (pound)	Inertia (pound * inch ²)	Part Drawing
Front	Gamma	2.27220e-01	Ixx Ixy Ixz 6.227630e-01 0.000000e+00 0.000000e+00 Iyx Iyy Iyz 0.000000e+00 7.774512e-02 5.532458e-02 Izx Izzy Izz 0.000000e+00 5.532458e-02 5.538451e-01	
	Beta	4.4052402e-01	Ixx Ixy Ixz 2.618703e+00 0.000000e+00 0.000000e+00 Iyx Iyy Iyz 0.000000e+00 1.651737e+00 6.267821e-02 Izx Izzy Izz 0.000000e+00 6.267821e-02 1.196265e+00	
	Coxa	5.5715623e-01	Ixx Ixy Ixz 1.860429e+00 1.724261e-02 -6.916335e-01 Iyx Iyy Iyz 1.724261e-02 2.098623e+00 -4.333645e-02 Izx Izzy Izz -6.916335e-01 -4.333645e-02 1.463382e+00	
	Femur	4.4036821e-01	Ixx Ixy Ixz 2.954373e+00 0.000000e+00 0.000000e+00 Iyx Iyy Iyz 0.000000e+00 2.998603e+00 -3.015959e-01 Izx Izzy Izz 0.000000e+00 -3.015959e-01 4.371915e-01	
	Tibia	2.2963843e-01	Ixx Ixy Ixz 6.335289e-02 1.355828e-03 -4.538435e-02 Iyx Iyy Iyz 1.355828e-03 8.607386e-01 0.000000e+00 Izx Izzy Izz -4.538435e-02 0.000000e+00 8.940065e-01	



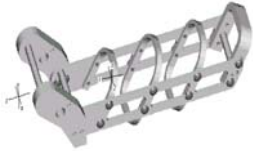




Middle	Beta	4.9607907e-01	Ixx Ixy Ixz 3.369076e+00 0.000000e+00 0.000000e+00 Iyx Iyy Iyz 0.000000e+00 1.837608e+00 1.963261e-01 Izx Izy Izz 0.000000e+00 1.963261e-01 1.762069e+00	
	Coxa	7.0766341e-01	Ixx Ixy Ixz 4.173249e+00 3.527953e-02 -1.272201e+00 Iyx Iyy Iyz 3.527953e-02 3.752697e+00 -3.673386e-02 Izx Izy Izz -1.272201e+00 -3.673386e-02 2.53096e+00	
	Femur	5.7020337e-01	Ixx Ixy Ixz 4.880589e+00 -2.310586e-04 -3.501823e-03 Iyx Iyy Iyz -2.310586e-04 5.123895e+00 -4.125603e-01 Izx Izy Izz -3.501823e-03 -4.125603e-01 7.741752e-01	
	Tibia	2.5642978e-01	Ixx Ixy Ixz 1.886782e+00 -1.355853e-03 0.000000e+00 Iyx Iyy Iyz -1.355853e-03 1.333535e-01 -2.094625e-01 Izx Izy Izz 0.000000e+00 -2.094625e-01 1.877678e+00	
Rear	Coxa	7.0427385e-01	Ixx Ixy Ixz 2.026711e+00 1.184812e-01 1.752532e-01 Iyx Iyy Iyz 1.184812e-01 5.259318e+00 -2.973040e-02 Izx Izy Izz 1.752532e-01 -2.973040e-02 5.742432e+00	
	Femur	5.7845330e-01	Ixx Ixy Ixz 5.734787e+00 -2.458232e-04 -3.823871e-03 Iyx Iyy Iyz -2.458232e-04 5.982993e+00 -4.468933e-01 Izx Izy Izz -3.823871e-03 -4.468933e-01 7.842661e-01	
	Femur	2.7244938e-01	Ixx Ixy Ixz 4.877462e+00 8.794328e-02 2.341940e-05 Iyx Iyy Iyz 8.794328e-02 6.465154e-02 -1.361265e-03 Izx Izy Izz 2.341940e-05 -1.361265e-03 4.877484e+00	

Table 8 : Mass and Inertia Values of Robot V.

5.2 Braided Pneumatic Actuator

5.2.1 McKibben vs. Festo Actuator

The “McKibben actuator” was patented in 1957 by Gaylord [2] and used by McKibben to actuate orthotic and prosthetic devices. McKibben actuators have been called Rubbertuators [3], [4], Braided Pneumatic Actuators (BPA), and air muscles. This actuator is made of a rubber inner tube covered with a braided mesh with a helical weave. Both ends are clamped onto end plugs to prevent leaks. One end has an inlet hose to enable the rubber tube to be filled with compressed gas. Figure 40 shows a McKibben actuator. When the actuator is inflated it expands circumferentially and contracts lengthwise. (This action depends on the weave. For some weaves the actuator will lengthen instead of contracting). The tensile force depends on gas pressure. McKibbens are naturally compliant. When an antagonistic pair of these actuators is used to power a joint, the joint’s compliance can be varied independently of its motion. The ratios of power to weight and force to weight for McKibbens is higher than for electric motors, air cylinders, and most animal muscle even when the weight of the valves is included.



Figure 40 : McKibben Actuator

McKibben actuators can easily be made in most any shop. They can also be purchased commercially from Shadow Robot Company [5]. The problem with McKibbens is that they fail in fatigue if they are operated at high pressure. The mesh wears the rubber tube, which eventually leaks. For this reason, Shadow recommends that they be used at pressures below 4 bars. However, we wish to run them at higher pressures (6 bar) to increase their force.

A variant of McKibben actuators called Festo air muscle are sold by Festo Inc [6]. Unlike McKibben actuators, the helical woven mesh in the Festo actuators is inside the rubber tube. Their fatigue life is reported to be in the tens of millions of cycles at high pressures. For this reason they have been implemented in Robot V. However, Festo actuators have a number of disadvantages. First, Festo actuators are made with large, aluminum ends, which increase their weight and reduce their strain capacity. Second, because their tubes are thick, the initial inflation pressure is higher than McKibbens. Third, the thick tubes also result in them being stiff when they are not inflated.

Some of the disadvantages of Festo actuators have been solved. Kingsley solved the problem of the Festo ends by replacing them with small, plastic ends [1]. Figure 41 (a) shows an original Festo air muscle end and (b) shows a modified end [7]. The problem of them being stiff in compression is solved in control software as is discussed in a later chapter.



Figure 41 : Festo actuator. (a) original Festo air muscle end. (b) modified end.

The static force-length properties of a 20mm diameter Festo actuator at different pressures is shown in Figure 42 [8]. Festo actuator is limited in contraction and extension. The contraction ratio, h is $(l_0 - l)/l_0$, where l_0 is the original length of the actuator and l is the length after pressurization. This actuator can stretch up to $h = -3\%$ and contract up to $h = 20\%$. The more the actuator contracts, the less actuator force is produced.

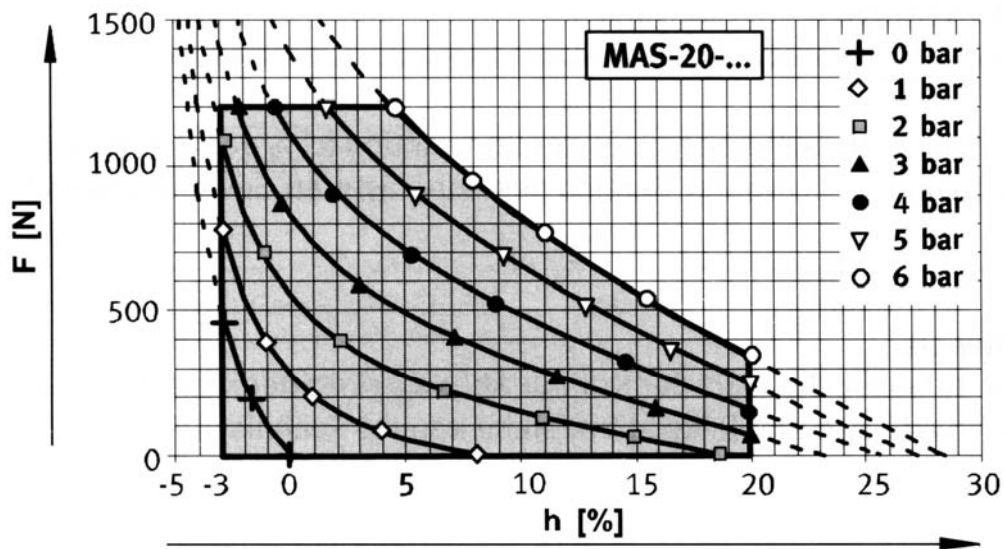


Figure 42 : Force vs. h (contraction ratio). $h = (l_0 - l)/l_0$ where l_0 is the original actuator length. l is the length after contraction. The diameter of the actuator is 20 mm.

5.2.2 Testing of Festo Actuator Properties

Robot V was the first robot in the Biorobotics Lab to use Festos and, in fact, it is one of the first in the world (AirInsect is the first [9]). To control the actuator, we have to know its properties. It was modeled and tested to verify it can produce enough force

before it was used in Robot V. Festo actuators were tested using the rig which is shown in Figure 43. When the actuator is pressurized, it contracts. The linear potentiometer measures the length of the actuator. The A/D board digitizes the potentiometer signal. The relationship between the sampled number and length is shown in Figure 44. A linear curve was fit to this relationship resulting in $y=0.014x + 2.5961$ where x is the sampled potentiometer data and y is the length. The pressure sensor was also calibrated. Figure 45 shows the relationship between the sampled signal and the actuator pressure. The relationship between the sampled pressure data (x) and pressure (y) is $y = 0.3712x - 6.9159$.

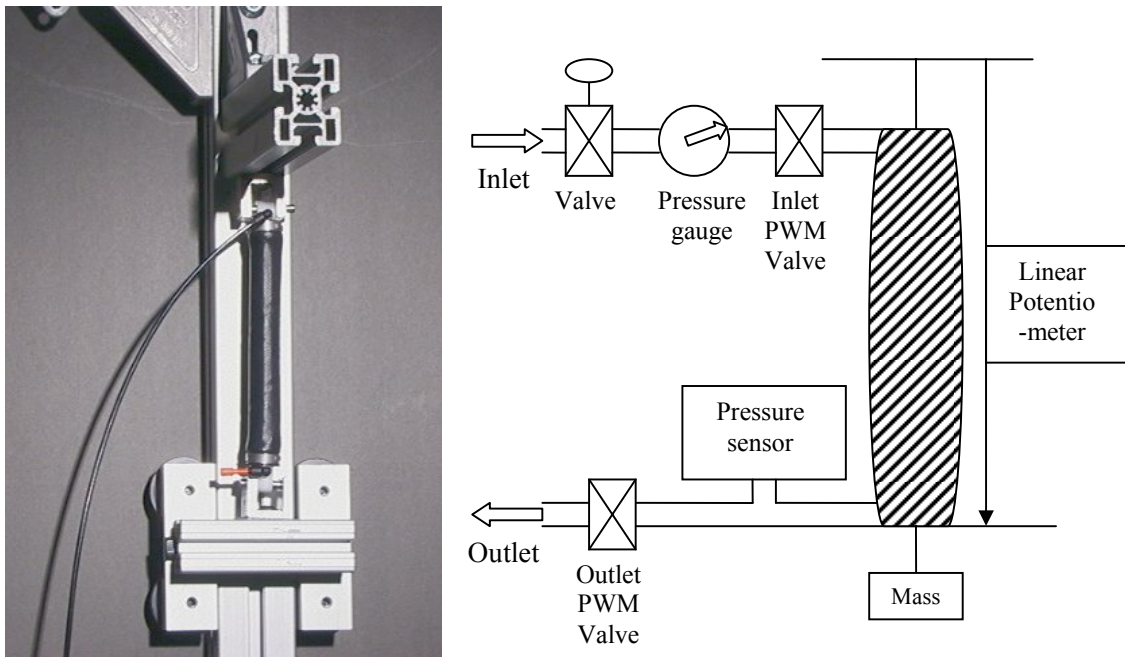


Figure 43 : Actuator Test Rig. The setup is for measurement of the pressure and length of Festo actuators.

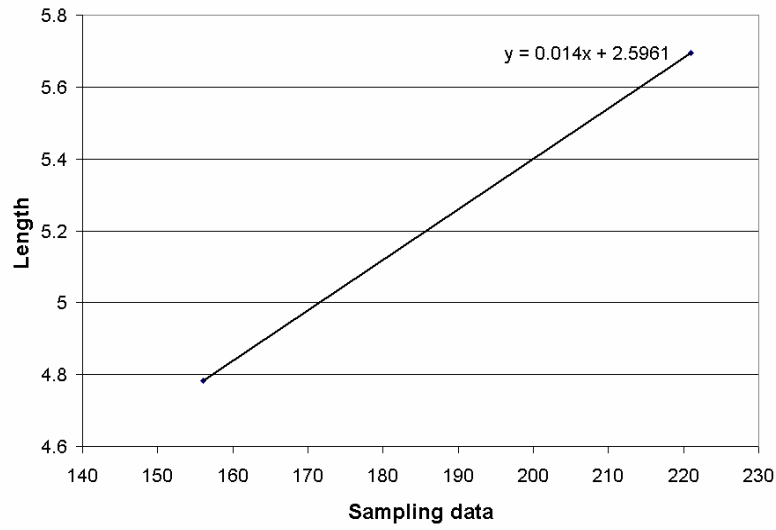


Figure 44 : Length Calibration. The relationship between length and sampled potentiometer data is shown.

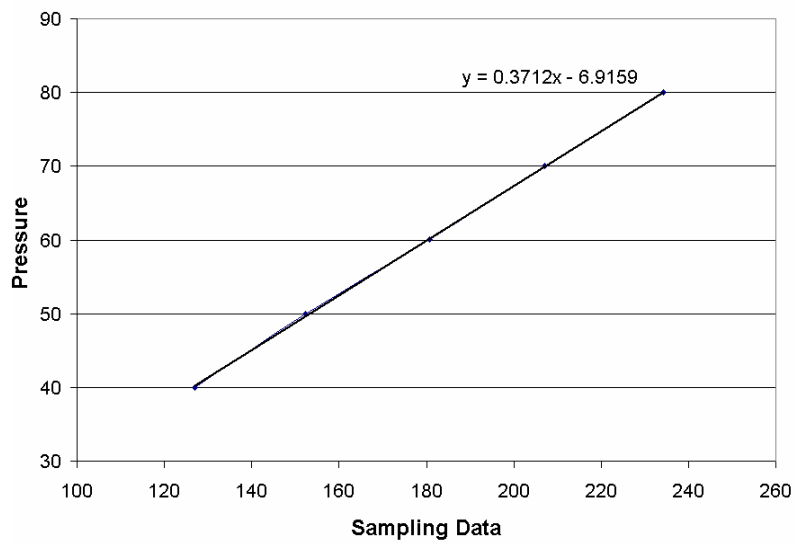


Figure 45 : Pressure Calibration. The relationship between pressure and sampled pressure data is shown.

We used the test rig to measure transient properties of Festo actuators. A 10 lb mass is applied to the end of the test actuator and the pressure regulator valve is open to

set the pressure to a constant value. The Festo actuator is connected to an inlet valve, which is controlled by a PC. The outlet valve is closed. As the inlet valve opens, air suddenly flows into the actuator. As the pressure increases, the length of the actuator decreases. Experimental results showing length vs. time for different pressures is shown in Figure 46.

When the inlet valve is opened suddenly, high pressure air makes the actuator shorten rapidly, but then the slope of the shortening decreases. The Festo actuator is a viscoelastic system. It takes time to stabilize. Modeling of this actuator is important because we have to determine how to control them. The next section describes modeling Festo actuators.

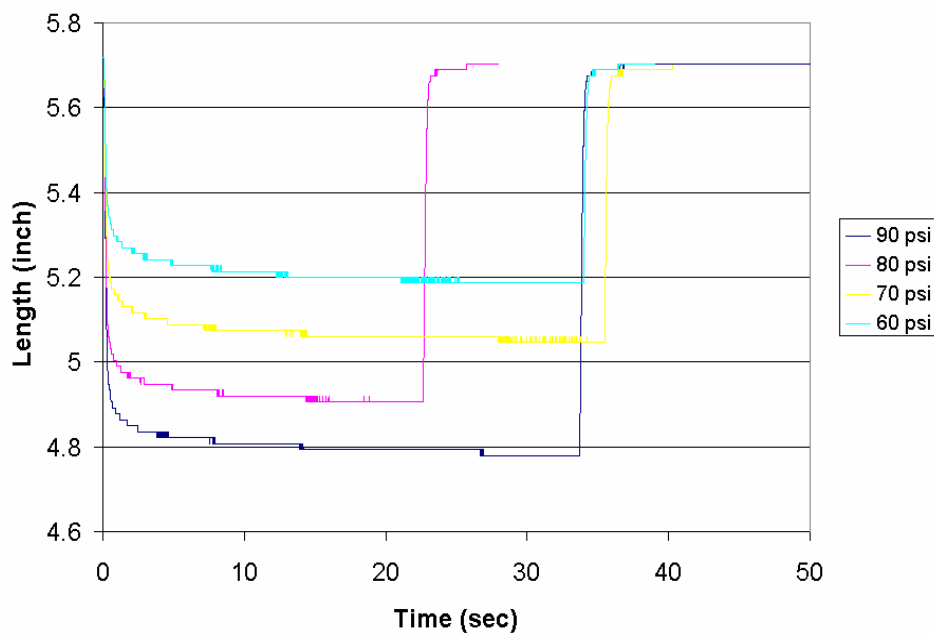


Figure 46 : Festo Experimental Results: Length vs. Time for different pressures

5.2.3 Modeling of Festo Actuators

McKibben actuators have been modeled by others. We assume that McKibben models fit Festo actuators with a few modifications. Hannaford et al. [10], [11] and [12] modeled them using the principle of virtual work. The input work from internal pressure can be expressed by $dW_{in} = PdV$ where P is the absolute gas pressure and dV is the volume change. The output work can be expressed by $dW_{out} = -Fdl$ where F is the actuator force and dl is the actuator length change. According to the virtual work theorem, the input work is supposed to be same as the output work, $dW_{in} = dW_{out}$.

$$PdV = -Fdl \quad (32)$$

$$F = -P \frac{dV}{dl} \quad (33)$$

To determine dV/dl , the middle portion of the actuator is assumed to be a cylinder. This assumption is even more applicable to Festo actuators than McKibbens. The length and diameter of the cylinder can be expressed in terms of the fiber length b and the weave angle θ as

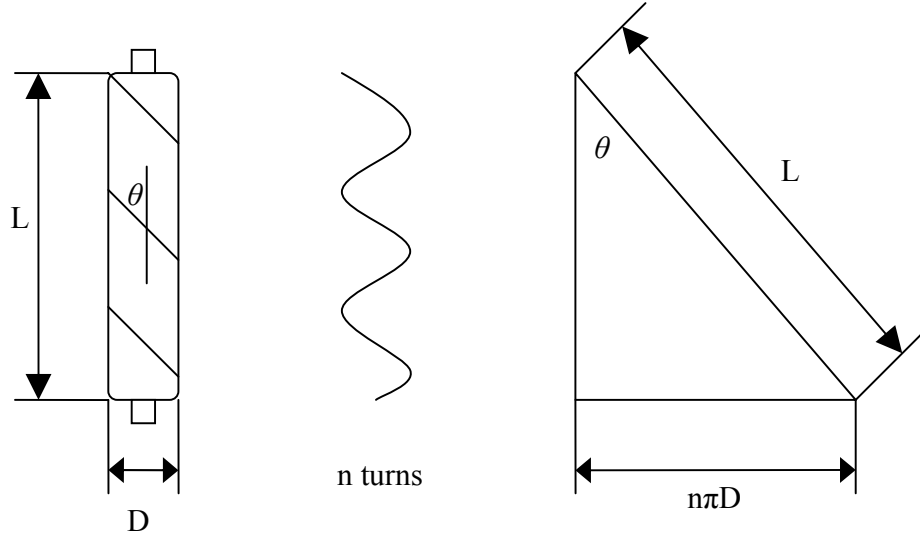


Figure 47 : Geometry of Festo Actuator. The braided fiber turns n times

$$l = b \cos \theta \quad (34)$$

$$D = \frac{b \sin \theta}{n\pi} \quad (35)$$

where the fiber turns n times around the cylinder. The volume of the cylinder can be expressed as

$$V = \frac{1}{4} \pi D^2 l = \frac{b^3}{4\pi n^2} \sin^2 \theta \cos \theta \quad (36)$$

From Eqn. (33), the actuator force can be expressed as

$$F = -P \frac{dV}{dl} = -P \frac{dV/d\theta}{dl/d\theta} = \frac{Pb^2(3\cos^2 \theta - 1)}{4\pi n^2} \quad (37)$$

This expression from Hannaford et al. is a function of the internal pressure, mesh weave angle and the length of the fiber [11], but it is difficult to measure the weave angle.

Tondu and P. Lopez [13], [14] modeled McKibbens using the virtual work principle similar to Hannaford et al. but, they separated the lateral force and axial force

produced by internal pressure. Figure 48 shows the virtual work principle applied to McKibben actuators. The lateral work and axial work produced by internal pressure are shown.

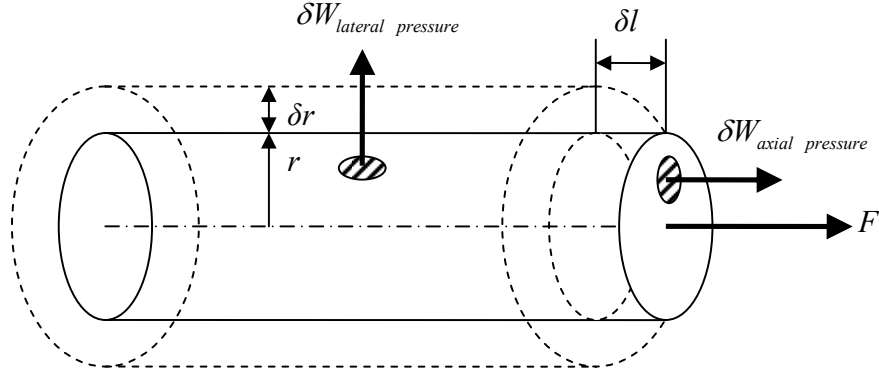


Figure 48 : Virtual Work principle applied to McKibben Actuator.

The virtual principle can be expressed as

$$\begin{aligned} \delta W_{lateral\ pressure} + \delta W_{axial\ pressure} + \delta W_{equilibrium\ force} &= 0 \\ \Rightarrow (2\pi r l P)(\delta r) - (\pi r^2 P)(-\delta l) - F(-\delta l) &= 0 \end{aligned} \quad (38)$$

where r is the radius of the actuator and P is the internal pressure.

The ratio of the current actuator length to the original actuator length is

$$\begin{aligned} (l/l_0) &= \cos \alpha / \cos \alpha_0 \quad \text{and} \quad r/r_0 = \sin \alpha / \sin \alpha_0 \\ \Rightarrow r &= r_0 [\sqrt{1 - \cos^2 \alpha_0 (l/l_0)^2} / \sin \alpha_0] \end{aligned} \quad (39)$$

By applying Eqn. (39) and the derivative of Eqn. (39) into Eqn (38), the actuator force can be expressed

$$F = (\pi r_0)^2 P [a(1 - \varepsilon)^2 - b], \quad 0 \leq \varepsilon \leq \varepsilon_{\max} \quad (40)$$

where $\varepsilon = (l_0 - l)/l_0$, $a = 3/\tan^2(\alpha_0)$ and $b = 1/\sin^2(\alpha_0)$.

Eqn. (40) shows that the actuator force depends on the internal pressure, P and the contraction ratio, ϵ . Figure 42 shows the relationship between the actuator force and contraction ratio (strain) with varying pressures. This is a more useful model because the pressure and contraction ratio are readily measured in the robot.

5.3 Valve System of Robot V

In this section, I will explain and describe a model of the valve system in Robot V. Robot III used 48 three-way valves. The disadvantage of using one three-way valve per actuator is that air can not be trapped in the actuator. The valve positions are (1) air is inlet to an actuator or (2) air is exhausted from the actuator. This is one of the reasons that Robot III did not walk well. Robot V uses 96 two-way valves or 2 on-off valves per actuator. If all of the actuators in the robot are pressurized and their valves are turned off, the robot can stand and withstand disturbances passively, with no sensors or control system. This is an important property for legged robots.

Figure 49 shows the valve configuration for each actuator on Robot V. Each actuator has 2 on-off valves. When the inlet valve is open it supplies pressurized air to the actuator. When it is closed the supply air can not flow into the actuator. When the outlet valve is open, compressed air in the actuator is exhausted to the atmosphere. If both valves are closed, air is trapped in the actuator.

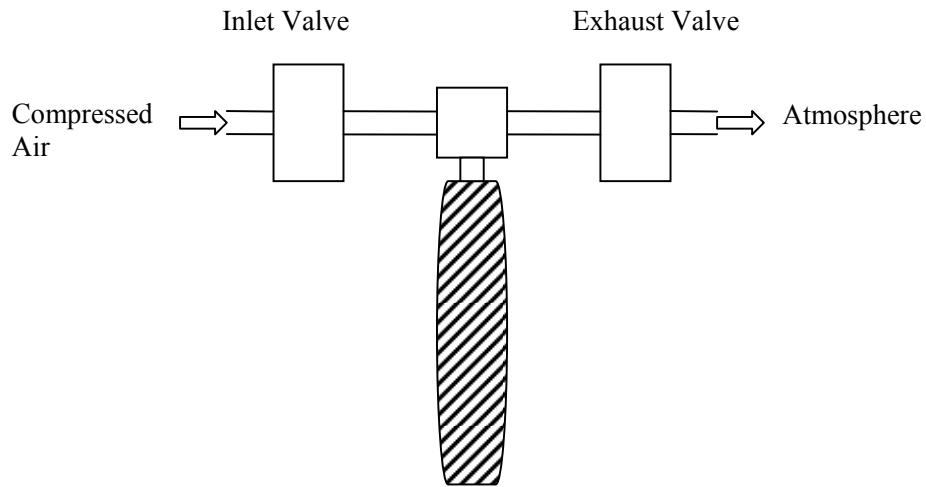


Figure 49 : Valve configuration for Robot V. The inlet valve is connected to a pressurized air supply and the outlet valve is connected to the atmosphere.

5.3.1 Inlet and Outlet Valves

The valves for Robot III and Robot V were purchased from Matrix of Italy. Inlet valves are series 850 nine-channel 2-way normally closed valves. Figure 50 shows a schematic of a Matrix series 850 valve. Air is provided by the reservoir (1) and the shutter is closed and opened by solenoids allowing air to pass into the actuator (2). The O-ring seals and closes the shutter. These valves have open and closing times of 2ms and 5ms, respectively, which is very fast relative to their competitors.

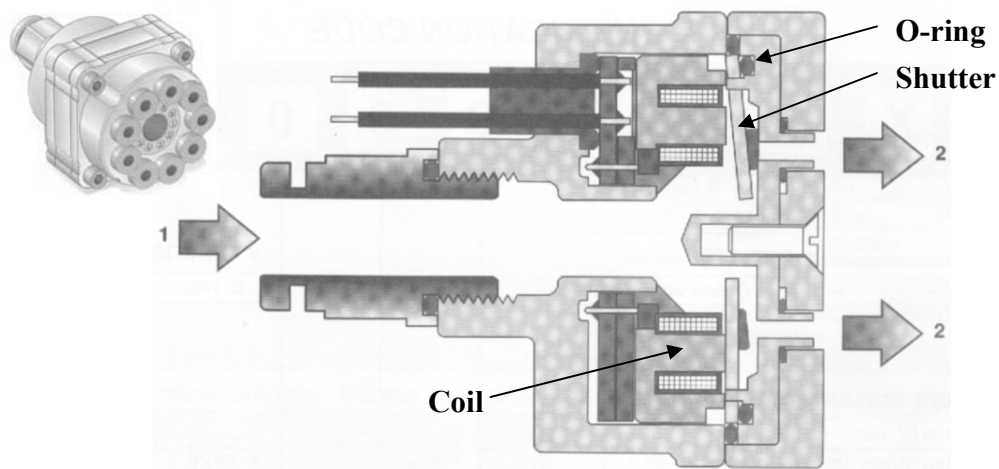


Figure 50 : Schematic of a Matrix Series 850 Valve

The exhaust valves are Matrix series 750 eight channel, 2-way valves. Problems were encountered on previous robots because these valves are intended to function as inlet valves, and are spring-loaded to provide a closing force against pressurized air [15]. In this implementation, the valves were operated in reverse so the high pressure air is on the opposite side of the shutter. Figure 51 shows a schematic of a Matrix series 750 valve. The (1) side is connected to the actuator and the (2) side is exposed the atmosphere. When this was first attempted the valves leaked and did not function properly as exhaust valves. Robb Colbrunn modified the valves by disassembling the valve, removing the springs, and filling their cavities with epoxy [15]. Colbrunn showed that these modifications made the series 750 effective exhaust valves.

Air was supplied from the valves to the actuator through 5/32" nylon tubing, with connections provided by nylon push-in fittings for easy assembly and disassembly. To make air line identification easier, blue hose was connected to extensors, and gray hose was connected to flexors.

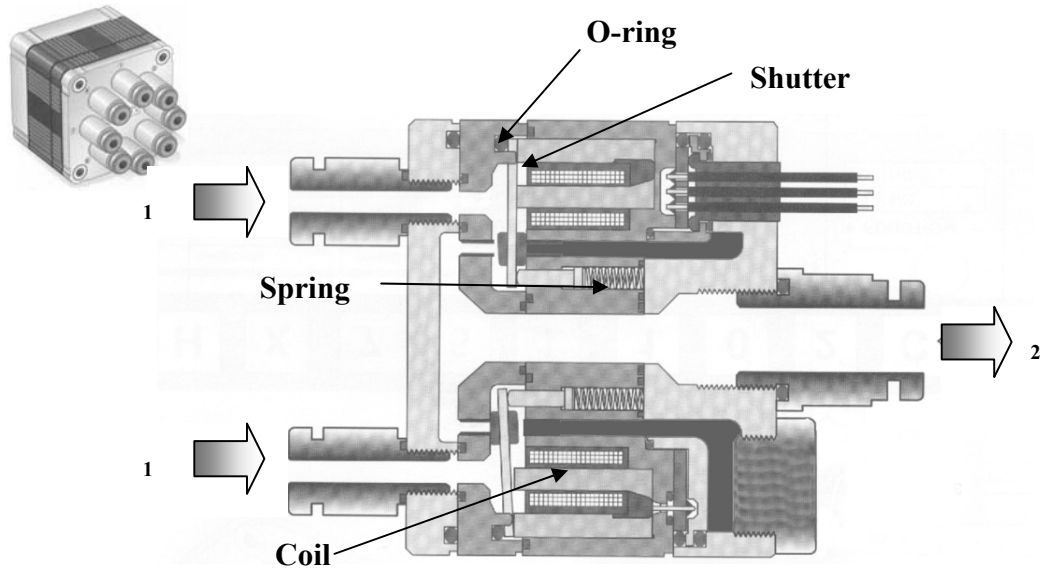


Figure 51 : The Schematic of a Matrix Series 750 valve.

5.3.2 Modeling of Valve

A valve model is necessary to determine the mass flow rate \dot{m} into actuators for an accurate simulation. The formulation of a valve model is included for completeness, but the modeling was done by Nelson [16]. The mass flow rate can be expressed by the following equation [17].

$$\dot{m} = \frac{A_p P_{up} C_q C_m}{\sqrt{T_{am}}}$$

Where A_p = Air passage area of solenoid valve

P_{up} = Upstream air pressure (absolute)

C_q = Flow rate coefficient

C_m = Flow rate parameter

T_{am} = Temperature of upstream air

C_q is empirically determined. It can be expressed by

(41)

$$C_q = \frac{\dot{m} \text{ measured}}{\dot{m} \text{ calculated with } C_q = 1} \quad (42)$$

C_m depends on whether the flow through the valve is choked or not. It is defined by the following expression

$$C_m = \begin{cases} \sqrt{\frac{\gamma}{R} \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}} & \frac{P_{down}}{P_{up}} \leq 0.528 (\text{choked}) \\ \sqrt{\frac{2\gamma}{R(\gamma-1)} \left[\left(\frac{P_{down}}{P_{up}}\right)^{\frac{2}{\gamma}} - \left(\frac{P_{down}}{P_{up}}\right)^{\frac{\gamma+1}{\gamma}} \right]} & \frac{P_{down}}{P_{up}} > 0.528 (\text{not choked}) \end{cases} \quad (43)$$

Where P_{down} = Downstream air pressure (absolute)

$$R = 53.34 \frac{ft \cdot lbf}{lbm \cdot ^\circ R} \text{ air constant}$$

Nelson used the above model and evaluated the empirically determined constants.

Due to the complex geometry of the solenoid valves, the A_p term was lumped with the C_q term and backed out of the pressure-flow rate plot given in the valve specifications.

Nelson concluded that $A_p C_q = 1.8 \times 10^{-5} \text{ ft}^2$ for these valves [16].

5.4 Applied Joint Torque

A relationship between joint torque and joint actuator forces is needed for control calculations. Two Festo actuators, flexor and extensor, are antagonistically coupled to propel each joint. Figure 52 shows a pair of coupled Festo actuators driving a joint. The applied joint torque, τ resulting from these actuator forces can be expressed as

$$\tau = r \cos \theta \cdot (F_{Flexor} - F_{Extensor}) \quad (44)$$

The main problem for control of the robot is determining joint torques that will produce efficient robot movement. One factor that complicates this problem is that there are losses in the actuators. Not all of the predicted actuator force will drive the joint. In the simulation, I assume that the actuators and resulting joint torques are 100% efficient. The entire joint torque given by Eqn. (13) is applied to a joint. The actuator efficiency problem will be discussed further in Chapter 8.

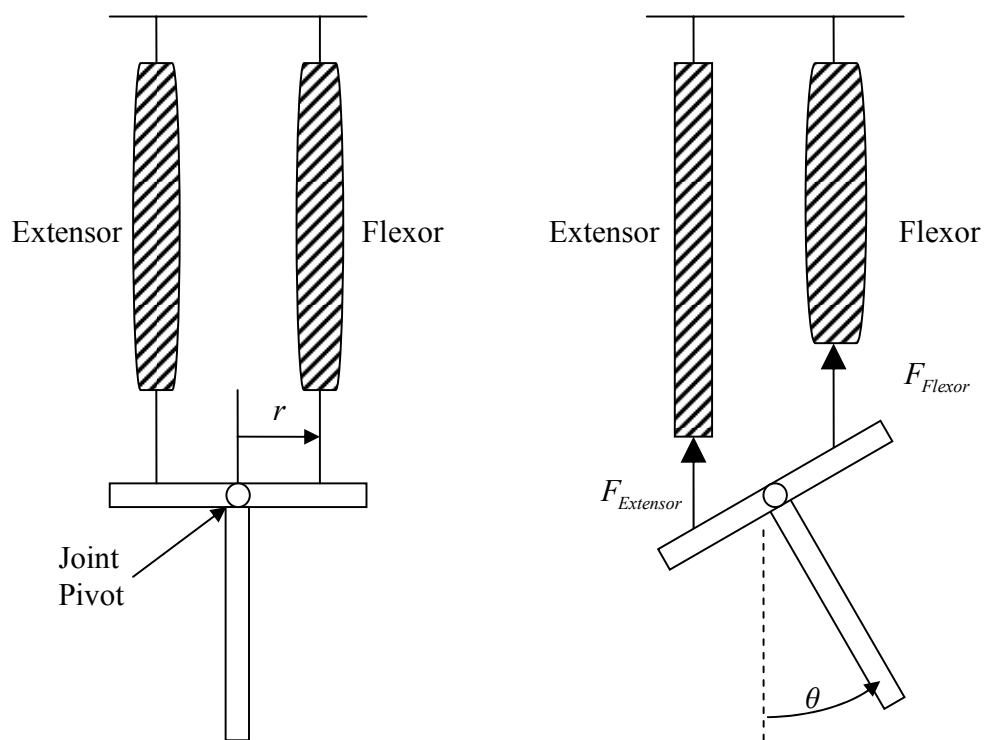


Figure 52 : Antagonistic actuators. Flexor and extensor tensile actuators drive each joint in the robot.

5.5 Dynamic Simulation of Robot V

The purpose of the dynamic simulation is to test the control systems developed in the previous chapters. Before testing control software in Robot V we tested it in simulation. Specifically, I tested my foot paths, neural network inverse kinematics solution, and the Cruse controller for which I determined the weights. By including the actuators and valves in my Robot V model I attempted to capture all of the important dynamics. Therefore, I expected the results to be applicable to Robot V.

In my master's thesis work I developed equations of motion, ground reaction forces and a terrain model for simulation of cockroaches climbing obstacles. The equations of motion were derived using a formulation by Nelson and Quinn [16]. I could have modified that simulation for Robot V, but my advisor wished me to use Yobotics simulation software instead [18]. If it had functioned as we thought this would have saved me time and it has the advantage of having a good visual interface. However, the software was new and it had some deficiencies. Yobotics ground reaction force model and terrain model were not as good as mine. Also, it was not clear how to incorporate my valve and actuator models into Yobotics. The Yobotics support staff worked with me and eventually these problems were solved with patches that permitted me to use my software as I wished. The result is that Yobotics software is now a good solution for this problem if my fixes are included.

Yobotics software is used to develop the equations of motion for Robot V. This software has a good visual interface. Figure 53 shows the robot simulation window, a control variable monitoring window and real-time graphs of variables. We can check any

variable in the monitoring window. Inertias and dimensions are plugged into the simulation. This software reduces the burden of coding equations of motion.

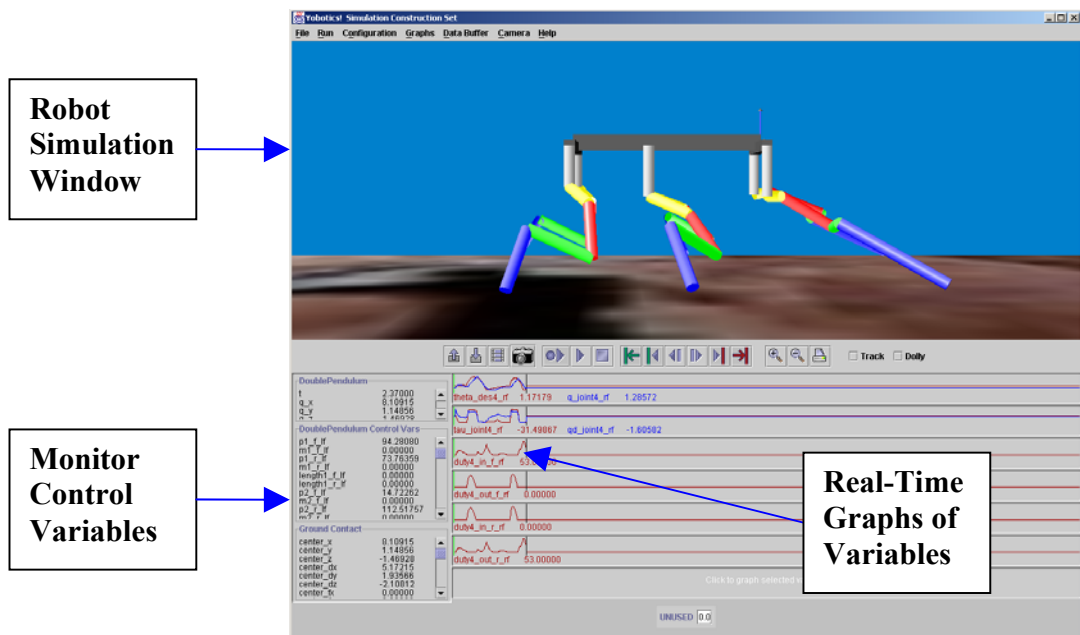


Figure 53 : Dynamic Simulation of Robot V. Yobotics Simulation Set is used for the simulation of Robot V.

5.5.1 Flow Chart of Simulation

The dynamic simulation consists of robot dynamics, controller, ground force model, terrain model and pneumatic actuator-valve model. Figure 54 shows the flow chart of my simulation of Robot V. The simulation starts with the gait controller, which uses a Cruse controller to decide switches to and from swing and stance for each leg to coordinate the legs. The gait controller also determines the desired foot path for each leg. The gait pattern is a function of the commanded speed of the robot, which is input by the user. The desired foot path, especially the swing path is modified by the path equations

based on the current location of the PEP. In the simulation, the paths of the legs are designed to be in a plane with a constant y value as discussed previously.

Foot positions generated by the gait controller are input to the inverse kinematics solving neural networks to get the desired joint angles. The neural networks produce the joint angles corresponding to the desired paths. These desired joint angles are compared with the actual joint angles. Every simulation time step, each joint error ($\theta_{desired} - \theta_{actual}$) is measured and a proportional (P) controller calculates a joint torque to compensate for that joint error. The output of the P controller determines the PWM (pulse width modulation) duty cycles for the four valves controlling that joint. The valves open for a percentage of the PWM period. That percentage is referred to as the duty cycle. If joint error is large, the duty cycle is closer to 100% and more air is input to or outlet from the actuator. The actuator force is determined by the actuator pressure, p and contraction ratio, k in Eqn. (40). Once the flexor and extensor actuator forces are calculated, the joint torque can be calculated using Equation (44). The joint torque is applied to the robot dynamics. The feet are modeled with point contacts and the ground is modeled with linear springs, viscous dampers and Coulomb friction. This ground modeling is explained in my M.S. thesis [19] in detail. The joint torque commands and the ground reaction forces are applied to the robot's equations of motion.

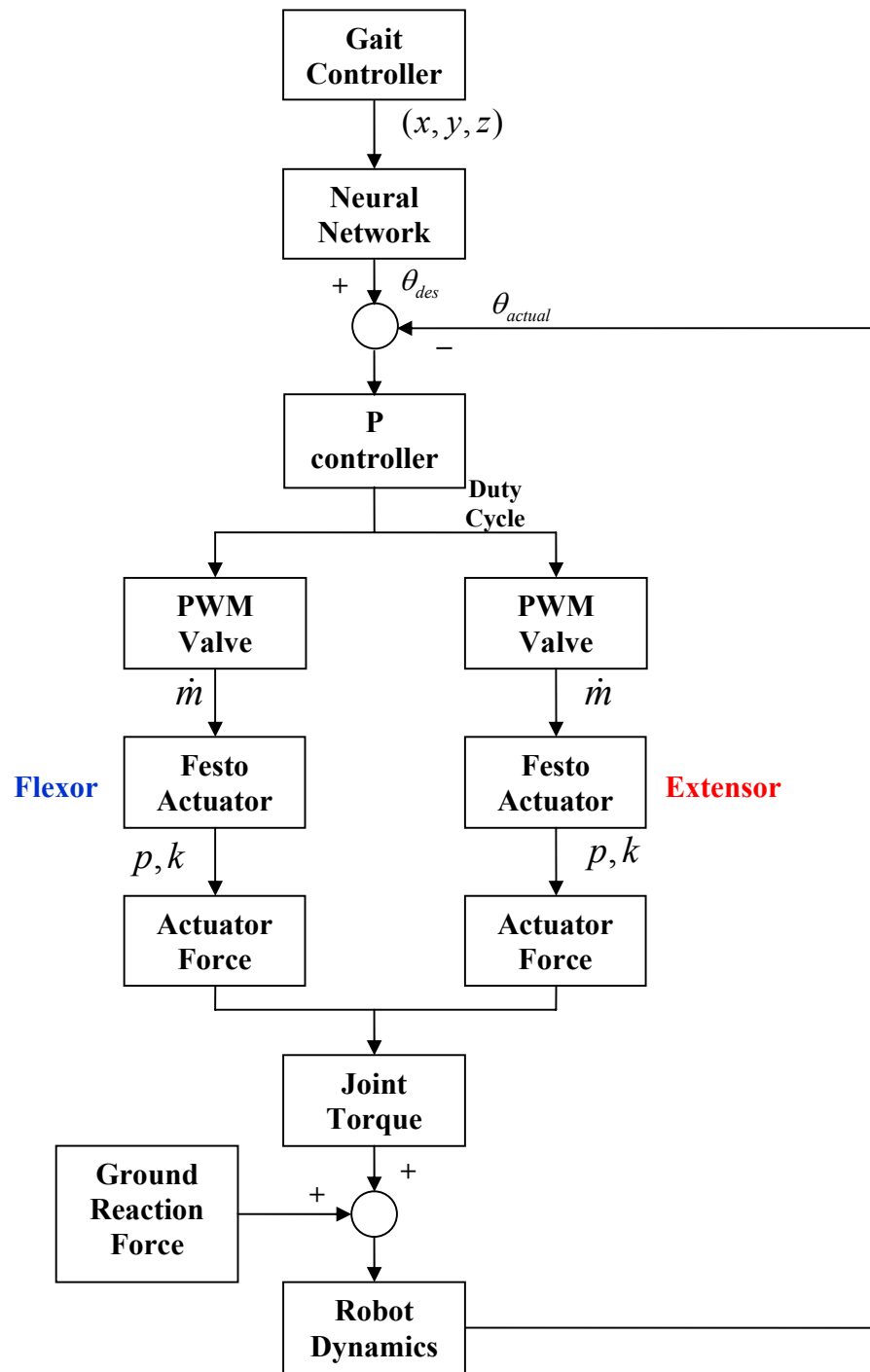


Figure 54 : Flow chart of simulation.

5.5.2 File System of Simulation

Figure 55 is a flow chart of the simulation. To develop a simulation using Yobotics software, the user must write code in the Java language. The simulation of Robot V consists of several Java files in Yobotics Jbuilder. In LinearGroundContactModel.java, the ground model is defined and the calculation of the ground reaction force is described. At every simulation time step, the ground contact algorithm decides whether the foot contacts the ground or not. When the foot contacts the ground, the ground contact algorithm calculates the friction force as well as the vertical ground reaction force. Comments or project notes are written in the file RobotV.html.

All control variables are defined in RoboVController.java for reference as global variables. If we want to monitor these variables in the simulation window, the control values should be included in the monitoring variable lists. All variables are initialized once in initControl() function. In doControl(), the cruseControl() first finds the gait pattern and the desired foot positions for all six legs. The neural networks for the six legs are used to find the desired joint angles. The duty cycles and valve states are determined in valve_state() function. From these the mass flow rate into or out of the actuators is determined.

RobotVIntegrate.java consists of computeDerivativeVector() and Festo_actuator() functions. All the state equations describing the Festo actuator are defined and passed into the Runge-Kutta routine for integration in the computeDerivativeVector() function. The Festo actuator model is defined in Festo_actuator() function. The current joint angle, joint angle velocity, pressure, mass, and states of inlet and exhaust valves are read

as inputs. The outputs of the `Festo_actuator` function are the actuator force, \dot{p} and \dot{m} . All state variables are integrated in the Runge-Kutta routine.

The physical robot configuration is defined in `RobotVrobot.java`. Robot V consists of the robot body and six legs. The simulated robot body is modeled like the real robot's body. The body is constructed frame by frame. Each leg consists of three segments. Each segment has inertia and length values which are equal to those in the real robot. All joint designs are the same as those in Robot V.

The variables which we wish to monitor are defined in `RobotVSimulation.java`. Real-time graphs of variables are shown in the left bottom window in Figure 53. The viewpoint, the viewing size and the magnitude of gravity can be changed in this file.

The ground height and the range of ground space are defined in `WavyGroundProfile.java`. It is possible to design step obstacles in the terrain model by defining the height of the step, but in this simulation, a flat plane is used for ground walking. The range of ground space defines the maximum and minimum values of x and y . If the robot walks outside the ground space, the feet can not detect the ground.

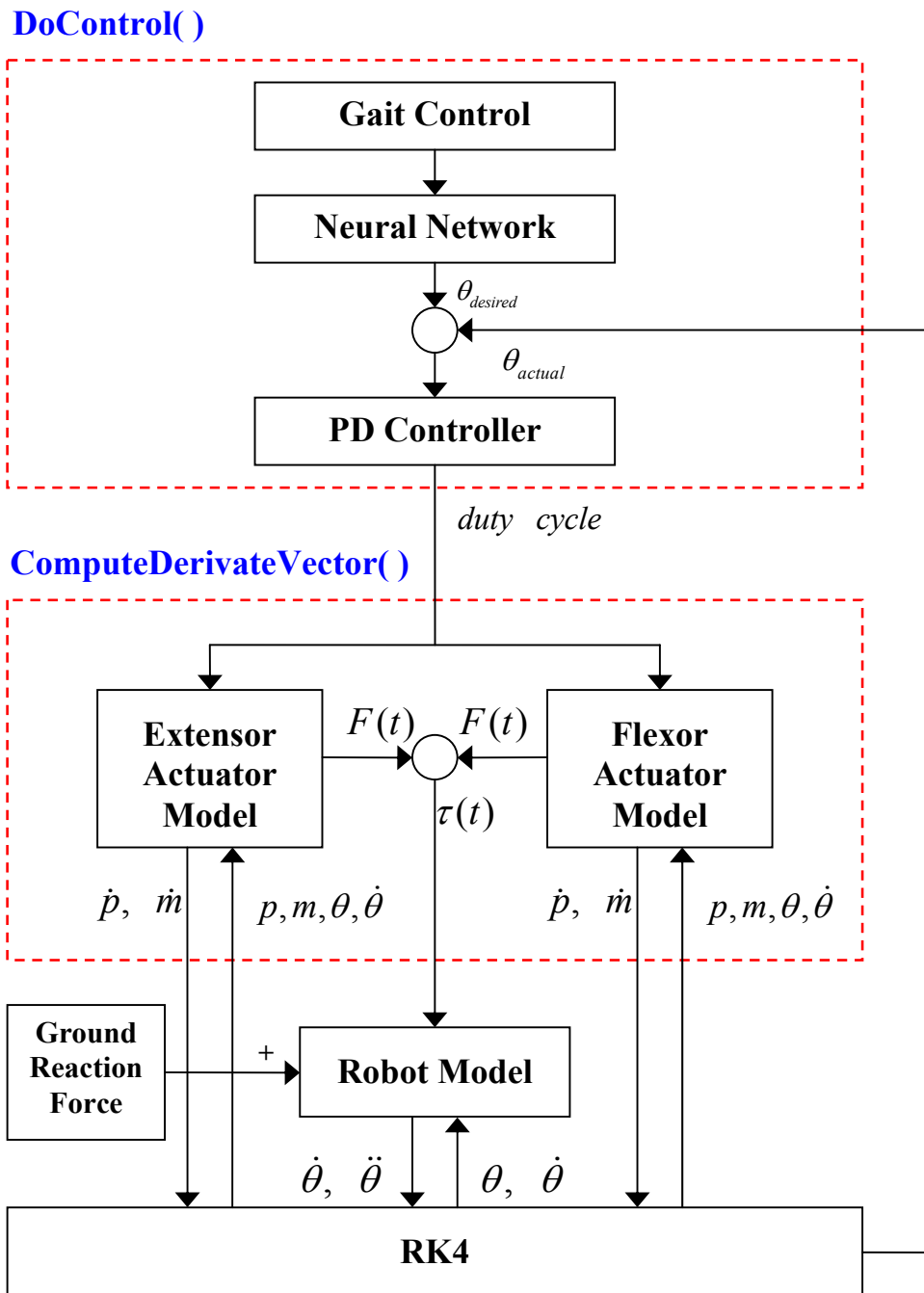


Figure 55 : Detailed algorithm of simulation.

File Name	Function	Function()
LinearGroundContactModel.java	- Definition of Ground Model - Ground Reaction Force	- doGroundContact()
RobotV.html	- Project notes	
RobotVController.java	- Cruise Controller - Neural Network - Control of Valve system - Decision of Valve State	- getControlVars() - cruseControl() - sigmoid() - nnLF(), nnLM(), nnLR() - nnRF(), nnRM(), nnRR() - initControl() - doControl() - valve_state() - valve_state_on() - valve_state_off()
RobotVIntegrate.java	- Festo Actuator Modeling - State Eqns of Festo Actuator	- computeDerivativeVector() - Festo_actuator()
RobotVRobot.java	- Robot Configuration - Joint Definition - Leg Segments Definition	- RobotVrobot() - createBody() - createRightFrontLeg() - createLeftFrontLeg() - createRightMiddleLeg() - createLeftMiddleLeg() - createRightRearLeg() - body(), link1(), link2() - link3(), link4(), link5() - link2_fixed3() - transformInertia()
RobotVSimulation.java	- Control of Monitoring Window	- RobotVSimulation() - main()
WavyGroundProfile	- Definition of Ground Surface	- heightAt(), surfaceNormalAt() - closestIntersectonTo()

Table 9 : File system of simulation

References

- 1 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.
- 2 Gaylord, R.H., 1958, United States Patent 2,944,126.
- 3 Van der Smagt, P., Groen, F. and Schulten, K. Analysis and control of a rubbertuator arm, *Biological Cybernetics*. 75, 433-440, 1996.
- 4 Pack, R.T., Christopher Jr, J.L. and Kawamura, K. A Rubbertuator-Based Structure-Climbing Inspection Robot, *Proceeding of the 1997 IEEE International Conference on Robotics and Automation*, pp. 1869- 1874, Albuquerque, New Mexico, April 1997.
- 5 Website : www.shadow.org.uk
- 6 Pneumatic Catalog 2004 – Pneumatic Muscle, <http://www.festo.com>
- 7 Kingsley, D. A., Quinn, R. D., Ritzmann, R. E. A Cockroach Inspired Robot With Artificial Muscles, *International Symposium on Adaptive Motion of Animals and Machines (AMAM 2003)*, Kyoto, Japan.
- 8 Berns, K., Grimminger, F., Hochholdinger, U., Kerscher, T. and Albiez, J. Design and Control of a Leg fro the Running Machine PANTER, *Proceedings of ICAR 2003, The 11th International Conference on Advanced Robotics* Coimbra, Portugal, Jung 30- July 3, 2003.
- 9 Kerscher, T., Albiez, J., Zoellner, J. M. and Dillmann, R. AirInsect – A New Innovative Biological Inspired Six-Legged Walking Machine Driven by Fluidic Muscles, *Proceedings of IAS 8, The 8th Conference on Intelligent Autonomous Systems*, Amsterdam, The Netherlands, 10-13 March 2004
- 10 Klute, G.K., Czerniecki, J.M. and Hannaford, B. McKibben artificial muscles: pneumatic actuators with biomechanical intelligence, *Proceedings. 1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 221 – 226, Sept. 1999
- 11 Chou, Ching-Ping, Hannaford, B. Static and Dynamic Characteristics of Mckibben Pneumatic Artificial Muscles, *Proceedings of 1994 IEEE International Conference on Robotics and Automation*, Vol.1, pp. 281 - 286. May 1994

- 12 Chou, Ching-Ping, Hannaford, B. Measurement and Modeling of McKibben Pneumatic Artificial Muscles, *IEEE Transactions on Robotics and Automation*, Vol. 12, Issue: 1 , pp. 90 – 102. Feb. 1996
- 13 Tondu, B., Lopez, P. Modeling and Control of McKibben Artificial Muscle Robot Actuators, *IEEE Control Systems Magazine*, Vol. 20, pp. 15-38. April 2000.
- 14 Tondu, B., Boitier, V. and Lopez, P. Naturally compliant robot-arms actuated by McKibben artificial muscles, *1994 IEEE International Conference on Humans, Information and Technology* , Vol. 3, pp. 2635 – 2640. 2-5 Oct. 1994
- 15 Colbrunn, Robb. Design and Control of a Robotic Leg with Braided Pneumatic Actuators, M.S. Thesis, CWRU. 2000
- 16 Nelson, Gabriel. Learning about control of legged locomotion using a hexapod robot with compliant pneumatic actuators, Ph. D dissertation. May 2002
- 17 Ye, N., Scavarda, S. Betemps, M. and Jutard A. Models of a pneumatic PWM solenoid valve for engineering applications, *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, Vol. 114, pp. 680-688. Dec 1992
- 18 Yobotics! Simulation Construction Set. Yobotics, Inc. October 19, 2001
- 19 Choi, Jongung. Dynamic Simulation of Cockroach Climbing. M.S. Thesis, CWRU. 2000

Chapter 6

Metrics

After changing the robot's design or controller, we have to determine how much the robot's performance is improved if at all. To do so, quantitative performance measures must be defined. These metrics are described in this chapter before simulation and implementation results are shown in later chapters.

6.1 Joint Angle Position Error

In the nominal control system, a proportional feedback controller drives each joint. A joint torque is computed as the difference between the actual and desired joint positions multiplied by a gain. The actual joint angles can not exactly follow the desired joint angles because of delays in the control system. These delays can be considerable because the system is pneumatic and it takes time to accelerate air from the supply line through the valve ports, through the air lines and into the actuators. Even small joint angle errors can cause significant foot position errors. Reducing the joint angle errors can improve the robot's performance.

To understand the definition of the joint angle error metrics, the right front leg will be used as an example. The right front leg is tested with and without a tensioning

reflex (with tensioning reflex in test175 and without tensioning reflex in test 176). Figure 56 shows plots of the desired and actual joint angles for the right front leg. Figure 57 shows the difference between the desired and the actual joint angles and the absolute values of errors are plotted in Figure 58. Then, to compare the quantitative values, the mean or average values of joint angle errors are calculated. In Figure 59, the joint angle errors for air-walking without tension reflex are compared with those for air-walking with tension reflex. The results will be discussed in a later chapter. These plots are presented here as an example of a joint angle error metric.

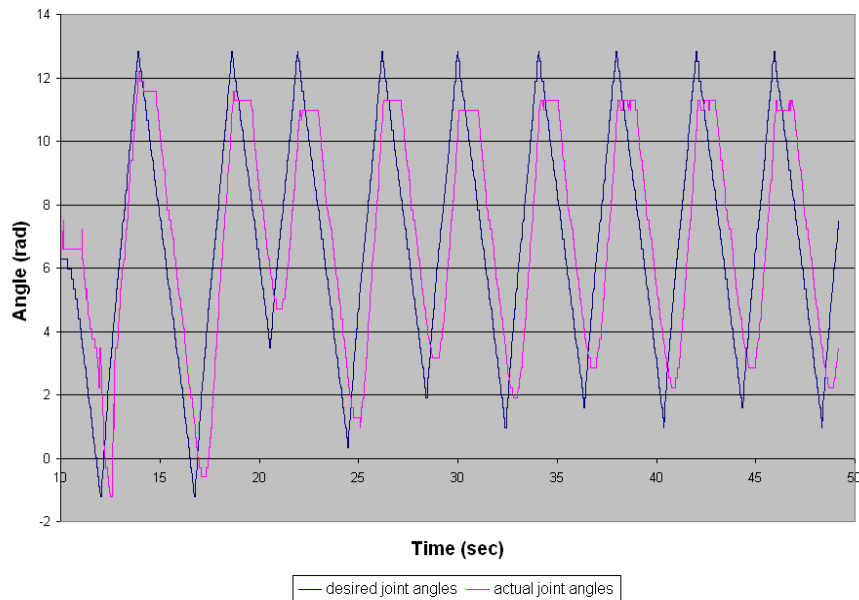


Figure 56 : Desired vs. actual joint angles of Gamma joint on right front leg.

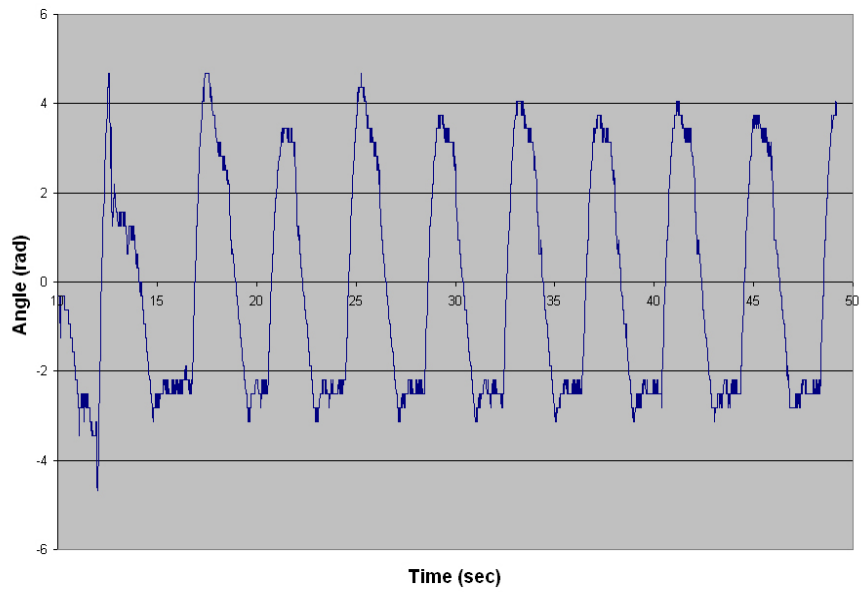


Figure 57 : Errors between desired and actual joint angles

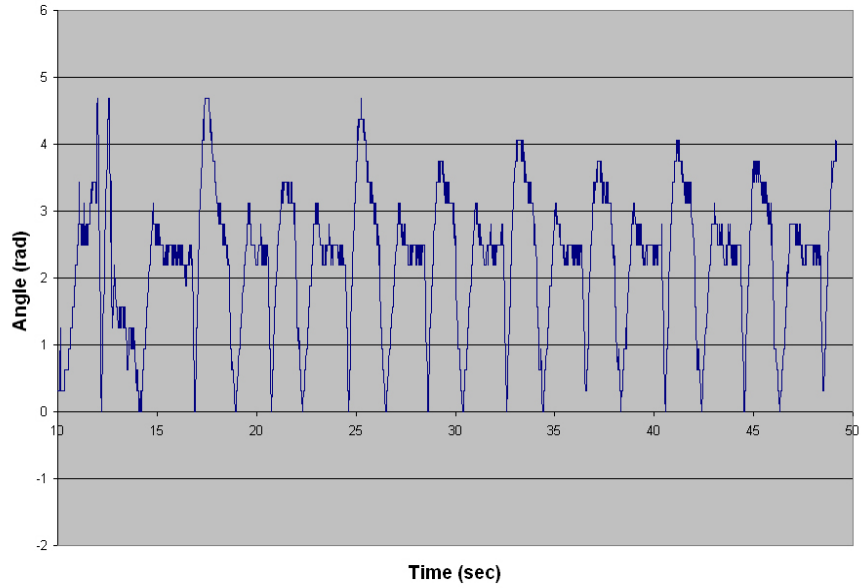


Figure 58 : Absolute errors between desired and actual joint angles

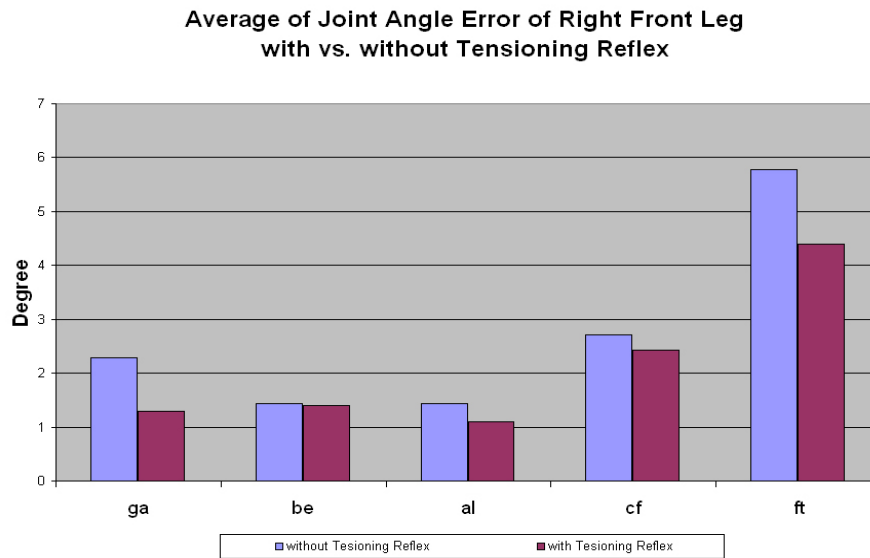


Figure 59 : Comparison of average joint angle error with and without tensioning reflex.

6.2 Foot Position

Foot position error is the basis for another performance metric. Even if the joint angle errors are small, the foot position should be checked to verify that the robot leg follows the desired path. For better performance, the average foot position error should be decreased and the range of position error should be small.

Foot position is found by forward kinematics using individual joint angles. Figure 60 shows the desired and actual foot paths of the right front leg with and without the tensioning reflex. The mean values of position errors and the range of position errors are calculated and plotted in Figure 61. The averages and ranges of position errors decrease

when the tension reflex is applied to the joint. Figure 61 shows the mean values of position errors and range of foot position errors.

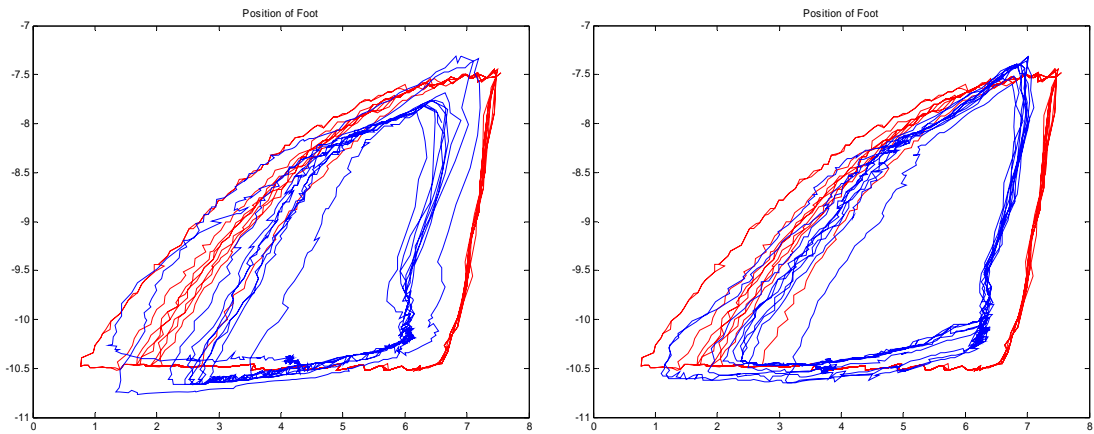


Figure 60 : Desired vs. actual foot path of right front leg with and without tensioning reflex

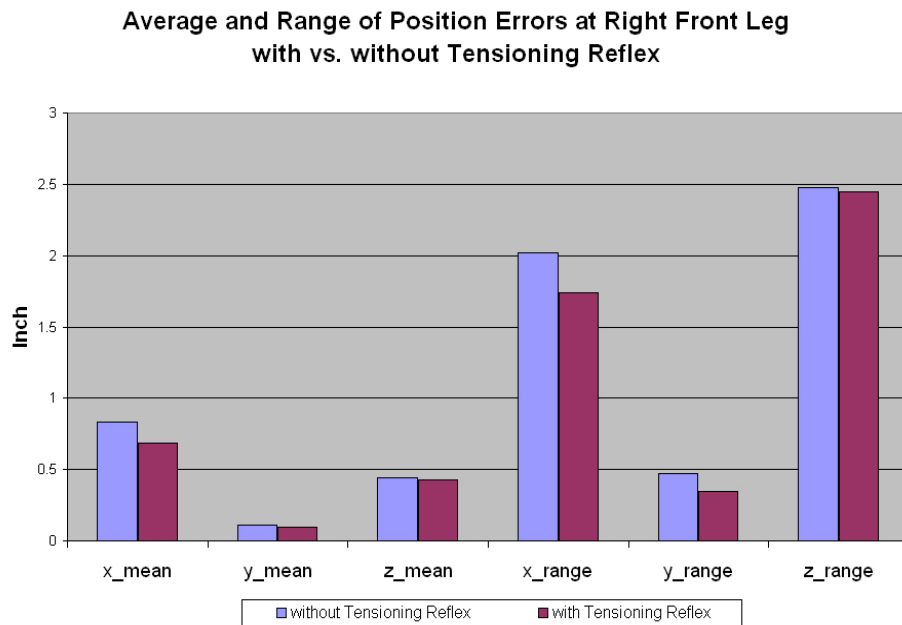


Figure 61 : Averages and ranges of foot position error without and with tensioning reflex

6.3 Velocity

The robot's velocity is another performance metric. The speed of the robot can be determined by measuring the distance the robot moves between each frame of a video. High speed video will be used when the robot moves rapidly. After making Robot V walk, this metric will be an important parameter to measure its performance.

6.4 Body motions during walking

Another performance metric is based on the body motions of the robot. The robot's body bounces, rolls, pitches and yaws when it walks. Because the robot's design is based on cockroach, if it walks in a cockroach manner, we expect that its body motions will also be similar to a scaled up in size cockroach. To make this comparison, the robot's body motions must be measured. There are two ways to measure its body motions. One is to videotape Robot V as it moves and then analyze the video. The other is to use the joint angle positions of the stance legs and forward kinematics to measure body motions. Once the robot's body motions are measured they can be compared to those of a cockroach walking in a similar gait. Bounce, roll, pitch, and yaw motions can be compared separately.

References

Chapter 7

Simulation Results

In this chapter, results of simulations of Robot V will be discussed. The foot path, inverse kinematics, and gait controllers are tested in simulation before they are tried on Robot V. This is done to reduce damage to the robot. I also discover if the actuators are strong enough to generate the forces predicted by the simulation.

7.1 Air and Ground Walking with Pre-Planned Gait

The pre-planned gait control was explained in section 5.1. Unlike the Cruse controller, the pre-planned gait controller was designed by hand to produce a fixed tripod gait (Figure 33). The simulated robot is tested in “air walking” and ground walking using the pre-planned gait. In air walking, the robot’s body is supported so that its legs can not contact the ground.

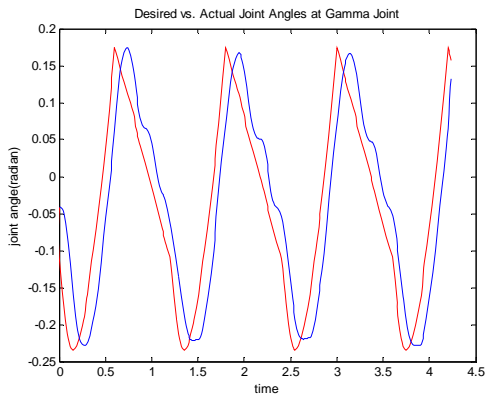
7.1.1 Joint Angles for Air and Ground Walking with Pre-Planned Gait

To see how well the legs follow desired joint positions without external disturbances from ground reaction forces, the simulated robot’s body is supported such

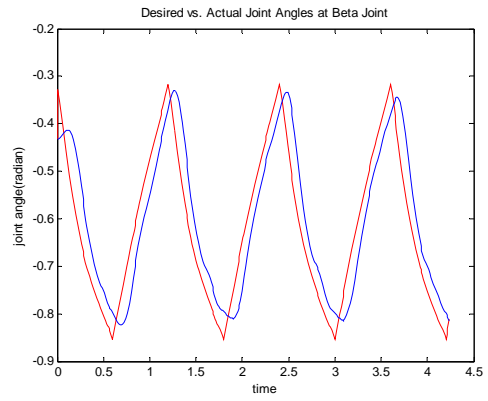
that its feet can not touch the ground. The desired swing foot paths were chosen as parabolic curves. After air walking performed well, the simulated robot was placed on the ground to test ground walking.

Figure 62 shows joint angles as functions of time for the right front leg. The red and blue lines represent the desired joint angles and actual joint angles for air walking, respectively. In Figure 62, the joint angles in the right front leg generally follow the desired joint angles, but with a time delay. The time delays between the desired and actual joint angle are caused by the inherent delay in a negative feedback system and the pneumatic actuator system. It takes time to move air from the supply line through valves and into an actuator. The Alpha joint angles are most affected by gravity because they lift the leg.

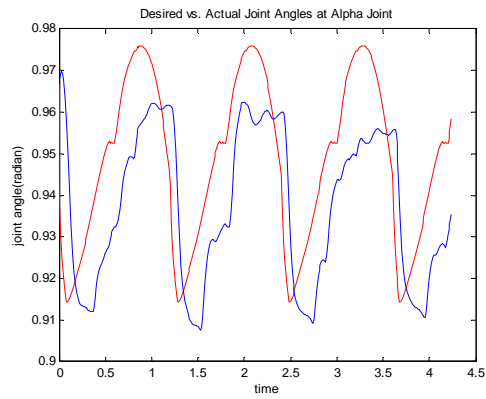
Figure 63 shows joint angles as functions of time for the right front leg for a ground walking case. After air walking test, the proportional gains are tuned for ground walking. The actuators produce enough joint torque for walking. The Alpha joint of right front leg is most influenced by ground reaction forces. The Alpha joint supports body weight, so tracking errors for this joint are expected to be larger than for other joints.



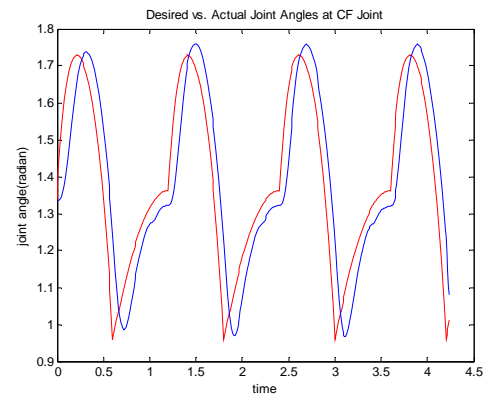
(a)



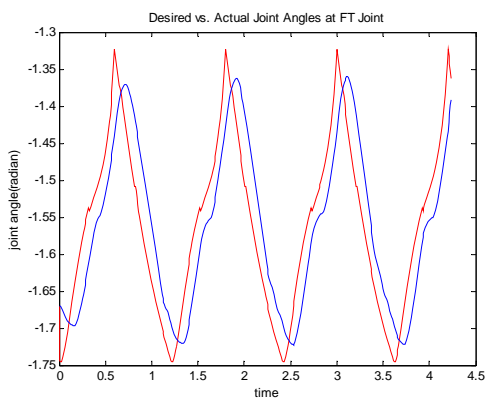
(b)



(c)

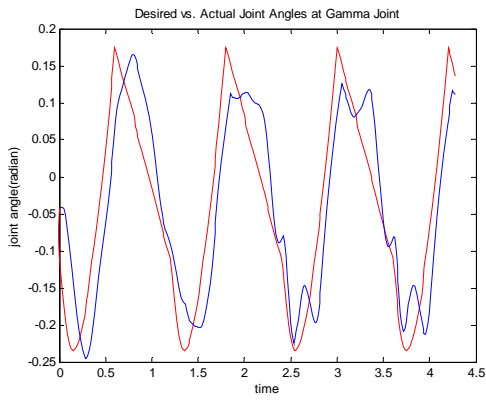


(d)

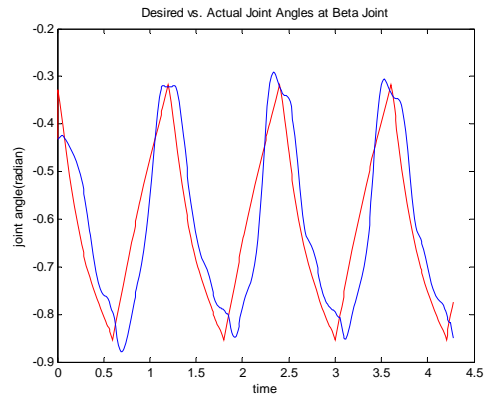


(e)

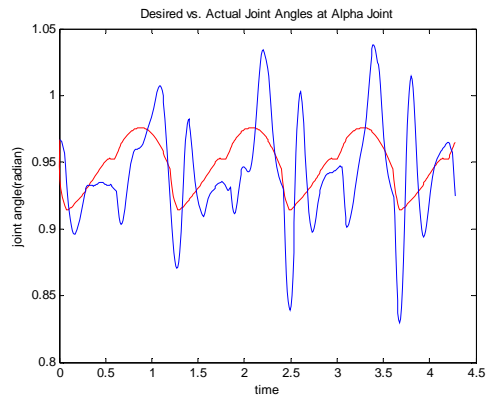
Figure 62 : Desired vs. actual joint angles and foot path on front right leg with simple gait during air walking. Figure 62 (a), (b), (c), (d) and (e) show the gamma, beta, alpha, cf and ft joints respectively. Figure 62 (f) shows the desired and actual joint angles.



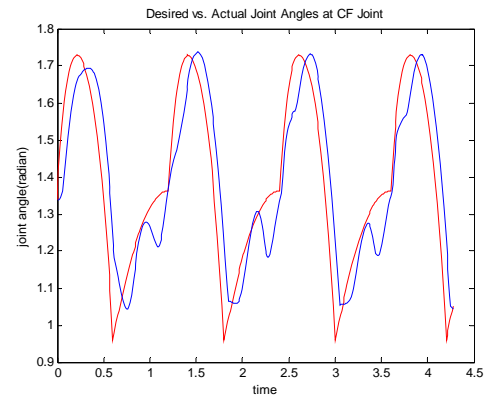
(a)



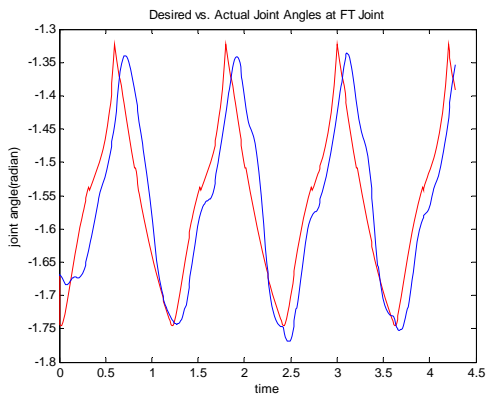
(b)



(c)



(d)



(e)

Figure 63 : Desired vs. actual joint angles on right front leg with simple gait during ground walking. From the left top, the gamma, beta, alpha, cf and ft joint are shown. The red and blue lines represent the desired and actual joint angles.

7.1.2 Foot Paths and Body Bounce using Pre-Planned Gait

The foot paths during air and ground walking are shown in Figure 64. All paths are expressed with respect to the Body-Coxa joint. The red and blue lines represent the desired and actual foot paths. Figure 64 (a) and (b) show the foot paths for air and ground walking for the right front leg, respectively. During air walking, in spite of individual joint angle errors, the foot paths follow closely the desired path. During ground walking, external ground reaction forces strongly affect the foot path and they have larger deviations from the desired. Figure 64 (c) and (d) show the foot paths of the right middle leg during air and ground walking. Because the middle legs support more body weight, the foot paths for ground walking can not follow the desired path. To maintain body elevation, the actual middle leg paths are moving back and forth on an inclined line. Figure 64 (e) and (f) show the foot paths of the right rear leg during air and ground walking. The main function of the rear leg is to push the robot body forward. During ground walking, the straight foot path of the rear leg along the inclined line effectively pushes the body.

During walking, the robot body undergoes cyclic motion. In a legged robot or animal, body motion is unavoidable and even desirable in some cases. The cockroach body bounces vertically, rolls, pitches and yaws [1]. Figure 65 shows the body bounce or vertical movement of the center of mass of Robot V during ground walking using the pre-planned tripod gait. The vertical bounce is 12% of body height, whereas cockroach body bounce has been measured to be 10% of its body height. In this simulation, Robot V is in

steady state motion after 1.5 seconds, but after stabilizing, the magnitude of body bounce grows larger.

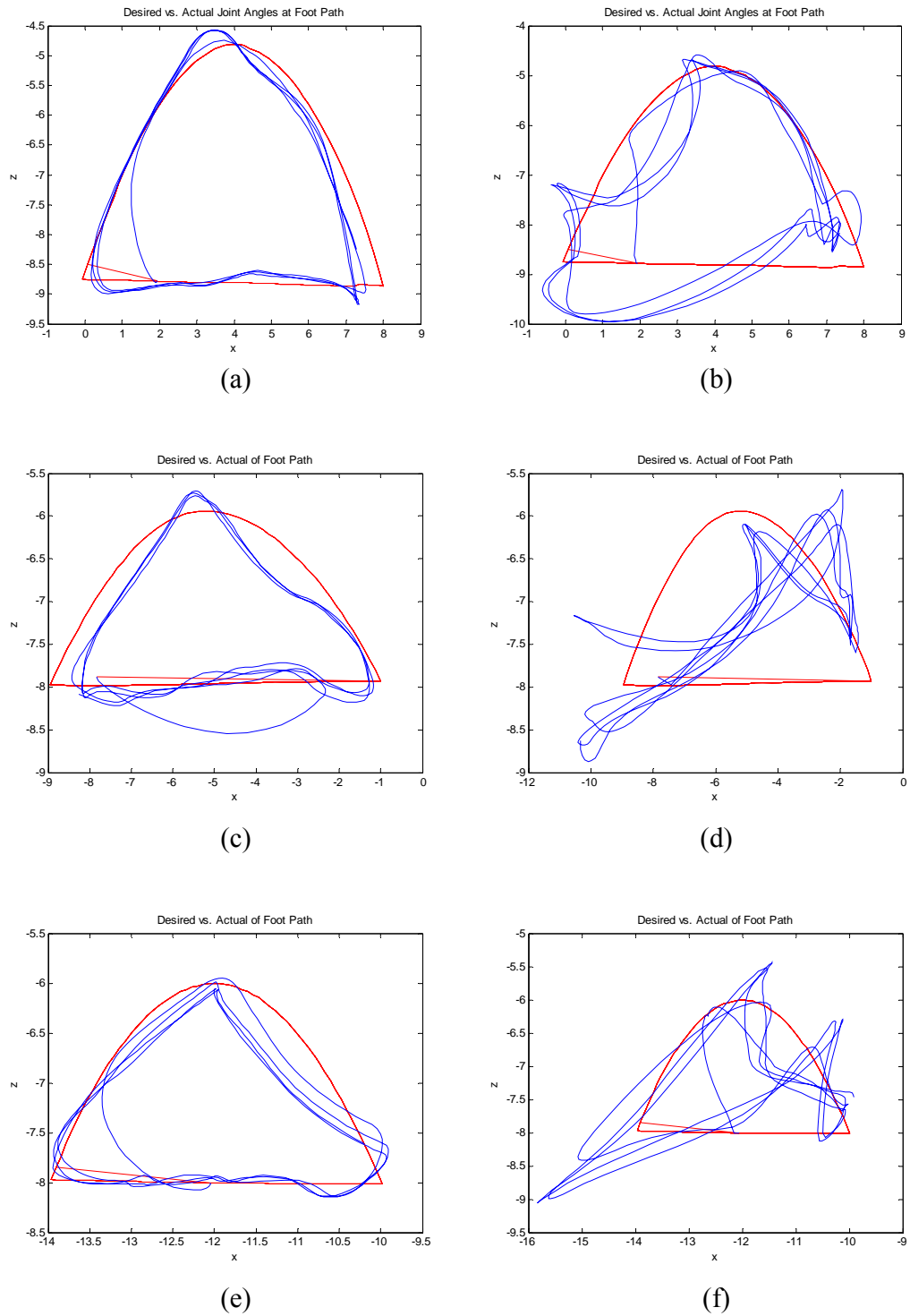


Figure 64 : Desired vs. actual foot paths for air and ground walking with pre-planned gait of right front, middle and rear legs. (a), (c) and (e) show the right front, middle and rear legs for air walking. (b), (d) and (f) show the right front, middle and rear legs for ground walking.

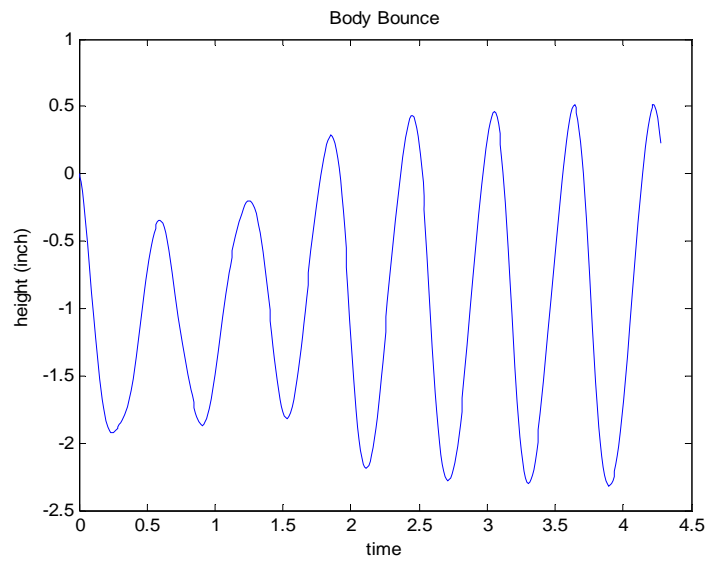


Figure 65 : Body bounce of Robot V with pre-planned tripod gait during ground walking.

7.2 Air and Ground Walking with Cruse Gait Control

In this section, simulation results for air and ground walking using the Cruse controller are shown. The Cruse controller produces gait patterns automatically, whereas the tripod gait used in the previous section was preplanned. The purpose of this work is to test the Cruse controller.

7.2.1 Joint Angles for Air and Ground Walking using Cruse Controller

For a particular command speed the Cruse controller produced the gait shown in Figure 37 during air walking. This command speed was constant for the results shown in

this section. However, note that the gait changes based on external disturbances such as ground reaction forces. The gait is shown in Figure 37 for ground walking results shown in this section.

Joint motions versus time for the simulated robot performing air walking using the Cruse controller are shown in Figure 66. Joint motions versus time for the simulated robot performing ground walking using the Cruse controller are shown in Figure 67. As with ground walking using the pre-planned gait, the Alpha joint on the front leg is most affected by ground reaction forces. During ground walking, the joint angle errors increase. The ground reaction forces prevent the leg from following the desired position. This causes a rapid production of large joint torques because joint torque depends on the joint errors. After this large increase in the angle error, the actual joint angle drops to compensate the joint difference. In Figure 67 (d), we can see large joint angle changes in the transient period from swing to stance ($t=1.2, 2.6, 4.0$).

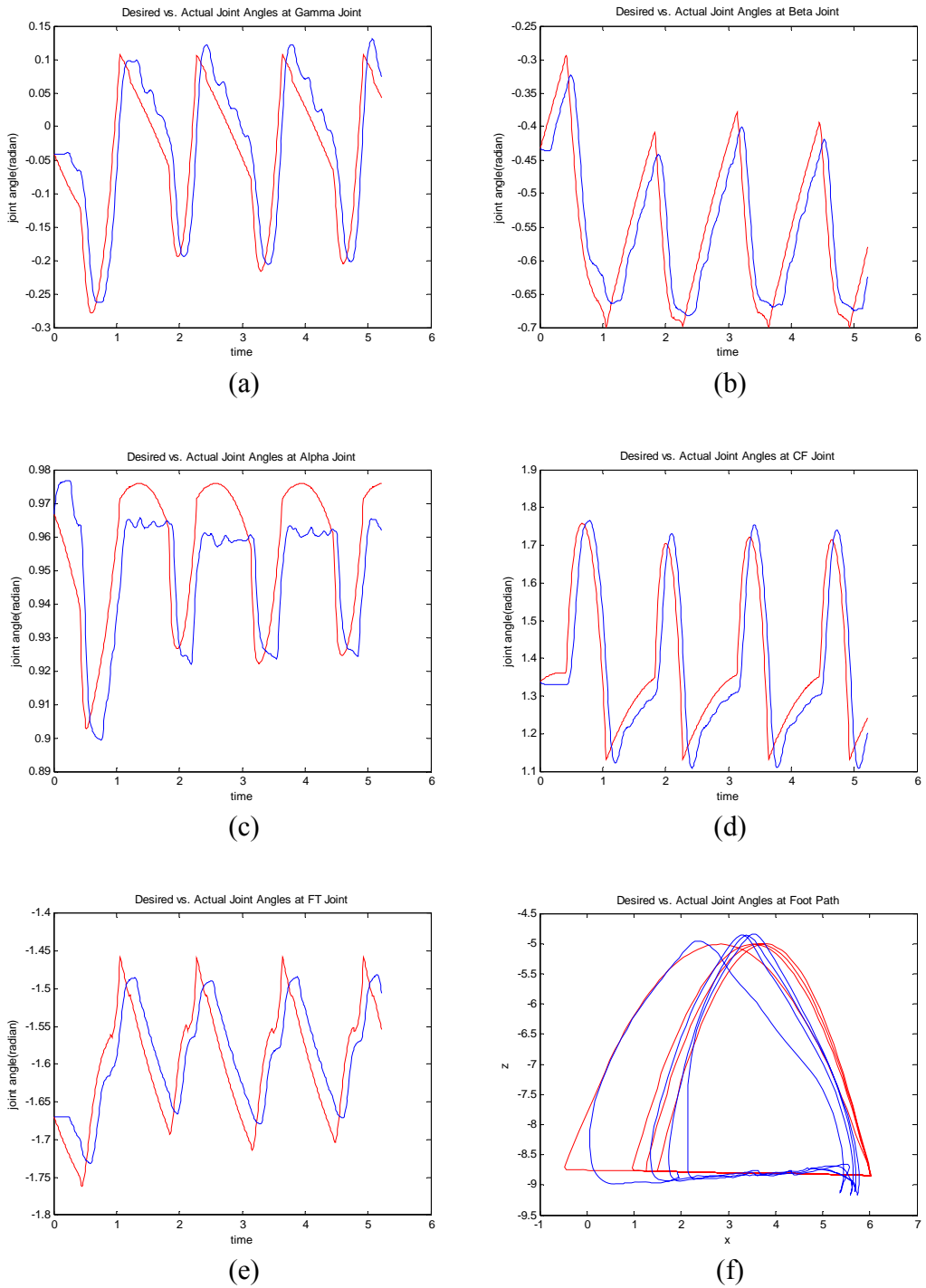


Figure 66 : Desired vs. actual joint angles on right front leg with Cruse Controller during air walking. From the left top, the gamma, beta, alpha, cf and ft joint are shown. The red and blue lines represent the desired and actual joint angles

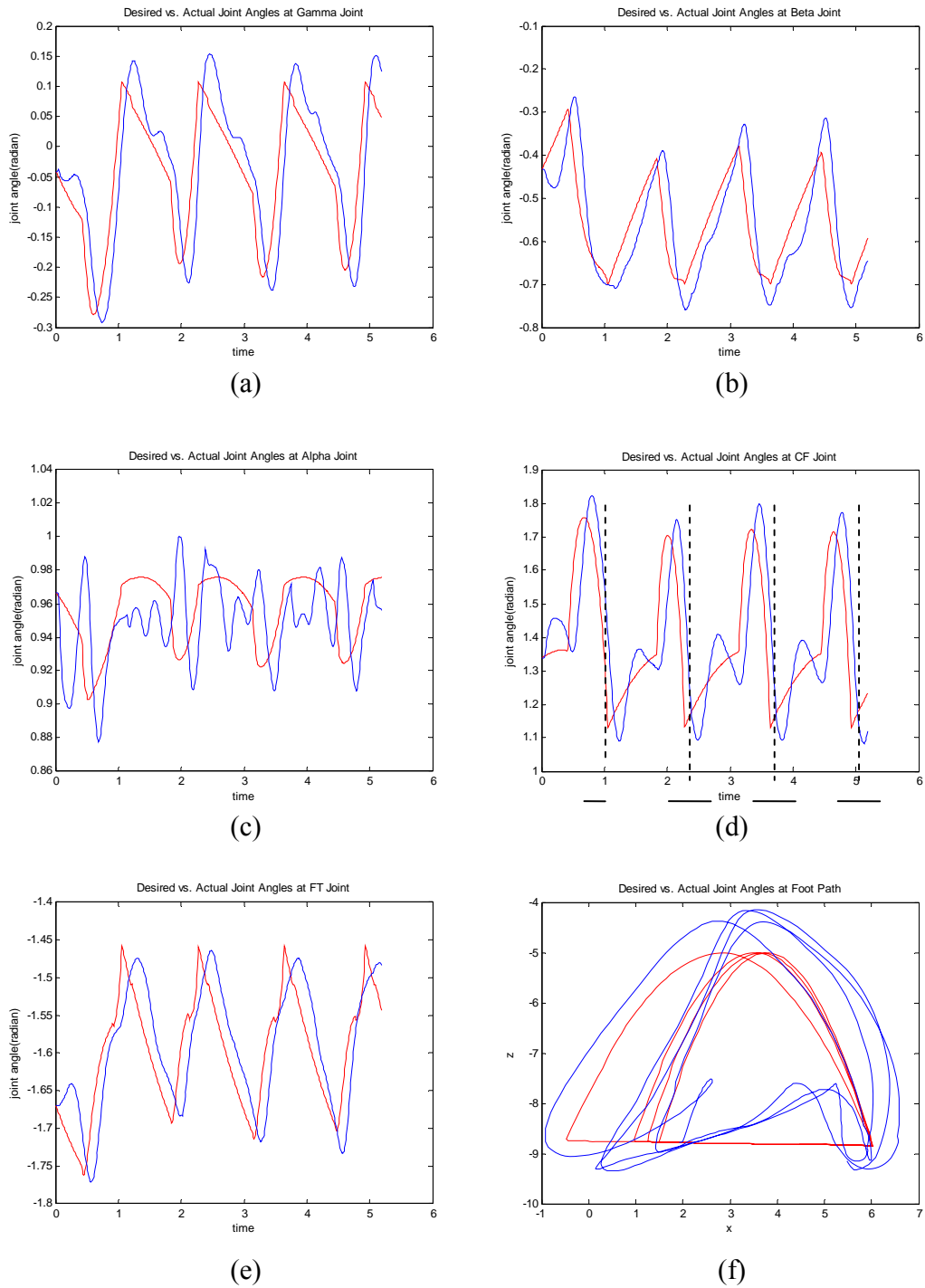


Figure 67 : Desired vs. Actual Joint Angles on Front Right Leg with Cruse Controller for simulated ground walking. From the left top, the gamma, beta, alpha, cf and ft joint angles are shown. The red and blue lines represent the desired and actual joint angles, respectively.

7.2.2 Foot Path and Body Bounce using Cruse Controller

The foot paths corresponding to the joint angles displayed in the previous section for air and ground walking with Cruse controller are shown in Figure 68. During air walking the joint angles closely follow the desired foot path. However, during ground walking, the actual foot paths are strongly affected by ground reaction forces. Body bounce, vertical motion of the center of mass, is shown in Figure 69 for ground walking. Because the PEP position is changed by the Cruse controller each cycle, the foot paths are also constantly being changed. This causes an irregular body bouncing motion, which is unlike body bounce with the pre-planned, fixed gait. In the initial state, Robot V is dropped from a small height. Before 2 seconds, Robot V is not in steady state. After that time, the body bounce is larger because of Robot V overcomes the dropping effect.

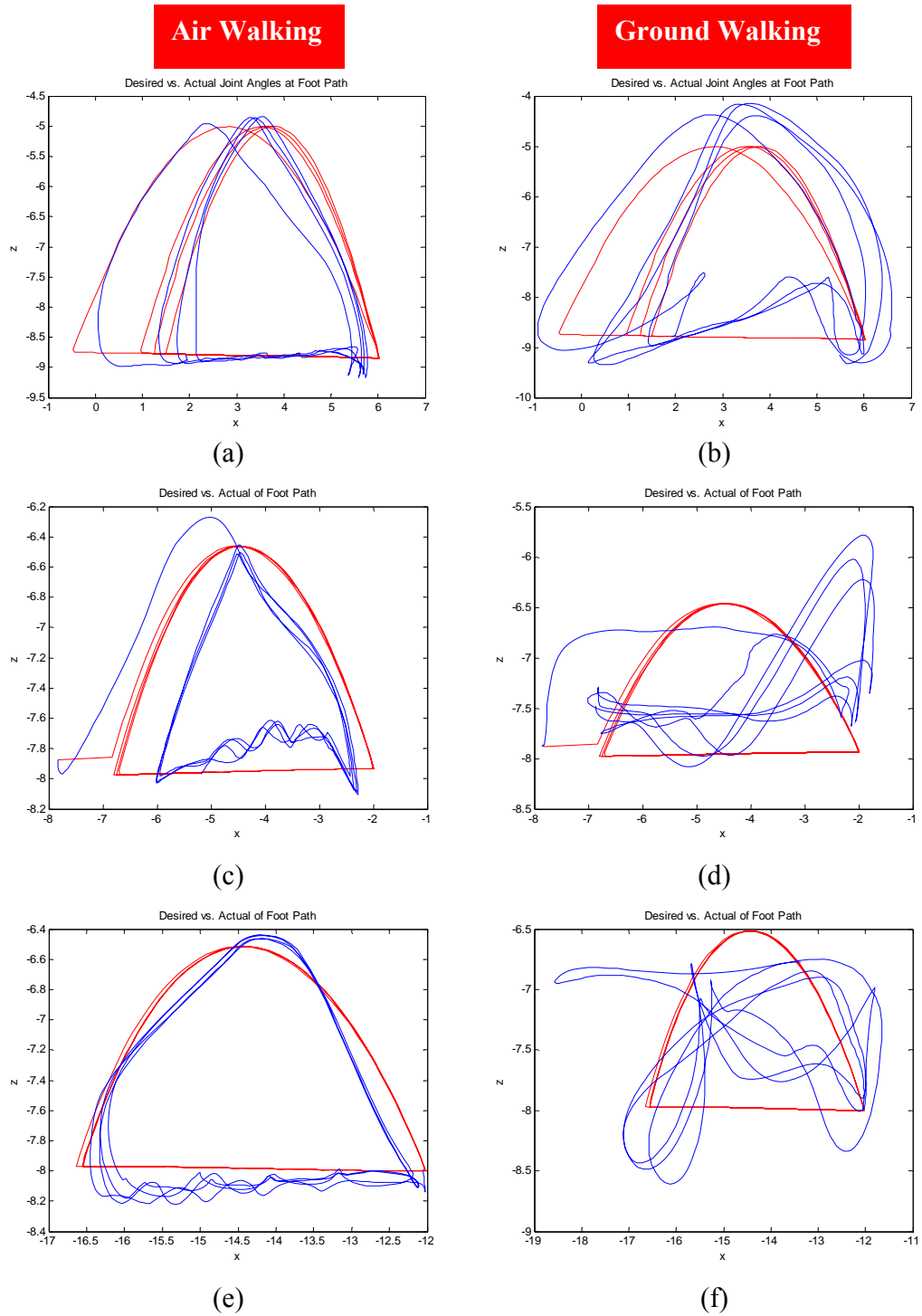


Figure 68 : Desired vs. Actual Foot Paths in right front, middle and rear legs for Air and Ground Walking with Cruse Controller. (a), (c) and (e) are for air walking. (b), (d) and (f) are for ground walking.

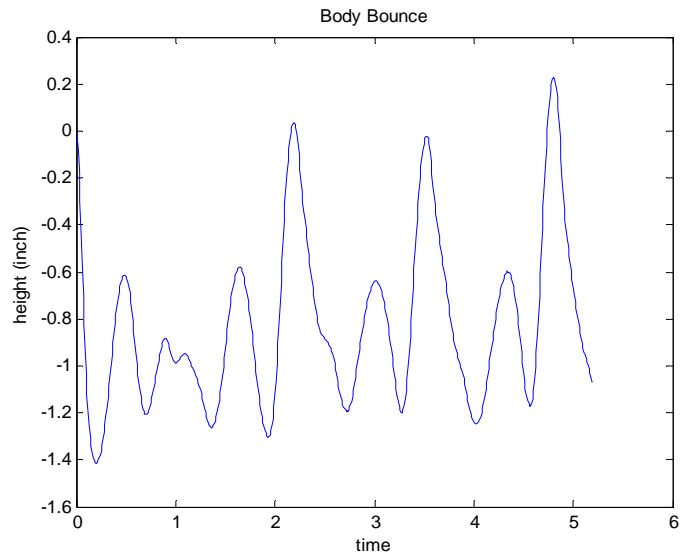


Figure 69 : Body Bounce of Robot V using Cruse Controller

7.3 Conclusions

When the modeled Robot V is suspended in air so that its legs can not touch the ground, it can move its legs to follow desired foot paths and coordinate its legs in insect gaits using the control algorithms I developed. Joint motion and foot path histories show that the desired paths are followed, but with a time delay that is to be expected when PD control and pneumatic actuators are used. The gait controller in combination with the neural network inverse kinematics solver successfully coordinates the joints and legs.

Simulated ground walking is not as successful. The joint histories and foot paths are very different than the designed trajectories. Despite these differences, the simulated Robot V can walk forward using my actuator and valve models. The simulated Robot V

walked with a speed of 7.32 inch/sec using the pre-planned fixed tripod gait and 6.47 inch/sec with the Cruse controller. The pre-planned gait used the tripod gait from the initial state. But as Kingsley pointed out in his dissertation [2] the Cruse controller needs time to form the tripod gait.

The joint angles and foot paths during air walking follow the desired trajectories, but this is not the case for ground walking. The joints that support more weight are perturbed the greatest by ground reaction forces. These results suggest that force feedback will be necessary for the robot to walk well. In later chapters, ground walking using position control will be attempted in hardware to confirm this.

My control algorithms were tested in simulation before they were tried in hardware. This reduces the possibility of damage to the robot. After this chapter, the simulation will continue to be used to test alternate control and mechanical design solutions. Different mechanical designs can be rapidly tested in simulation without changes to the hardware.

As I was modeling and simulating the robot I noticed that the joint designs of Robot V differed from that of Robot III and the cockroach. This became clear as I was inputting ranges of motion of various joints. As described in future chapters, this discovery led to changes to the hardware and will lead to more changes in the future.

References

- 1 Schroer, R. T. Body Motions and Climbing Abilities of a Whegs Robot with Cockroach Comparisons, M.S. thesis. Case Western Reserve University, January 2004.
- 2 Kingsley, D. A., A cockroach inspired robot with artificial muscles, Ph. D. dissertation. January 2005.

Chapter 8

Implementation into Robot V Hardware

The inverse kinematics solver and gait controller were shown to successfully coordinate the joints and legs of a model of Robot V in simulation if it is supported such that its feet do not touch the ground. In this chapter, I explain how to implement these strategies to control the robot's hardware.

8.1 Robot V Sensors

Robot V is an ongoing project. The main goal of the Biorobotics lab is to make legged robots that can walk, climb, run and turn with the efficiency and style of animals. This is not a trivial issue because a complex neuromechanical problem must be solved. After building Robot V, Kingsley [1] demonstrated that it could walk using open-loop control with no sensors. Festo air muscles provide the advantage of passive joint compliance. When the actuators are pressurized the joints have stiffness that can passively reject disturbances and enable the robot to stand stably. This passive stability helps the robot stand by itself and walk without sensors. However, the robot needs sensory feedback for agility and adaptability in walking, running and climbing. Robot V has two types of proprioceptors: joint angles sensors and pressure sensors.

8.1.1 Joint Angle Sensor

Potentiometers were used to measure the joint angle positions in Robot III. A combination of flexible joint sensors and potentiometers are used to measure joint positions in Robot V. Kingsley chose flexible joint angles sensor manufactured by Spectra Symbol to measure some of the joints. This sensor is thin and flexible and its resistance varies as it is bent. They are mounted on the 1095 spring steel. The spring steel provides a restoring force to maintain the sensor's shape. This sensor is easy to mount firmly to most joints and its mountings are not likely to loosen with time as is the case with potentiometers.

One problem with this type of joint angle sensor is that its signal is nonlinear at the extreme positions. Another problem is that the rubber where the joint angle sensor is mounted exhibits hysteresis. The signal is not repeatable in cases of extending and flexing.

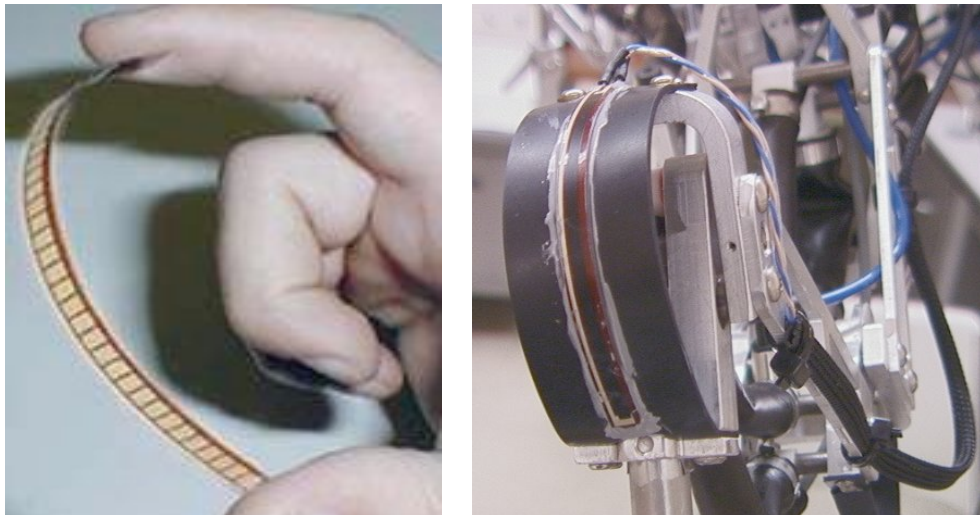


Figure 70 : Spectra Symbol flexible joint angle sensor is mounted on a rubber strip, which is then attached to each side of a joint.

8.1.2 Pressure sensor

It is possible to independently control position and stiffness (or position and force) of the joints in Robot V. However, active control of force or stiffness requires measurement of one of those parameters. Actuator force or stiffness can be calculated from actuator pressure, which is readily measured. Robot V has 24 joints and it needs a minimum of 48 artificial muscles (flexors and extensors) to drive it. Some joints have pairs of flexors or extensors, so there are actually many more actuators. No matter how many actuators are used for one joint, there are only two pressure lines for each joint, one for the flexors and one for the extensors. Robot V uses 48 Motorola MPX 5700 pressure sensors to measure pressure in each of these lines. These sensors were chosen because they are small and lightweight. See Kingsley for details [1]. A low pass filter is inserted into the tube to reduce the noise.

8.2 Robot V Hardware Setup

The hardware setup for Robot V is shown in Figure 71. The robot's off board controller resides in a PC running the RTLinux operating system. 96 channels are needed to control the robot: $24 \text{ joints} \times 2 \text{ (flexor and extensor)} \times 2 \text{ (inlet and exhaust)}$. The PWM duty cycle commands from the PC to the 96 valves are converted to voltage signals through the D/A board. The valves open and close to supply or exhaust high pressure air into the actuators. A regulator is used to supply high pressure air at 90 psi. The antagonistic pairs of flexor and extensor actuators produce joint torques, which drive the robot.

The analog signals from the pressure sensors and joint angle sensors are digitized by the A/D board. The signals from the joint angle sensors vary from 0-5 Volts. The A/D board has 96 channels and 48 channels are reserved for angle sensors. The remaining 48 channels are available for the pressure sensors. The sensor analog signals are converted to integer numbers between 0 and 255. The control system in the PC reads the joint sensor signals and compares them with the desired joint angles. The control system has a two level hierarchy. I developed the high level controller, which calculates desired joint angles corresponding to the desired foot paths and gait. Brandon Rutter developed the low level controller, which communicates with the hardware through the A/D boards. The angle sensor readings are compared with the desired joint positions. The errors are compensated by the PC controller.

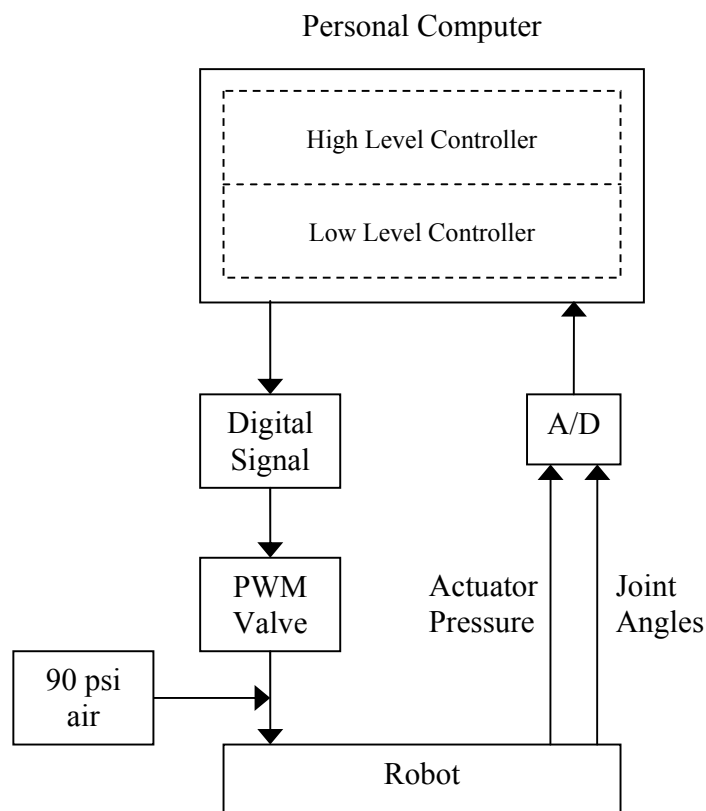


Figure 71 : Hardware setup for Robot V.

8.3 Initial Implementation of Leg Control

The implementation of the controller in the robot is a very interesting but challenging process. The main problem is to develop an adequate control system. A secondary problem is how to measure the robot's performance to show that one method is better or worse than another method. Intuitively when we see a robot walking, we can judge whether it is walking well. This is a subjective measure that depends on case by case and person by person. It is not trivial to define metrics to quantify robot performance. The basic metrics are joint angle, foot path and velocity errors. These metrics are used in the following experiments to measure the performance of the robot.

8.3.1 Initial implementation of Leg Control

The control methods for individual legs are explained in Chapter 4. Each leg of Robot V is tested separately before the gait controller is tested. Each individual leg has its own trajectory. This trajectory is designed within the workspace of each leg. In cockroach the swing path is not in a plane, but for convenience of finding swing paths, all robot foot paths are designed in a plane. The desired joint angles corresponding to the desired foot path are found using my neural network inverse kinematics solver. A proportional controller for each of the leg's joints tries to follow these desired joint angles. Figure 72 shows desired and actual joint angles for the right middle leg of the robot in one experiment. The robot is trying to track the desired joint angles, but there are larger errors particularly in the FT joint and these errors lead to larger errors in foot trajectory.

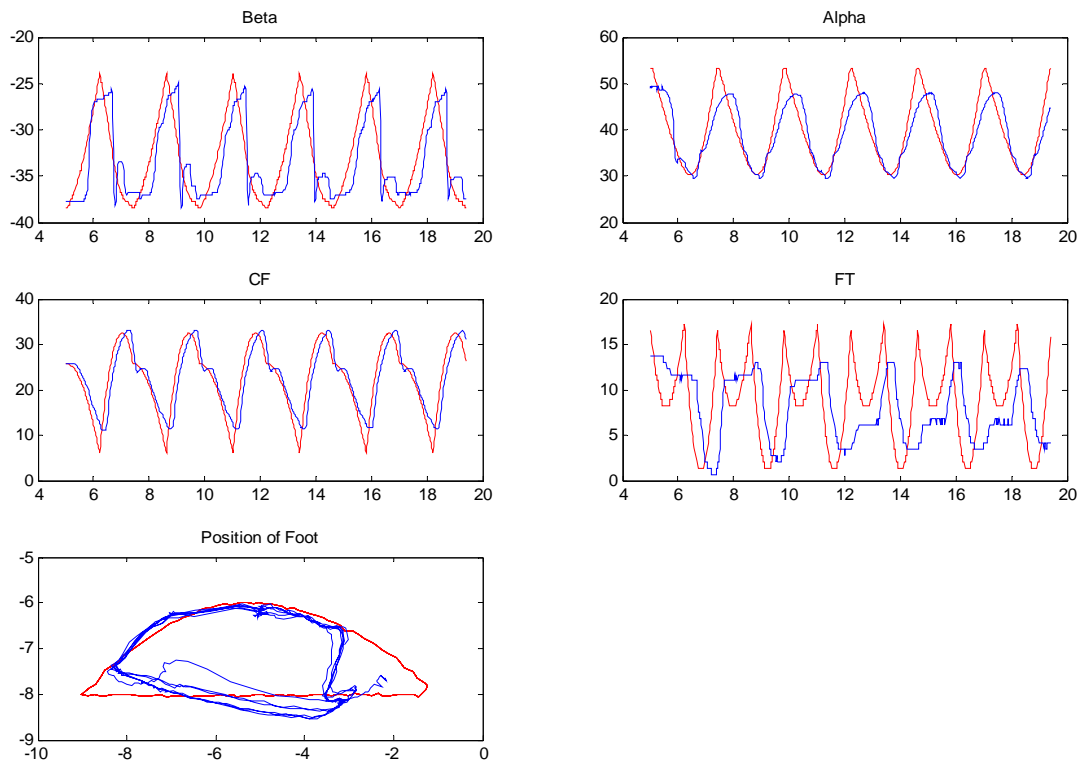


Figure 72 : Initial implementation of individual leg control for right middle leg in air walking. The Beta, Alpha, CF and FT joint angles are shown as well as the foot path.

8.4 Initial Implementation of Gait Control

After initial implementation of individual leg controls, the coordination of legs is tested. To isolate external forces, the robot is supported so that its feet do not touch the ground. The Cruise controller produces the gait pattern and the foot paths are calculated as described in section 5.2. In this case the desired joint angles are produced off-line. These joint angles are applied to the robot after verification in simulation.

8.4.1 Air-walking and Supported walking

After implementation of individual leg control, Cruse control is implemented into Robot V in case of air walking. Figure 73 shows the air walking test of Robot V. The desired paths are generated by my gait controller offline. The neural network generated the joint angle commands which correspond to the desired path. The joint position commands are saved in a formatted file. This puppet file is read before Robot V starts to move. Robot V is tested in air walking. Some joints can not follow the desired joint angles due to ROM limits. Figure 74 shows the results of air walking of left front leg. The red and blue lines represent the desired and actual joint angles. The CF joint can not reach the maximum joint angles. There are two reasons why the joint angles can not reach the desired joint angles. The first reason is the measured ROM is not correct. The second reason is the air system. We have 96 channels for inlet and outlet valves. Air is supplied to all actuator simultaneously. A joint may reach the maximum joint angle when it is tested separately, but when all joints are tested at the same time, air is not supplied to all joints equally.

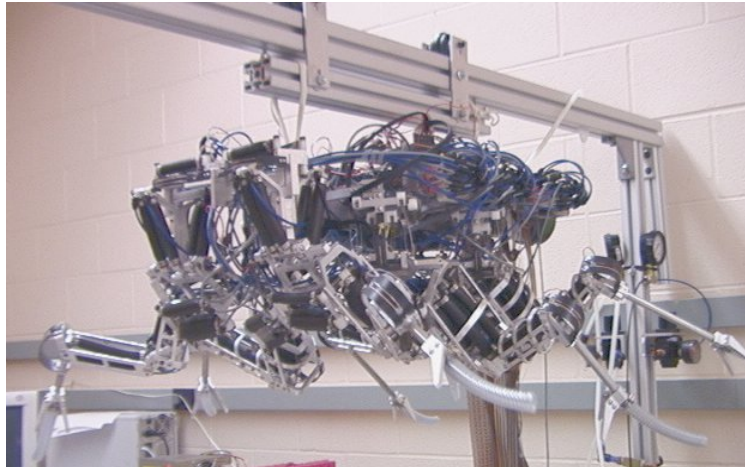


Figure 73 : Air walking test of Robot V

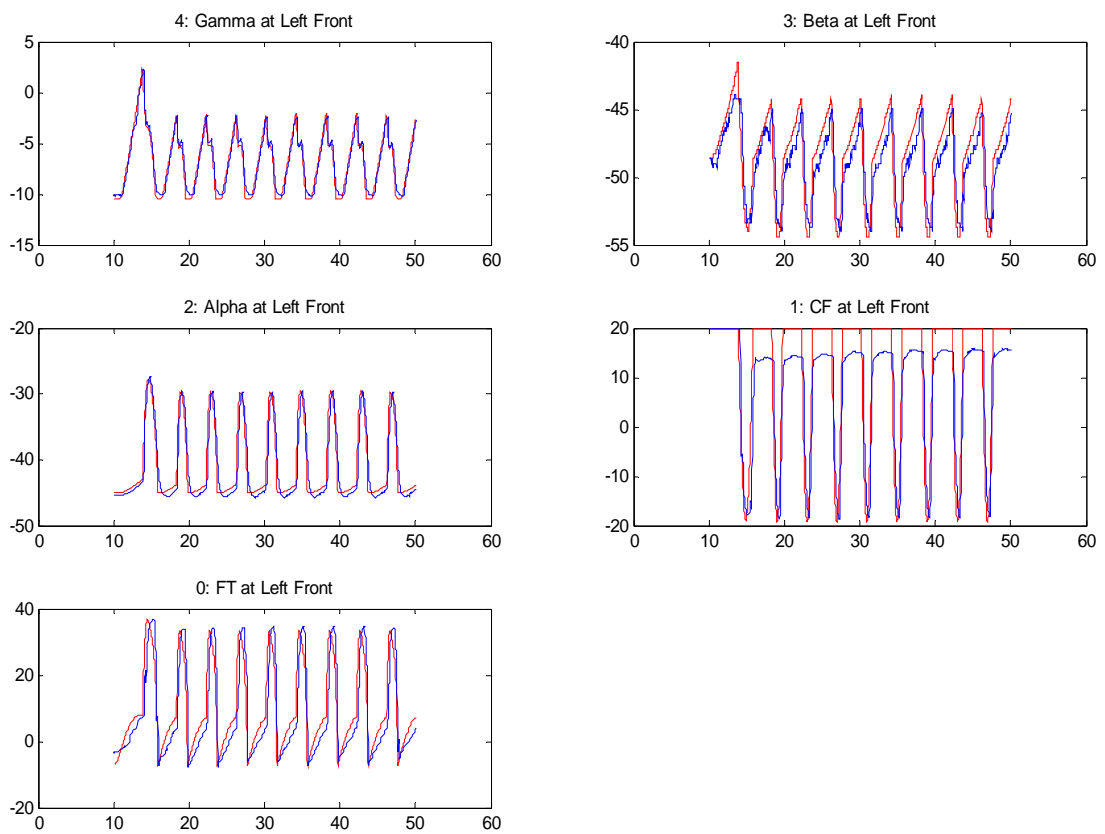


Figure 74 : Joint angle motions vs. time for left front leg during air walking

During ground walking attempts, Robot V has many problems. It had to overcome its weight and then walk. We decide to reduce its apparent body weight by partially supporting the body during the walking. The robot is lowered until the foot can contact the ground. Robot V's supported walking is fairly successful. The plastic tube for the foot reduces foot slipping during walking. Figure 76 shows the results for the left front leg in a supported walking test. Beta and CF joint are most affected by ground reaction forces. We tested supported walking with and without claws. The metal claws help to prevent the CF joints from touching the ground, but because the contact area of the metal claw is a point, the metal claw slips on the ground. The plastic tube foot provides friction to reduce slipping, it also sometimes drags on the ground and prevents Robot V from walking.

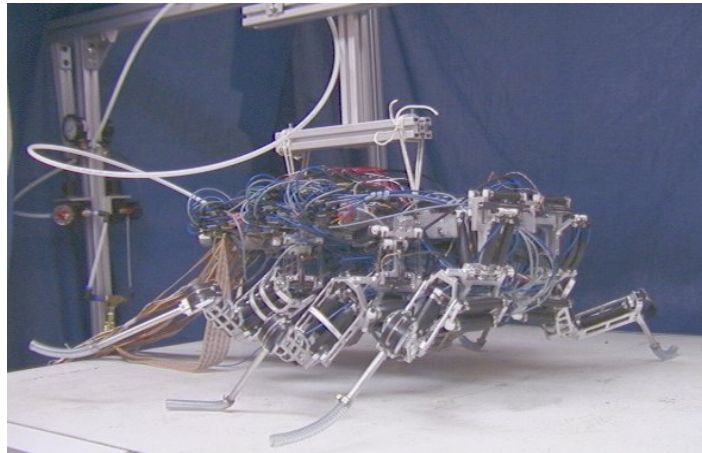


Figure 75 : Supported walking test of Robot V.

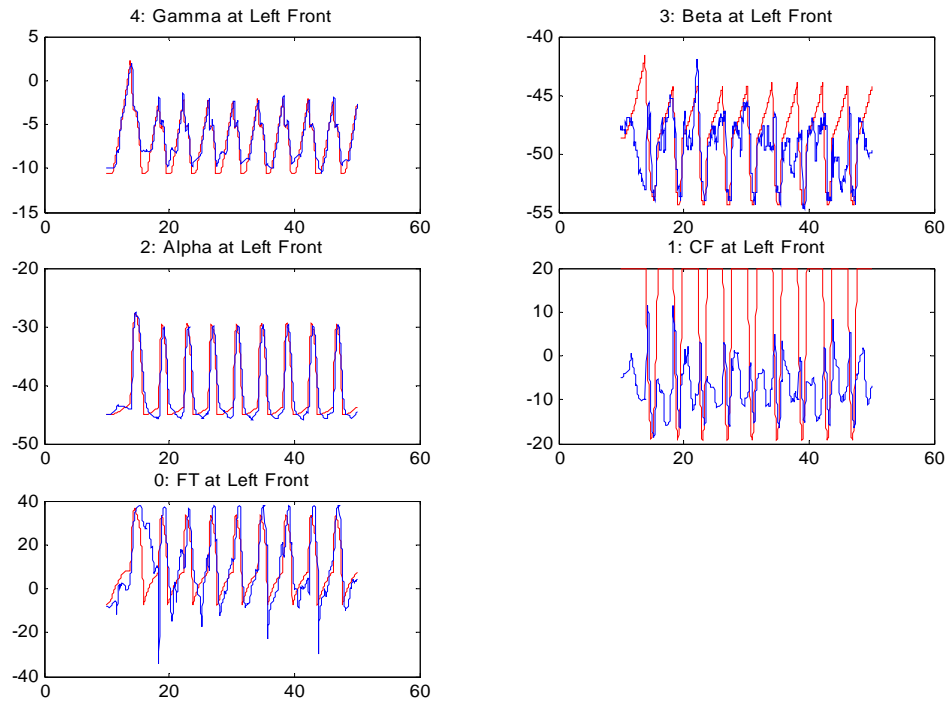


Figure 76 : Joint angle motions vs. time for left front leg during supported walking.

8.5 Robot V Problems and Solutions

8.5.1 Parallel Joints in Front Legs

The front leg has 5 DOF. The Gamma and Beta joint axes intersect at one point. The initial front leg designs in Robot V have two parallel joints. Movement of the Gamma joint affects the Beta joint and vice versa. Figure 79 (a) shows these parallel joints in the right front leg. The Gamma (γ) and Beta (β) actuators are both attached the robot's frame. The range of motion of each joint was within design specifications when

checked separately. However, the Gamma and Beta joints interfered with each other. Figure 77 shows poor joint tracking caused by interference between the Gamma and Beta joints during air walking. The red and blue lines represent the desired and actual joint angles, respectively. Figure 77 (a) shows the Gamma joint in the left front leg. The Gamma joint interferes with the Beta joint, and conversely, the Beta joint also interferes with the Gamma joint. The restricted Beta and Gamma joints hold the front legs tightly during walking. These restricted joints maintain the body high enough to walk stably. But as shown in Figure 78, the actual foot paths could not follow the desired paths. Due to the limitation in joint ranges of motion, the right and left front legs could not follow the desired paths.

To fix this problem, Kingsley redesigned the robot's front legs. To separate the two joints, the actuators for the β joint should be attached the Gamma joint segment rather than to the robot's frame. The revised serial joints are shown in Figure 79 (b). Two beams (highlighted in yellow) are added to the leg on the distal end of the gamma joint to support actuator attachment points for the Beta joint. The Beta joint is independent of Gamma joint rotation.

Figure 80 shows desired and actual joint angle motions of left (a) and right (b) front legs after redesign of the Gamma-Beta joints. The actual joint angles follow the desired angles much more closely than with the parallel joint design (Figure 80). The serial joint design permits a much greater range of motion.

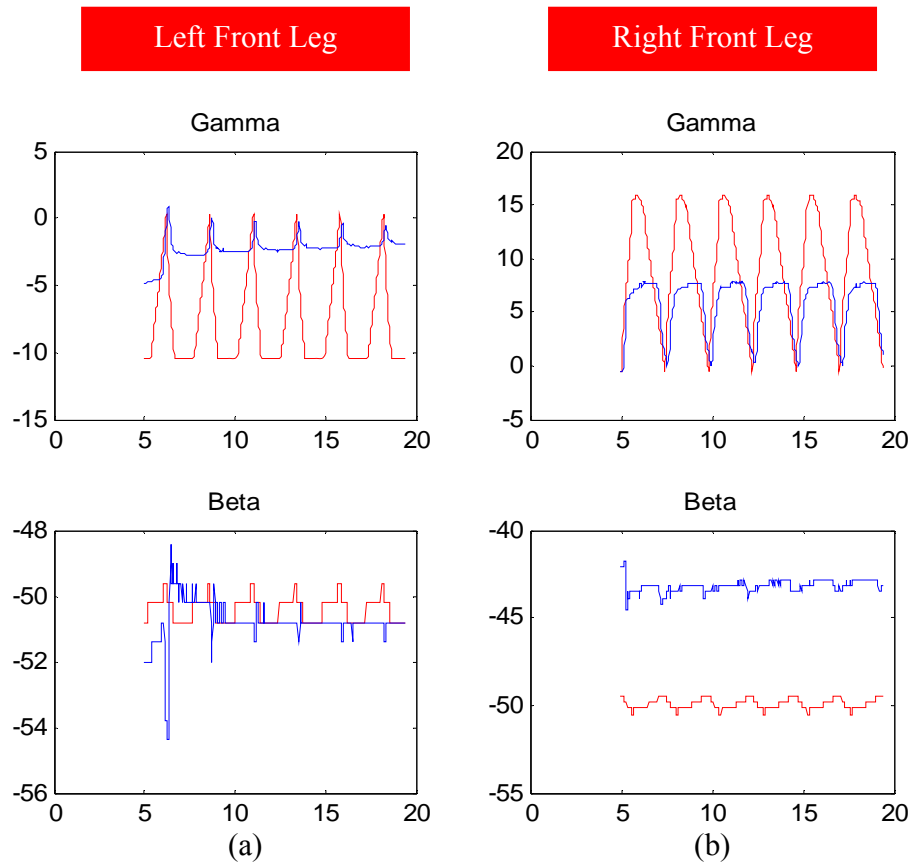


Figure 77 : Limited range of motion in serial front leg design in air walking.

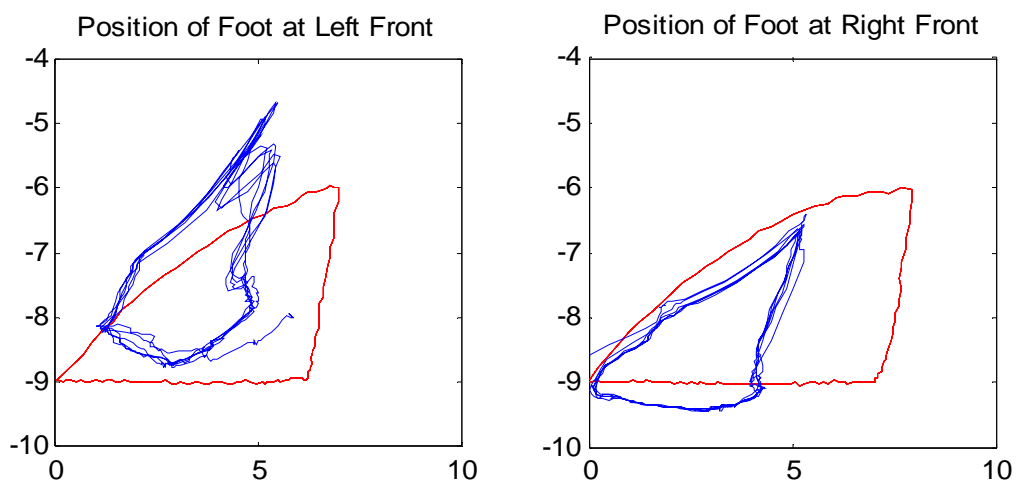


Figure 78 : Desired and actual foot paths of left and right front legs in air walking

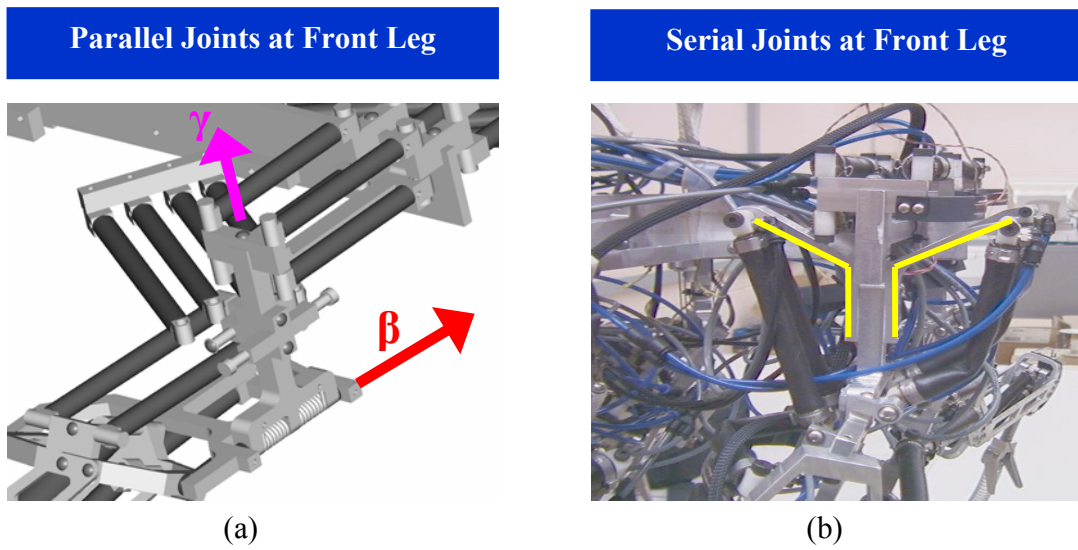


Figure 79 : Parallel and serial joints designs in right Front Legs. (a) parallel gamma and beta joint designs. (b) revised serial gamma and beta joint design.

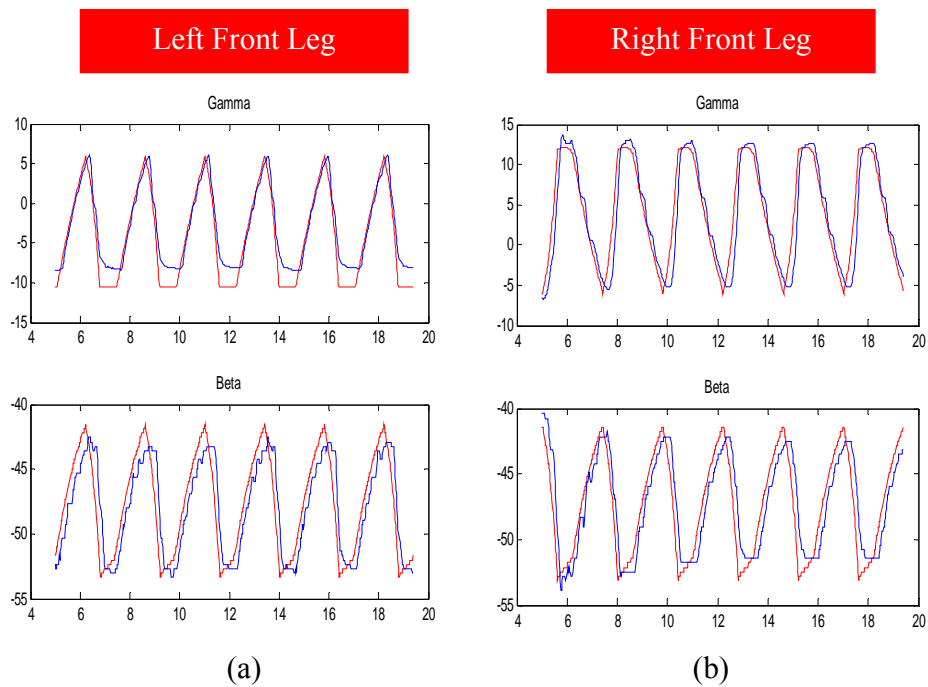


Figure 80 : Desired vs. Actual Joint Angle motions for air walking after design revision. (a) joint angles in left front leg. (b) joint angles in right front leg.

8.5.2 Non-symmetric Joint ROM and Sensor Tuning

The joints in Robot V have smaller ranges of motion than those in cockroach. In fact, the selected joint ranges of motion for Robot V were reduced to ease its design. The design ranges of motion (ROM) were chosen similar to those in Robot III, which were in turn chosen based on those used by cockroach in walking and climbing. However, for a number of reasons, the ROMs of the joints in Robot V are out of the desired ROM. Both mechanical changes to the joints and inaccurate sensory signals narrow the joint ROM. The Festo actuator ends were replaced with smaller, lighter plastic ends. The ends sometimes become detached and have to be replaced. When an actuator is replaced it must be exactly the same length as the original or the ROM is changed. Therefore, the ROMs of the joints are changed every time an actuator is replaced. Also, in the initial calibration of Robot V, all measurements were made in its right side and I assumed the left side was the same. Figure 81 shows joint measurement for Robot V. The Gamma rotation is out of paper. Beta rotation is perpendicular to the body center line. Alpha rotation is along the forward direction. CF joint is measured from the perpendicular line to Coxa segment. FT joint is measured from the perpendicular line to Femur segment. Table 10 shows the range of motions of all legs before joint sensor fixing and tuning.

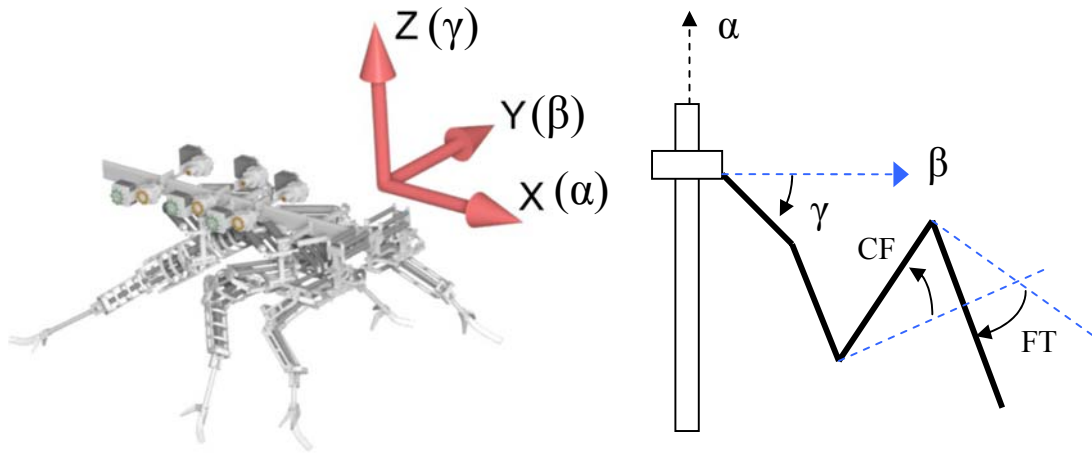


Figure 81 : Joint angle measurement for Robot V

JOINT	DESIRED ROM	Initial Measured ROM	Final ROM before Tuning	
			Left Side	Right Side
Front				
GA	-52.16 – -12.16	-69.16 – -8.16	18.16 – 42.16 *	-35.16 – -6.16
BE	-80 – -50	-73 – -32	-82 – -43	-74 – -36
AL	20 – 60	12 – 50	-49 – -13	18 – 40*
CF	-20 – 30	-21 – 20	-28 – 20	-22 – 48
FT	-30 – 45	-28 – 43	-34 – 40*	-15 – 48*
Middle				
BE	-80 – -50	-88 – -40	-80 – -40	-76 – -23*
AL	20 – 60	27 – 42	-35 – -23*	27 – 37*
CF	-10 – 40	3 – 41	-40 – 0	5 – 41
FT	-30 – 45	-23 – 46	-50 – 14*	-40 – 55
Rear				
BE	-40 – 0	-46 – -25	-46 – -11*	-43 – -21
CF	-10 – 40	-8 – 39	-29 – 22*	-5 – 38*
FT	-30 – 45	-37 – 54	-37 – 38*	-36 – 40*

Table 10 : Minimum and maximum range of motion of all joints. *less range of motion than desired.

The desired ROM represents the initial design criteria for all joints. The initial measured ROM means the range of motion of all joints measured by Kingsley when he initially fabricated Robot V. The final actual ROM before tuning means the ROM before

fixing and tuning. Note that the left and right sides were designed with the same ROM, but the result is that they are different. After minor design changes and fixing broken actuators, this non-symmetry can not be avoided and it causes control problems. Even if the desired joint angles are the same, the actual joint angles can not follow the desired joint angles due to joint limits. This non-symmetric ROM can be fixed by changing the actuator lengths. All actuator's lengths are measured and cut and replaced.

In spite of changing the actuators, the ROM of joints are not symmetric. The reason for non symmetries is not the physical actuator problem but the joint angle sensor problem. The joint angle sensors needed to be calibrated individually.

The best way to fix this problem is calibrating each joint sensor. After calibration, we get the coefficient to convert the joint sensor digital signal to degrees.

$$\text{(sensor reading)} = b + m \times \text{(angle in degrees)} \quad (45)$$

where b and m are determined by calibration.

The first step is to measure the physical joint angles at the extreme positions and compare these numbers with joint angle sensor's signal after calculation. When the two joint angle readings match each other, the coefficients are tuned. Table 11 shows the final joint calibration coefficients after tuning. After tuning, the joint angle sensor signal becomes more reliable and it is matched with the actual physical joint angles.

Joint #	Joint	b	m
0	FT of Left Front	152.8622	-2.5276
1	CF of Left Front	159.7112	3.7690
2	AL of Left Front	220.1393	5.0000
3	BE of Left Front	280.1892	2.6643
4	GA of Left Front	176.9188	-4.5811
5	FT of Left Middle	215.8173	-0.7988
6	CF of Left Middle	247.0626	4.7567
7	AL of Left Middle	189.3786	5.3101
8	BE of Left Middle	375.0104	3.0569
9	FT of Left Rear	140.7046	-1.9579
10	CF of Left Rear	180.8099	3.1845
11	BE of Left Rear	193.0000	2.0400
12	BE of Right Rear	120.2049	-2.7883
13	CF of Right Rear	202.2528	-2.9453
14	FT of Right Rear	100.4080	2.7494
15	BE of Right Middle	-15.9808	-2.4808
16	Al of Right Middle	300.8938	-6.9366
17	CF of Right Middle	190.5376	-4.8032
18	FT of Right Middle	180.0186	1.0530
19	GA of Right Rear	125.9083	3.2006
20	BE of Right Rear	-17.6193	-2.9095
21	Al of Right Rear	340.3840	-4.0651
22	CF of Right Rear	156.2399	-1.7331
23	FT of Right Rear	115.6756	1.0667

Table 11 : Joint calibration coefficient after tuning.

Table 12 shows the new ROM of all joints after joint angle sensor tuning. After changing the length of actuators and calibration of joint angle sensors, the ROM is more reliable and the non-symmetries are reduced. With new range of motion, the workspaces are updated and the inverse kinematics are solved again. The neural network inverse kinematics solvers are trained based on the new ROM.

JOINT	DESIRED ROM	PREVIOUS ROM	FINAL ACTUAL ROM	
			Left Side	Right Side
Front				
GA	-52.16 – -12.16	-69.16 – -8.16	12.16 – 52.16	-52.16 – -12.16
BE	-80 – -50	-73 – -32	-75 – -35	-75 – -35
AL	20 – 60	12 – 50	-45 – -20	20 – 45
CF	-20 – 30	-21 – 20	-25 – 25	-20 – 30
FT	-30 – 45	-28 – 43	-40 – 35	-30 – 50
Middle				
BE	-80 – -50	-88 – -40	-80 – -40	-76 – -30
AL	20 – 60	27 – 42	-35 – -23	27 – 37
CF	-10 – 40	3 – 41	-45 – 0	5 – 40
FT	-30 – 45	-23 – 46	-45 – 40	-50 – 50
Rear				
BE	-40 – 0	-46 – -25	-50 – -25	-43 – -21
CF	-10 – 40	-8 – 39	-30 – 20	-8 – 34
FT	-30 – 45	-37 – 54	-35 – 40	-35 – 40

Table 12 : Range of Motion after Sensor Tuning

8.5.3 Actuator Buckling Problem

Festo actuators are lightweight and strong, but they have some disadvantages. One problem is that they are stiff in compression because they have relatively thick tubes. When they are inflated they contract in length. However, if they are not activated and an external force applies a compressive force on them they do not easily comply as do McKibben actuators. Instead, they are stiff and resist contraction until they suddenly buckle and offer little resistance. Buckling is a nonlinear phenomenon and it is difficult to predict or control so it is to be avoided. Figure 82 shows the buckling of two Festo actuators. To prevent actuator buckling and the subsequent control delay, a tensioning reflex was suggested by Brandon Rutter [2]. The minimum actuator inflation pressure is calculated offline and added to the actuator pressure to prevent it from buckling. The tensioning reflex improved the tracking of joint angles. Figure 83 shows the desired joint

angles, joint angles without tensioning reflex and joint angles with tensioning reflex, all versus time for the FT joint in the right front leg during air walking. The delay in response to commands and tracking of joint angles are improved by the tensioning reflex. Figure 84 shows the error between the desired and actual joint angles of the right front leg without vs. with reflex. The tensioning reflex reduces the delay and improves the tracking performance.

Figure 84 shows the average joint angle errors for the right front leg during 5 cycles in air walking. The tensioning reflex reduces all joint angle errors in the right front leg. Figure 85 shows foot paths for several cycles of the right front leg. Improvement of all joint errors ensures that the foot path follows the desired foot trajectory more closely with the tensioning reflex than without it. The vibratory motion of the foot path is reduced and the tracking is improved at the sharp corners of the trajectory. Figure 86 shows the mean value and range of error of foot position. The range of errors is improved by the tensioning reflex as well as are the mean values.

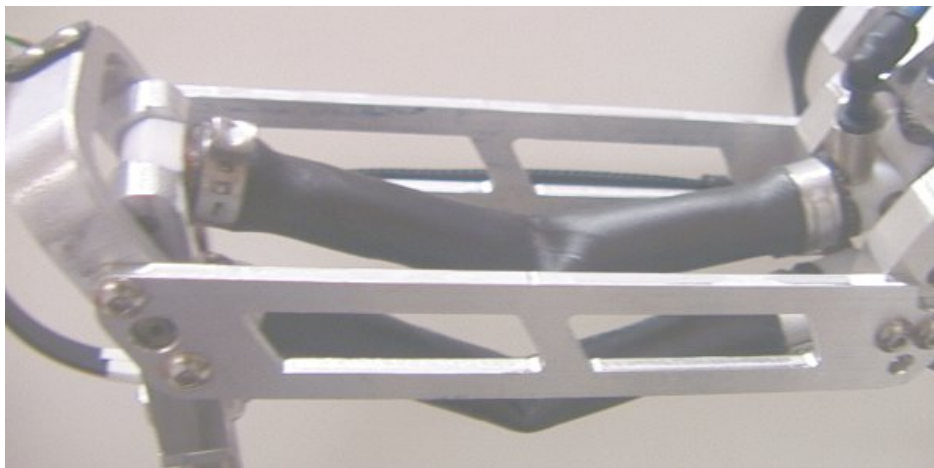


Figure 82 : Festo actuator Buckling Problem.

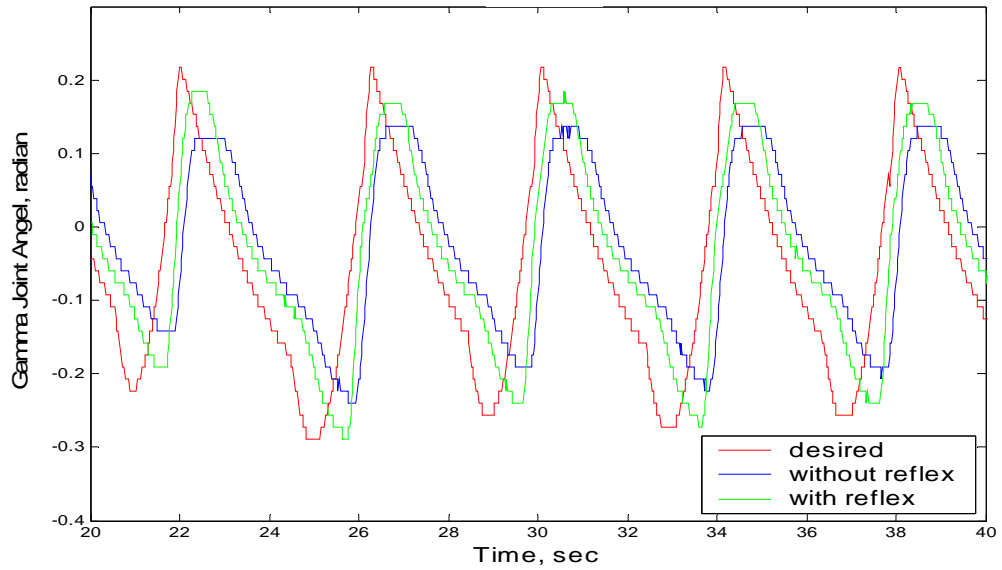


Figure 83 : FT joint angle motion vs. time in the right front leg. The red, blue and green lines represent the desired joint angles, joint angles without reflex and joint angle with reflex, respectively.

Average of Joint Angle Error of Right Front Leg with vs. without Tensioning Reflex

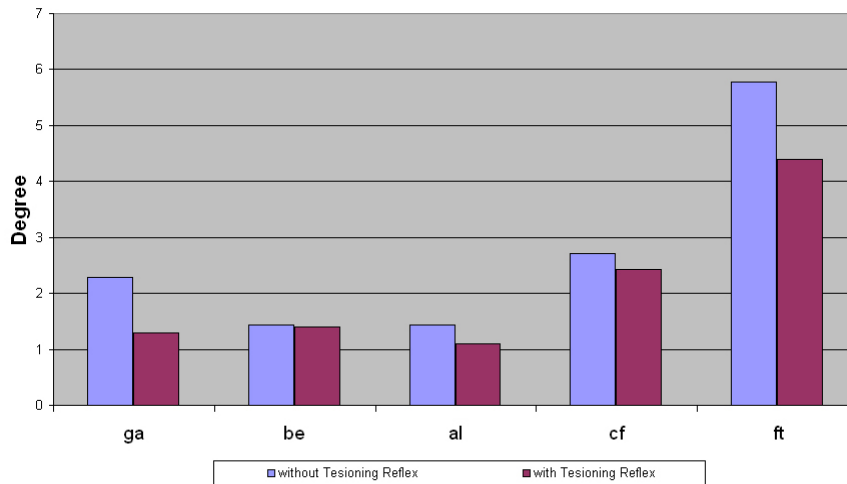


Figure 84 : Average Joint Angle Errors in Right Front Leg with vs. without Tensioning Reflex

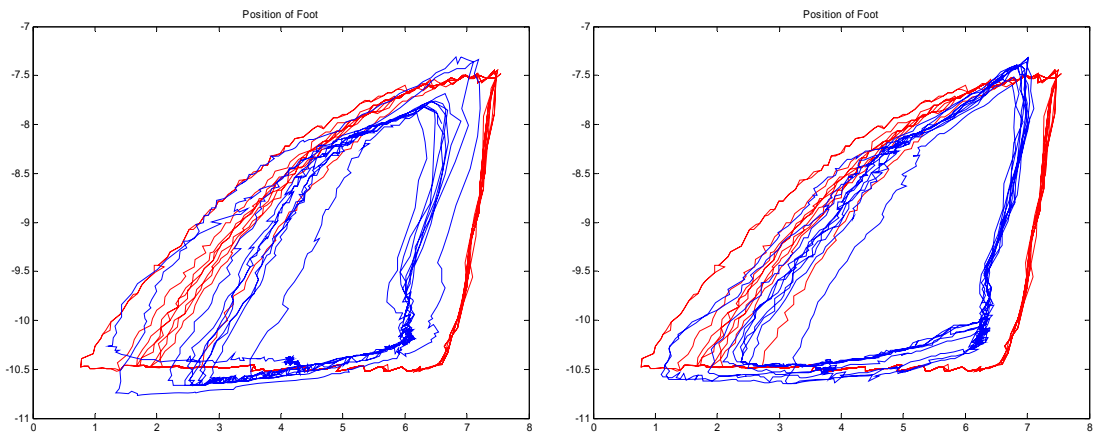


Figure 85 : Foot trajectory of Right Front Leg. (Left) without tensioning reflex (right) with Tensioning Reflex

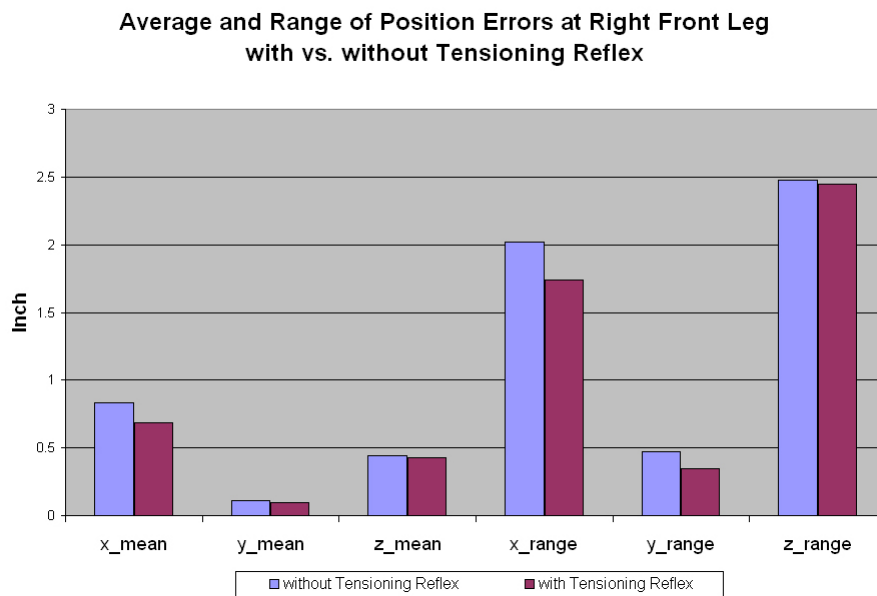


Figure 86 : Average and Range of foot Position Errors in Right Front Leg with vs. without Tensioning Reflex.

8.6 Conclusions

In this chapter, the joint angle sensor and pressure sensor are introduced. With sensors, the joint angles and the pressure are monitored for closed loop control. After the

joint angle sensors and pressure sensors are mounted, the initial tests of air and ground walking are done.

After initial experiments, mechanical design, sensor and actuator problems are found. The front leg was designed with parallel joints, and the front legs could not follow desired paths. The Gamma and Beta joints in the front legs interfere with each other. It limited the ROM of Gamma and Beta joints. These joints are revised with a serial design, which increased the joint ranges of motion.

Two problems were discovered and fixed that improved the practical ranges of motion of the joints. The left and right legs were found to have non-symmetric ROM for two reasons. The actuator lengths on the right and left side were not equal and the joint angle sensors need to be calibrated more carefully. When these problems were fixed the left and right side joint ROM became similar.

One of the main problems with using Festo actuators is that they buckle when compressed. Buckling causes sudden and uncontrollable joint movements. A tensioning reflex maintains minimum pressure in the actuators and prevents buckling. This reduced response time and joint angle errors.

Reference

- 1 Kingsley, Daniel. A Cockroach Inspired Robot with Artificial Muscles, Ph. D. dissertation. Case Western Reserve University, January 2005.

2 Jong-ung Choi, Brandon L. Rutter, Daniel A. Kingsley, Roy E. Ritzmann, Roger D. Quinn. A Robot with Cockroach Inspired Actuation and Control. Presented

Chapter 9

Comparisons with Cockroach and Robot III

Robot V is modeled after the cockroach, *Blaberus discoidalis*, but after applying my control strategies I have found that the posture and movement of Robot V are different than those of the cockroach. The standing posture is higher than that of the insect. Based on high speed video data, another big difference between cockroach and Robot V is that the cockroach's CTr (or CF) joint touches the ground and sometimes (not often) it even slips on the ground. When the CTr (or CF) joint of Robot V touches the ground, it prevents the robot from walking forward. In this chapter the differences between cockroach and Robot V will be discussed as well as the differences between Robot III and Robot V.

9.1 High Speed Video Analysis of Cockroach

Generally speaking, Robot III and Robot V are modeled after the cockroach, *Blaberus discoidalis*. Figure 87 shows the design process from the cockroach to the robots. In step (a) cockroach motions are recorded using high speed video in the Ritzmann Lab [1] and positions on its legs and body are painstakingly digitized. The head, tail and all joint positions are marked and then the movements are recorded using high-speed video. All positions are digitized frame by frame. In (b) a model is used to

display the raw digitized motion data. In (c) I extract the joint angle histories from the digitized data. The cockroach has 7 joint DOF in each leg. These joints are reduced in analysis so that the front, middle and rear legs have 5, 5 and 5 DOF, respectively. Gabriel Nelson [2] developed the algorithm for finding joint angles. In (d) I use the joint angle data as desired trajectories in a P controller for a dynamic simulation of the cockroach. In (e) and (f) Robot III and Robot V are designed and fabricated using the joint data.

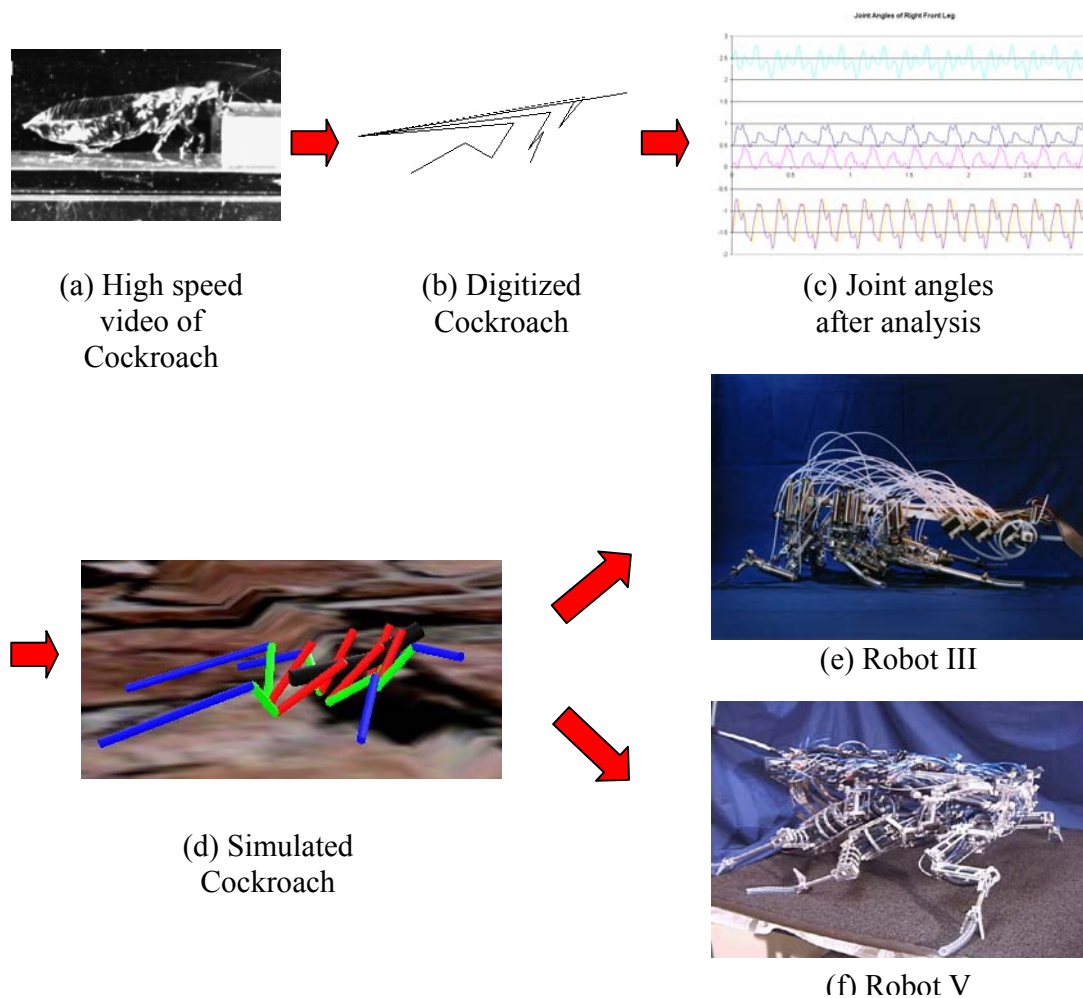


Figure 87 : Design process from the Cockroach to Robot III and Robot V. A cockroach is video taped and positions are digitized. The joint angles motions are found and these joint angles are used for the simulated cockroach. Robot III can use the joint angles from the digitization due to its physical similarity. The joint angles of Robot V are different from those of the cockroach.

9.2 Cockroach vs. Design of Robot III

Several assumptions are used to extract joint angles from the digitized cockroach data. The first assumption is that a cockroach leg has 5 DOF. There are actually 7 joint DOF in each leg. We neglect the Trochanter-Femur joint and the activation of the tarsus. The second assumption is that Coxa, Femur and Tibia segments lie in a plane. In reality, the CTr joint can move the Femur and Tibia segments out of this plane. The third assumption is that the Gamma, Beta and Alpha joint axes intersect at one point. These basic assumptions caused the differences between the real cockroach and the digitized cockroach. Figure 88 shows the front legs of the digitized cockroach and Robot III.

The cockroach is simulated using joint angles from digitization as the desired joint angles in a PD controller. All three leg pairs in the simulated cockroach have 5 DOF. The simulated cockroach can walk on a virtual ground using joint angles from the digitization process. Nelson found that the middle and rear legs' DOF could be reduced to 4 and 3 DOF, respectively, and still provide cockroach-like foot motions. For this reason, Robot III was designed with 5, 4 and 3 DOF on the front, middle and rear legs.

Robot III was designed based on the simplified, digitized cockroach. Figure 88 (b) shows the basic design of Robot III's front leg. The Gamma, Beta and Alpha joint axes meet at one point. Because Robot III has enough ROM in all of its joints, it can attain cockroach-like body posture with the simplified joint angles from digitization.

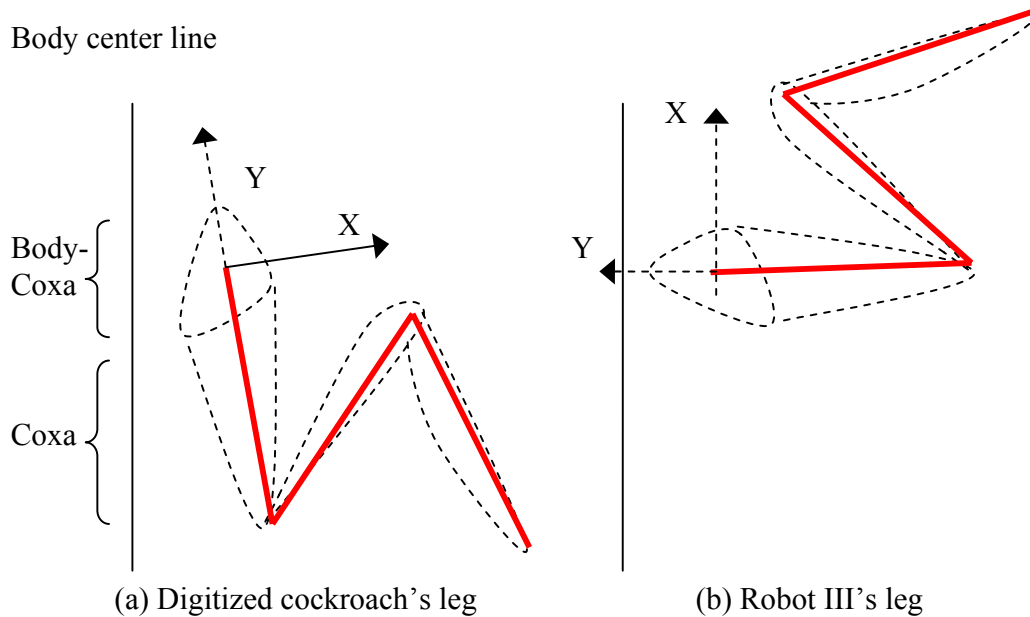


Figure 88 : Digitized cockroach vs. Basic Design of Robot III.

9.3 Cockroach vs. Design of Robot V

Robot V's legs were designed based on what we thought was the anatomy of the BC joint presented by Dresden and Nijenhuis [3]'s (it is now not clear that our understanding of this anatomy was correct). Figure 89 shows the supposed anatomy of the BC joint and the basic design of Robot V. In Figure 89 (a), the Gamma joint of the BC joint rotates about point I along an axis perpendicular to the plane of the figure. The Beta joint in the front leg rotates about the Y axis. The Gamma rotation and Beta rotation meet at point I. The Alpha joint rotates about the X axis. The Alpha and Beta joint intersect at point II. Mechanically this BC joint can be designed easier than that in Robot III. In Robot III, Coxa, Femur and Tibia are in a plane. Kingsley [4] designed the front

leg of Robot V as shown in Figure 89 (b). At point I, the Gamma joint and Beta joint intersect. At point II, the Beta and Alpha joints intersect.

The main difference between Robot III and Robot V is the Coxa segment. The Coxa segment of Robot V is pre-inclined. This inclination is based on the supposed cockroach anatomy.

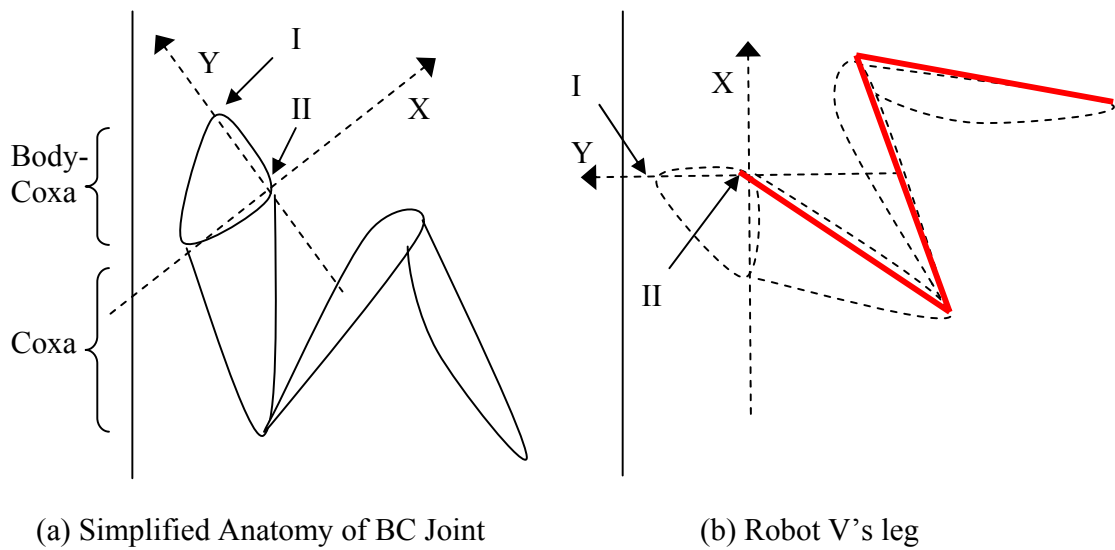


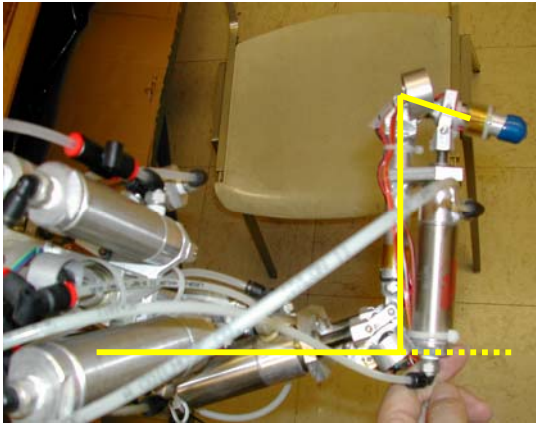
Figure 89 : Cockroach vs. basic design of Robot V

9.4 Comparison with Robot III

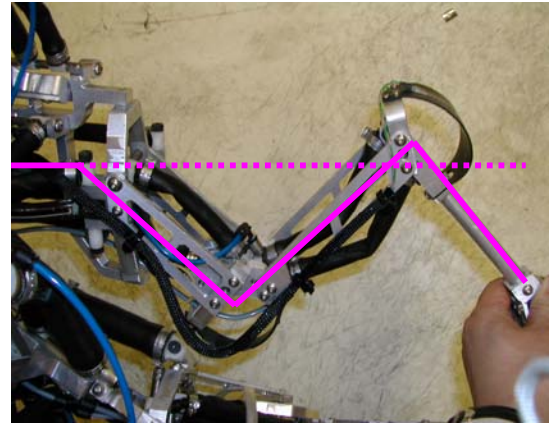
Because Robot III's joints have a wider range of motion than those in Robot V, Robot III can demonstrate cockroach-like body postures with joint angles from digitized cockroaches. The differences between Robot III and Robot V are discussed in terms of the initial configuration of the Coxa segment and the resulting differing CF Beta axis of rotation.

9.4.1 Beta and Alpha Axes Differ in Robot III and Robot V

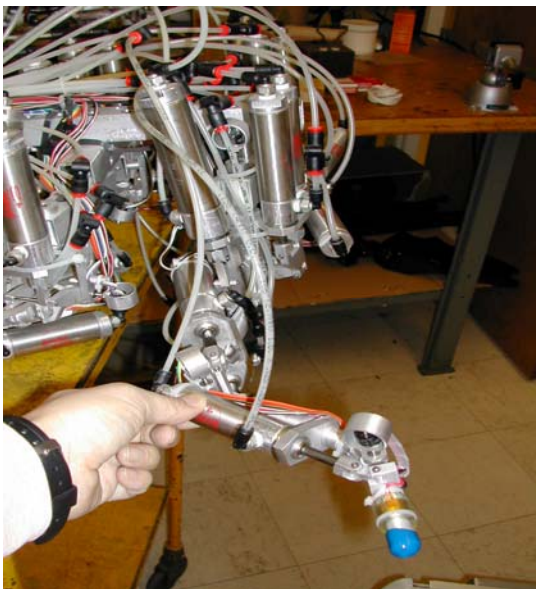
Before we compare the designs of Robot III and Robot V, the initial joint configurations of their legs should be defined. Nelson [2] and Choi [5] simulated cockroach using joint angles from digitization. The assumed initial joint configurations of those model animals agree with that of Robot III. All legs are stretched out away from the body in the y or $-y$ direction in their initial (zero) configuration. Figure 90 (a) shows the top view of the right front leg of Robot III where Gamma, Beta, Alpha, CF and FT joints take the values $0, 0, 0, +90, -90$ degrees and (c) shows the side view of the right front leg. Due to their design differences the initial configurations of Robot III and Robot V are different. Figure 90 (b) shows a top view of the right front leg of Robot V where the Gamma, Beta, Alpha, CF and FT angles take the values $0, 0, 0, +90, -90$ degrees. Figure 90(d) shows the side view of the right front leg of Robot V. In the initial state, the big difference between Robot III and Robot V is in the Coxa segment. The Coxa segment of Robot V is inclined about the z axis by 32.16° . So even if the joint angle commands are the same, the resulting joint configurations of Robot III and Robot V are different. In its initial state, the front foot position of Robot V is located behind its BC joint unlike Robot III. Robot V must use different joint angle trajectories to move its legs like a cockroach, but its ROM is not large enough to do this. Figure 90 (c) shows the side view of Robot III. Robot III can reach the zero Beta joint angle position but In Figure 90 (d), due to the joint limit, the Beta joint of Robot V can not reach the zero position.



(a) Top View of Robot III



(b) Top View of Robot V



(c) Side View of Robot III



(d) Side View of Robot V

Figure 90 : Right Front Legs of Robot III and Robot V. (a) initial configuration of Robot III. (b) initial configuration of Robot V. (c) side view of Robot III and (d) side view of Robot V.

The pre-inclined angle of the coxa segment has an important ramification in the BC joint axes of rotations. The result is that the Beta and Alpha axes are different in Robot III and Robot V. Robot III's Beta rotation axis is along the Coxa segment whereas

the Beta rotation axis of Robot V is perpendicular to the body center line. Figure 91 shows the Beta rotation axes in Robot III and Robot V given the same Gamma rotation. The sequence of Euler angles is Gamma, Beta and then Alpha from proximal to distal. In Robot III, the position of the CF joint is not affected by Beta rotation because the Beta axis is along the Coxa segment. However, in Robot V, the position of the CF joint is lowered toward the ground when Beta rotates in the negative direction. That's why Robot V's CF joint is more likely to contact the ground during walking. Note that this was not a problem in simulations of Robot V because the feet are the only points that can contact the ground in the model. The CF joints can pass through the ground without causing a ground reaction force.

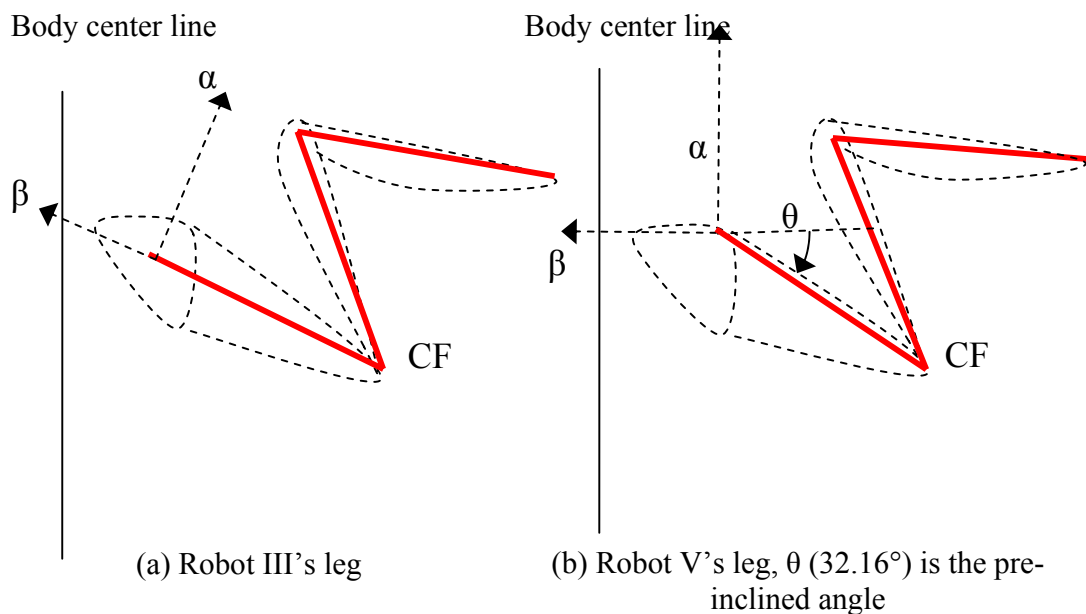


Figure 91 : Different Beta rotations between Robot III and Robot V

9.4.2 Comparison of Forward Swing

Due to their different designs, the foot positions of the front legs of Robot III and

Robot V are different even if the same joint angles are input. When $\text{Beta} = -45^\circ$ and $\text{Alpha} = 45^\circ$, the foot position of Robot III is in front of its BC joint (Figure 92 (a)). Because of its inclined Coxa segment, the foot position of Robot V is behind the BC joint with same joint angles. The foot of Robot III can swing forward with a small Beta rotation because the foot position is already in front of the BC joint. To swing its front leg forward, Robot V uses the Beta joint extensively and the CF joint touches the ground often. Because Robot III's Coxa segment is stretched out, when the Beta joint rotates the CF joint does not touch the ground. When Robot V's Coxa segment rotates about the Beta axis, its CF joint touches the ground because its coxa segment is inclined. That is why the Coxa joint of Robot V touches the ground often.

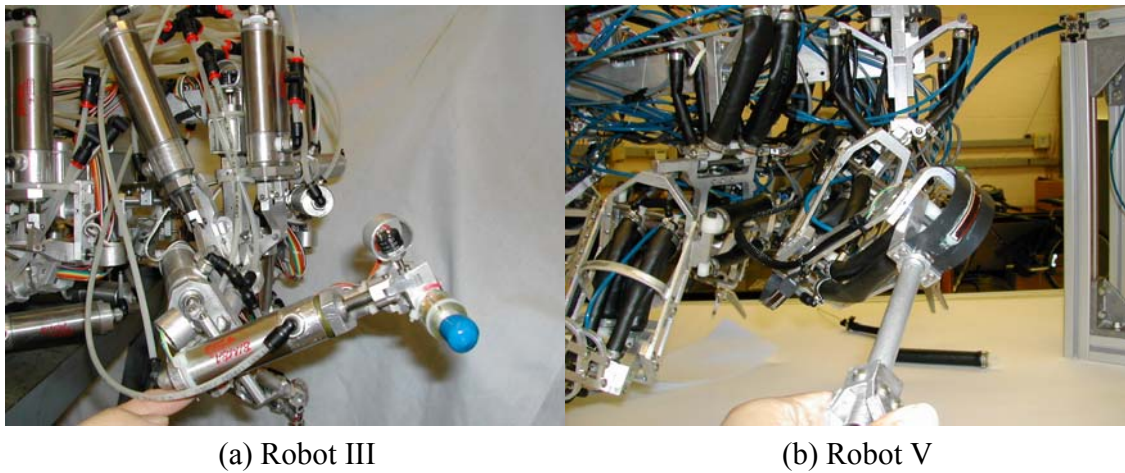


Figure 92 : Foot Position of Robot III and Robot V when $\text{Beta} = 45^\circ$ and $\text{Alpha} = 45^\circ$

9.4.3 Comparison of ROM of Coxa Segments

In terms of design, the main difference between Robot III and Robot V is the design of the Coxa segment. The Coxa segment of Robot V is separated into two parts. The Coxa I segment rotates about the z axis (Gamma rotation) first and rotates about new y axis (Beta rotation). The Coxa II segment rotates about the x axis (Alpha rotation).

In Figure 93, the difference between Robot III and Robot V is apparent when their Coxa segments are compared. As mentioned before, the Coxa joint of Robot V is designed as an inclined segment. But the Coxa segment of Robot III stretches outward along the y axis in the initial state. For more similarity with the insect, the Coxa joint of Robot V should be designed like the red line in Figure 93 (b). During walking, the Gamma joint of the Coxa segment of Robot III has the ROM from 0° to 90° . But due to the actuator limitations, the Coxa joint of Robot V has the ROM from -52.16° – -12.16° . This limitation forces Robot V to maintain an upright posture. The influence of the upright posture will be discussed later when energy efficiency is discussed.

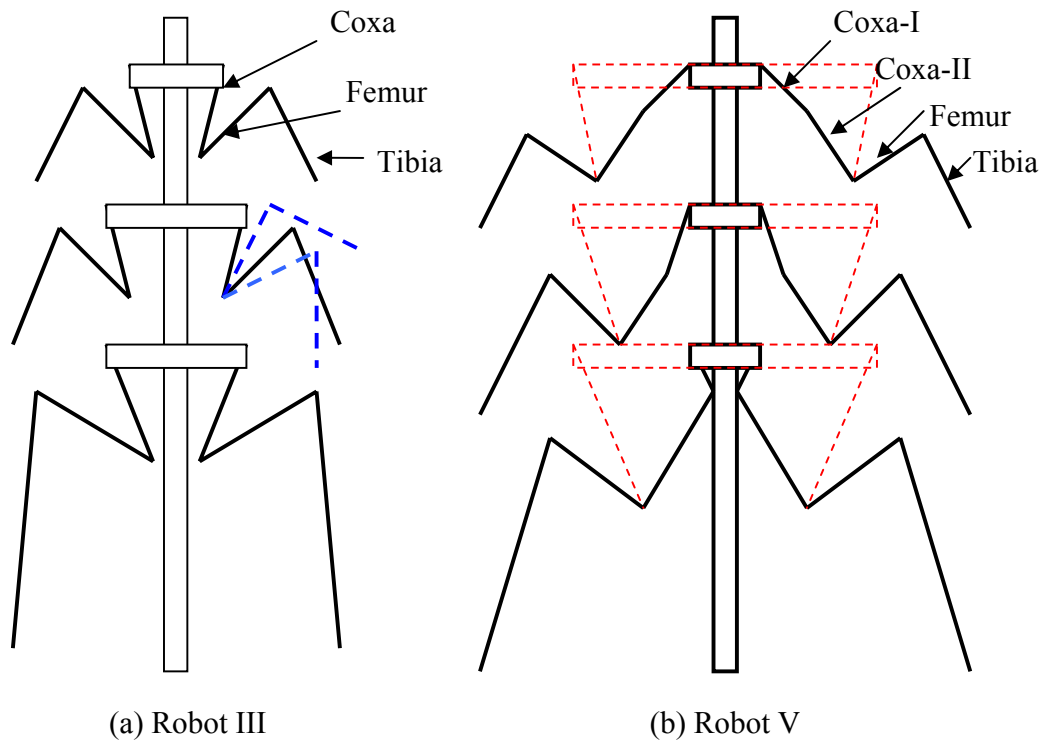


Figure 93 : Physical Difference between Robot III and Robot V. (a) bottom view of Robot III. (b) bottom view of Robot V. In (b), the red lines show a design more similar to Robot III.

9.5 Comparison with Digitized Cockroach

The joint angles from digitization of cockroach movement are compared with the joint angles for Robot V. Joint angle synchronization and comparison between cockroach and Robot V will be explained. These comparisons are useful to understand what the differences are between the animal and the robot. For the next generation cockroach-inspired robot, these comparisons tell what design factors are important.

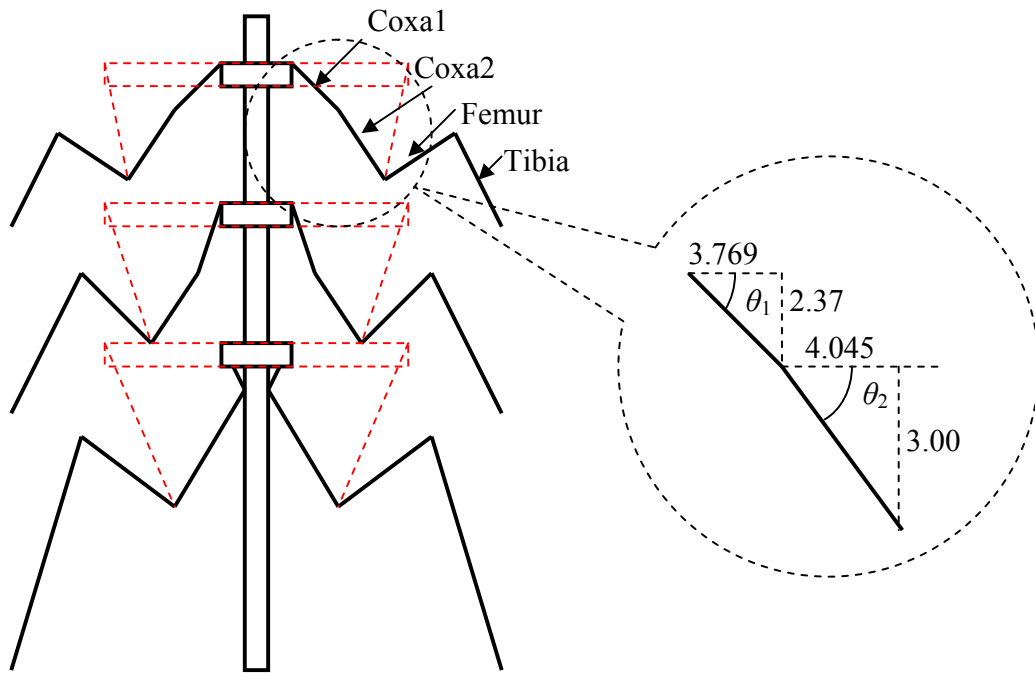
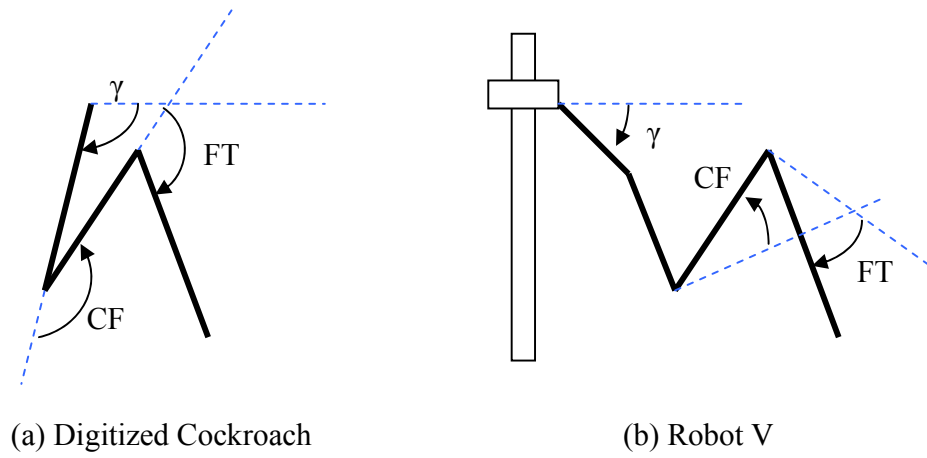
9.5.1 Joint Synchronization between Digitized Cockroach and Robot V

Due to the physical differences between the modeled cockroach (and Robot III) and Robot V, their joint angles are measured with a different convention. To compare joint angles, synchronization between them is necessary. Figure 94 shows the joint angles' origins and the directions of positive rotations for Robot III and Robot V. The Gamma joint in Robot III is measured from the y axis. Similarly the Gamma joint of Robot V is measured from the y axis. But when the Gamma joint is zero, the Coxa segment of Robot V is pre-inclined. So to compare the Gamma joint angles of Robot III and Robot V, $\theta_1 = 32.16$ (negative rotation, in Figure 94 (c)) should be subtracted from the Gamma joint of Robot V. The CF joint angle of Robot III is measured from a line extended from the Coxa segment. But the CF joint angle of Robot V is measured from a line perpendicular to the Coxa-II segment. The CF joint angle of Robot III equals the CF joint angle of Robot V + 90°. The FT joint of Robot III is measured from a line extended from the Femur segment. The FT joint of Robot V is measured from a line perpendicular to the femur segment. The FT joint angle of Robot V should be subtracted by 90° to compare with the FT angle of Robot III.

$$\text{Gamma joint angle of cockroach} = \text{Gamma joint angle of Robot V} - 32.16^\circ$$

$$\text{CF joint angle of cockroach} = \text{CF joint angle of Robot V} + 90^\circ \quad (46)$$

$$\text{FT joint angle of cockroach} = \text{FT joint angle of Robot V} - 90^\circ$$



(c) Pre-inclination of Gamma joint of Robot

Figure 94 : Joint Angle Synchronization between Cockroach and Robot V. (a) joint angle measurement of cockroach. (b) joint angle measurement of Robot V. (c) pre-inclination of Gamma joint of Robot V.

9.5.2 Comparison of Joint Angles

After synchronization, Figure 95 to Figure 107 show the joint angles of the front, middle and rear legs. Note that the range of motion tables appearing in Chapter 8 have already been synchronized so that all angles are measured using the Robot V convention. Figure 95 shows the desired, robot air walking, robot supported walking and animal joint angles of right front Gamma joint. Due to the design difference, the Gamma joint motion of the cockroach is bigger than that of the robot. The cockroach's coxa rotates inside more than the robot's leg does. Figure 96 shows the desired, robot air walking, robot supported walking and animal Beta joint angles. The cockroach's Beta joint angle moves back and forth around the zero position. The Beta joint of Robot V rotates in the negative direction to contact the ground. The range of motion of the robot's Beta joint is $-60 \sim -40$. Figure 98 shows the desired, robot air walking, robot supported and animal joint angles for the right front CF joint. Due to the different orientation of the Coxa segment, to get a similar posture, the CF joint angle of cockroach is larger than that of Robot V. The large CF joint angle of cockroach means the Femur segment approaches the Coxa segment. The CF joint angle in the robot changes from $0 \sim 30$ degrees. It prevents the Femur segment from approaching and colliding with the Coxa segment. The CF joint angles of Robot V are smaller than those of cockroach. The Femur segments of cockroach and Robot V have similar orientations as shown in Figure 94 (a) and (b). Figure 99 shows the desired, robot air walking, robot supported walking and animal walking joint angles of the right front FT joint. The cockroach's FT joint angles are larger than those of Robot V. The cockroach uses its FT joint extensively to get maximum reach. The joint angle motions of the front legs of cockroach and the robot illustrate major differences between

cockroach and Robot V. The design difference causes corresponding differences in joint angles and movement strategies.

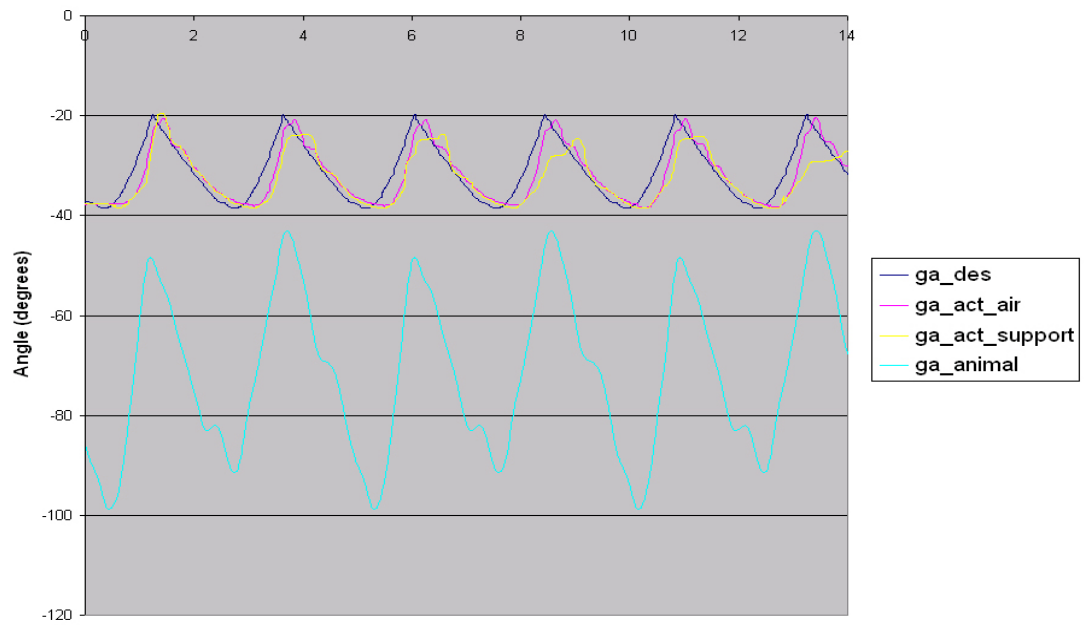


Figure 95 : Right Front Gamma Joint.

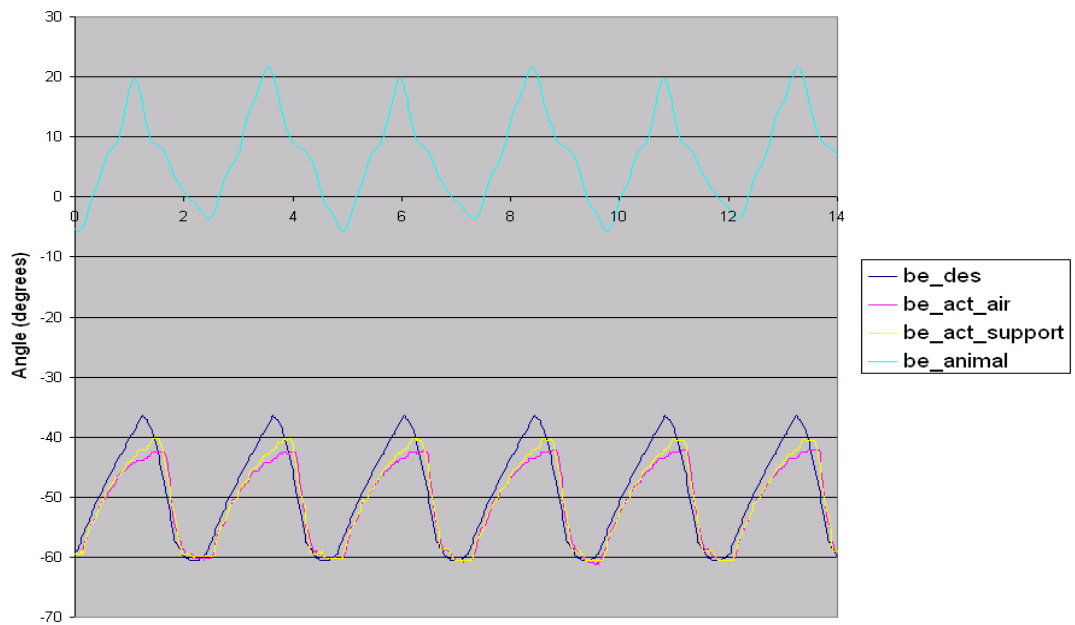


Figure 96 : Right Front Beta Joint.

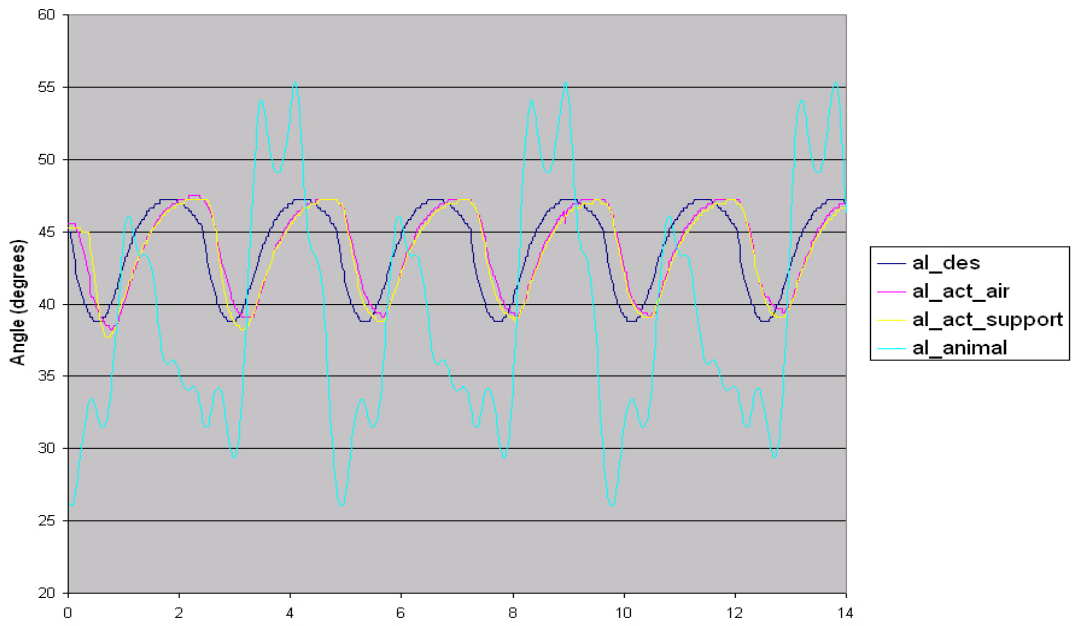


Figure 97 : Right Front Alpha Joint.

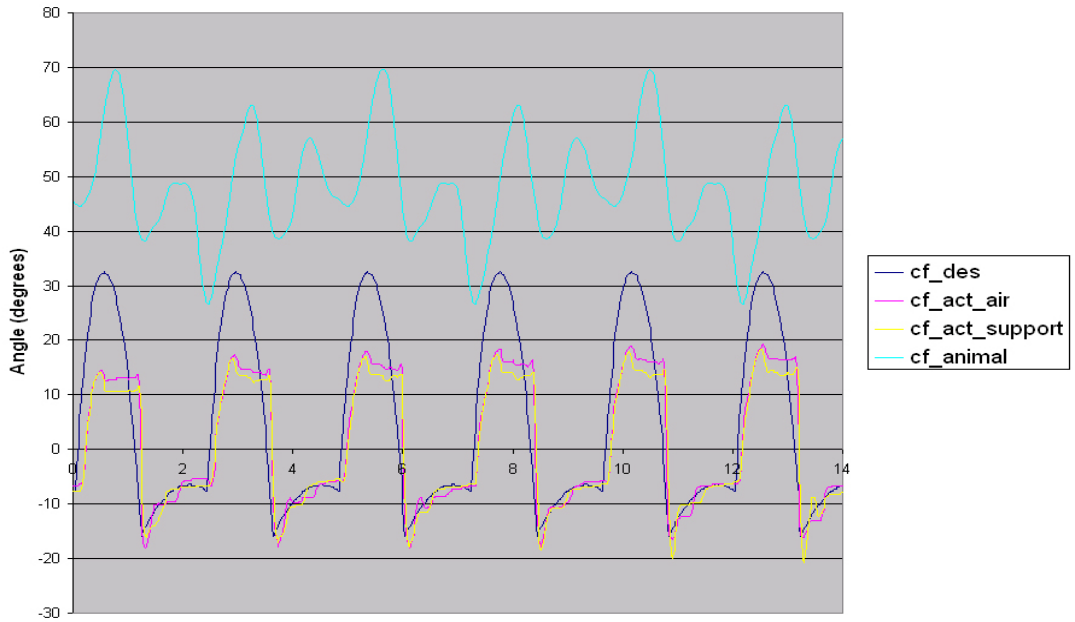


Figure 98 : Right Front CF Joint.

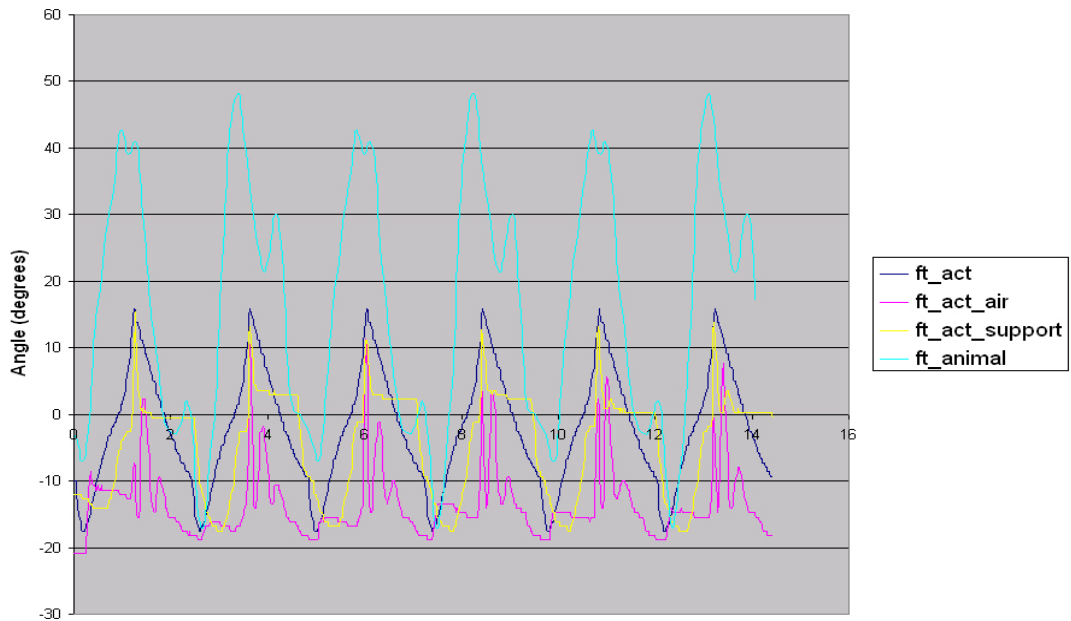


Figure 99 : Right Front FT Joint.

The gamma joint in the middle leg of Robot V is fixed at -35° . Like the front leg of Robot V, the Coxa segment of the middle leg is designed as an inclined one. The Coxa1 segment of Robot V is rotated by -72.9° . The cockroach's Gamma joint moves between -90 to -120° in Figure 100, which is located more inside the body. Figure 101 shows the Beta joint angles of desired, robot air walking, robot supported walking and animal walking for right middle leg. The Beta joint angles of the animal are smaller than those of the robot. Like the front leg, the cockroach and robot use a different strategy to swing the middle leg forward. The larger rotation of the Gamma joint in the robot makes the CF joint angle closer to that of the cockroach compared to the front leg (Figure 103). In the middle leg of the robot, the increased Gamma rotation makes the CF joint motion more similar to the joint angle motion of the cockroach. The FT joint of cockroach is more stretched out to reach further.

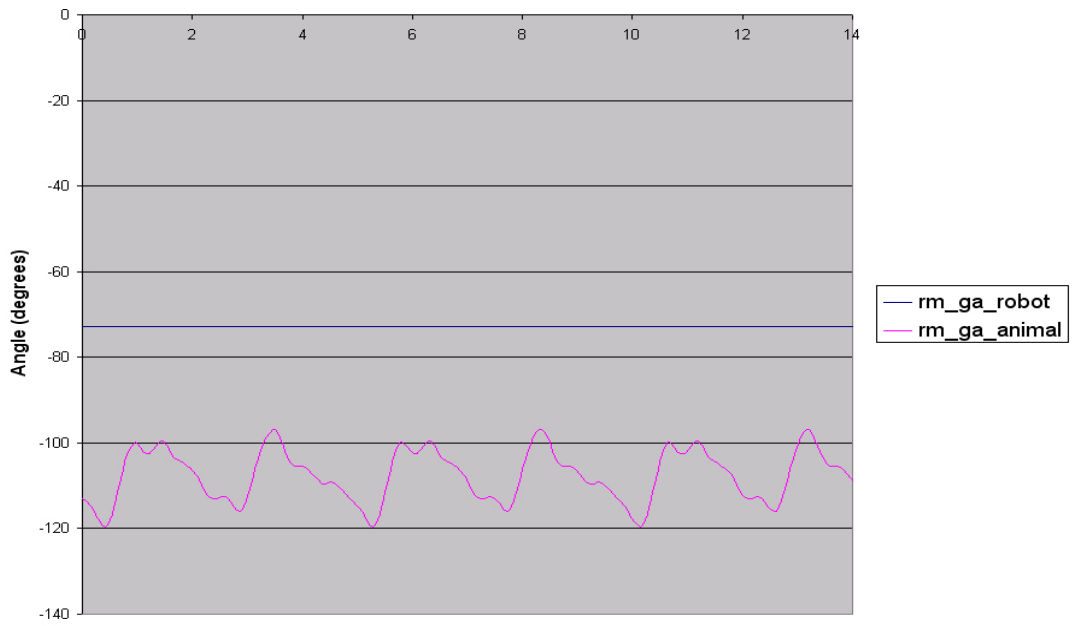


Figure 100 : Right Middle Gamma Joint

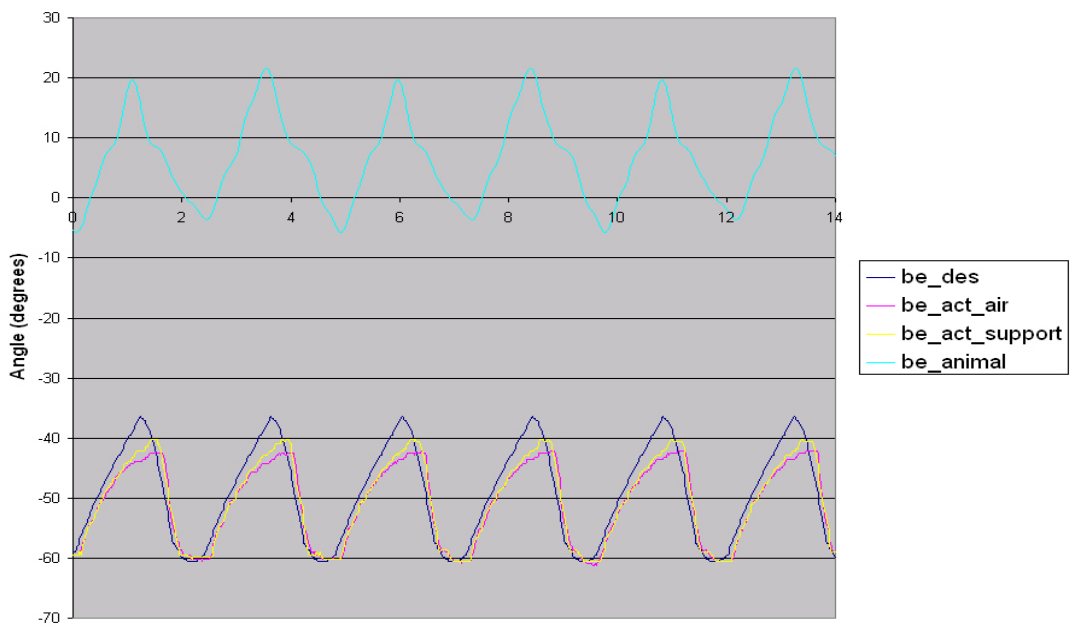


Figure 101 : Right Middle Beta Joint.

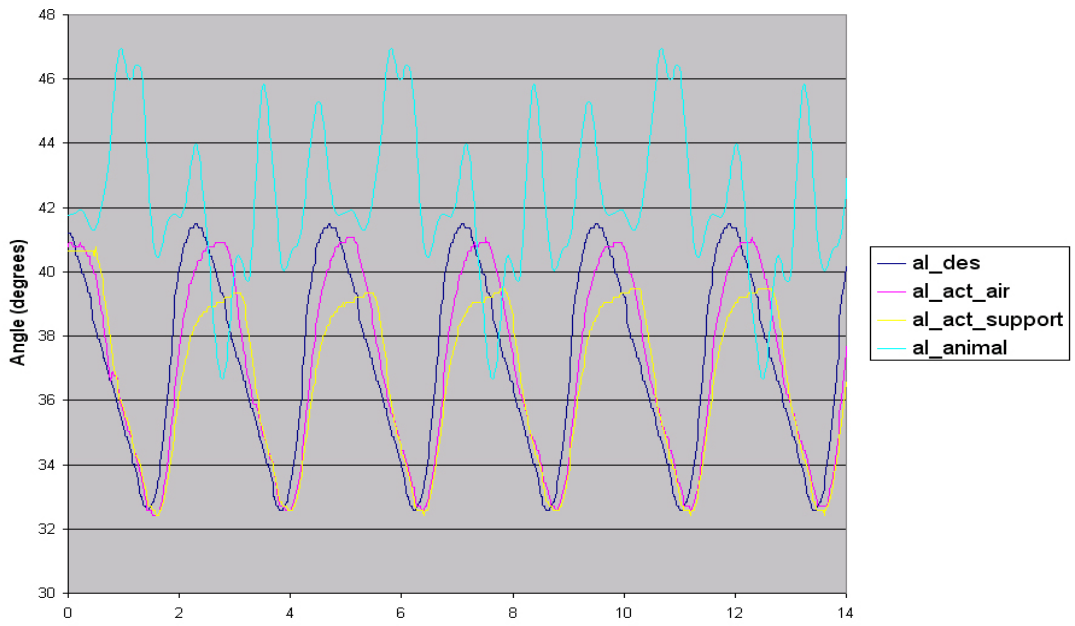


Figure 102 : Right Middle Alpha Joint.

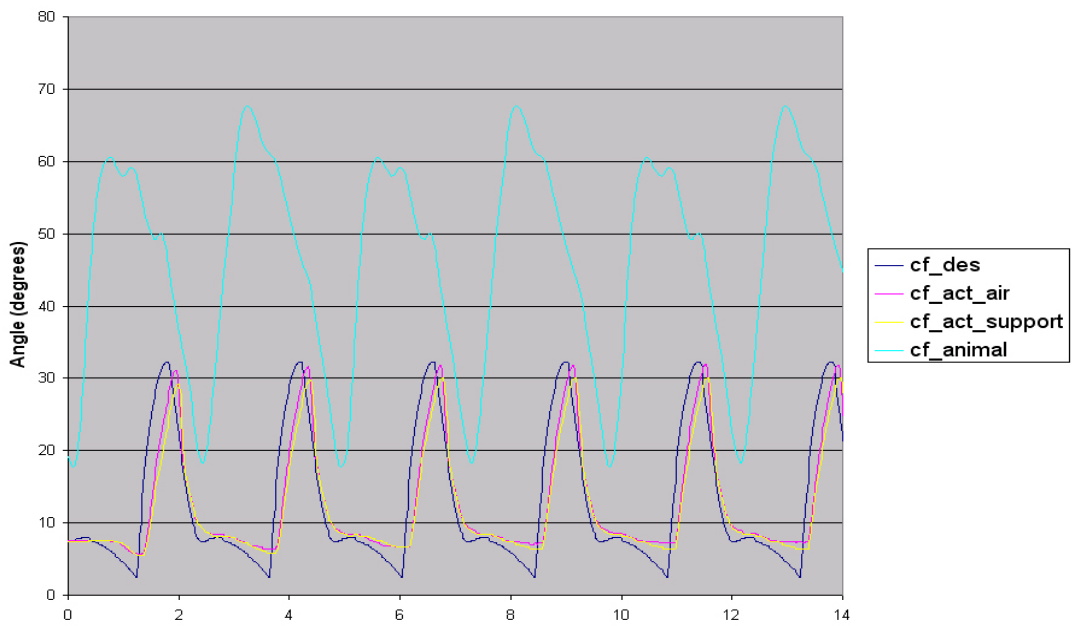


Figure 103 : Right Middle CF Joint.

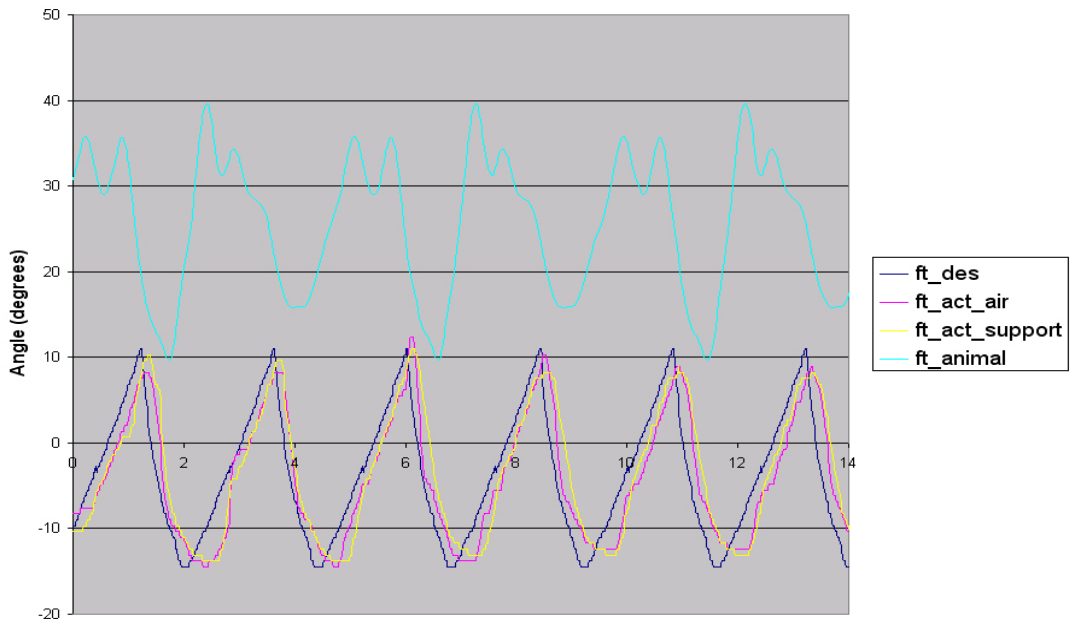


Figure 104 : Right Middle FT Joint.

The BC joint in the rear leg has fixed gamma and alpha joints. The Coxa segment of the rear leg is shifted about the x axis shown in Figure 17 and rotated by 35° about the z axis first and rotated by -30° about new y axis. The whole rear leg was rotated about the z axis by -35° and then fixed to the body frame. This design enables the rear leg to mimic the cockroach's rear leg posture. The Alpha joint rotation was replaced by the rotation about the z axis and the rotation about new y axis. In spite of the different configuration, the CF joint and FT joints in the robot are similar to those in the cockroach. Unlike in its front and middle legs, the cockroach's beta joint in its rear legs moves in the negative range. The rear leg's function is to push the body forward and it does not need to swing its leg passed its BC joint. Therefore, the beta joint does not need to move forward.

In the robot's rear leg, the alpha joint rotation is fixed. However, without Alpha joint rotation, the elevation of the rear leg's foot is higher than that of the front and

middle leg. The robot's CF joint motion in its rear leg is smaller than that observed in cockroach, but the robot's CF tracks that of cockroach. The FT joint cycles like that in the cockroach.

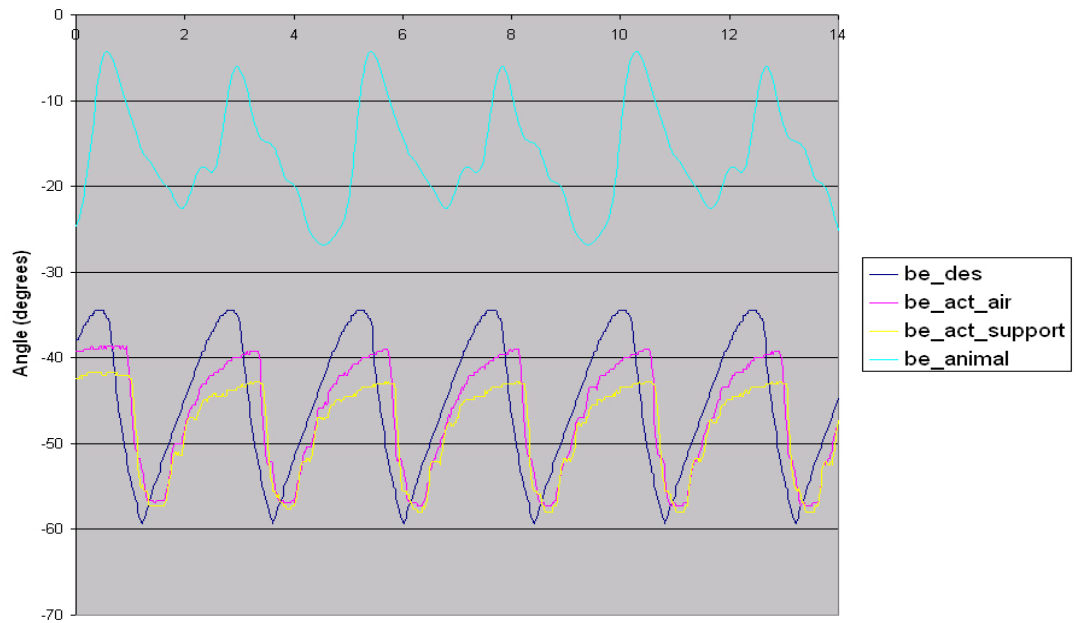


Figure 105 : Right Rear Beta Joint.

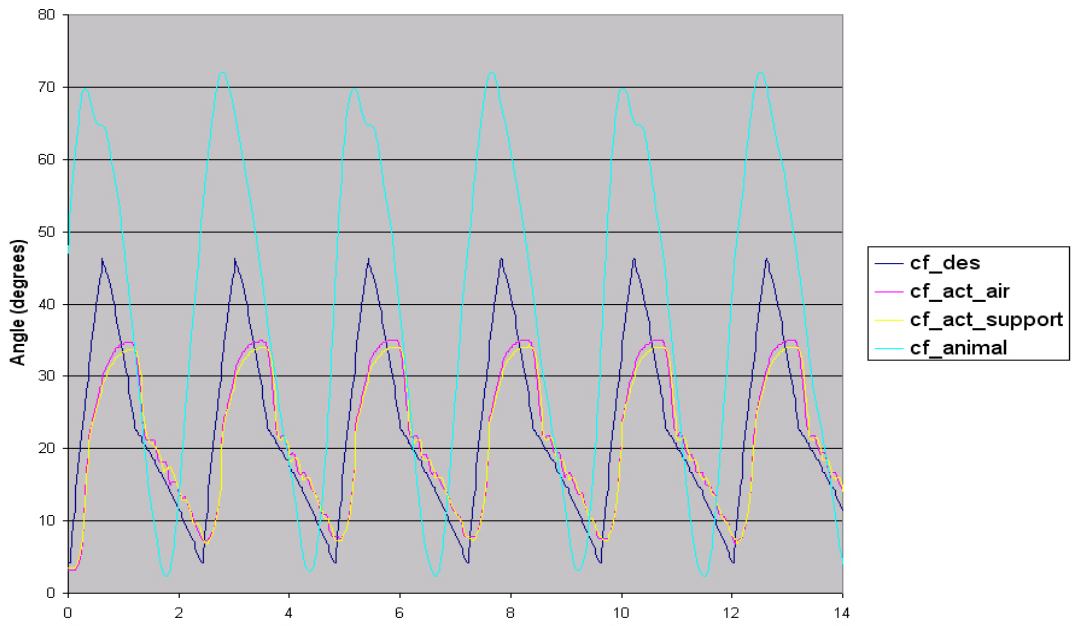


Figure 106 : Right Rear CF Joint.

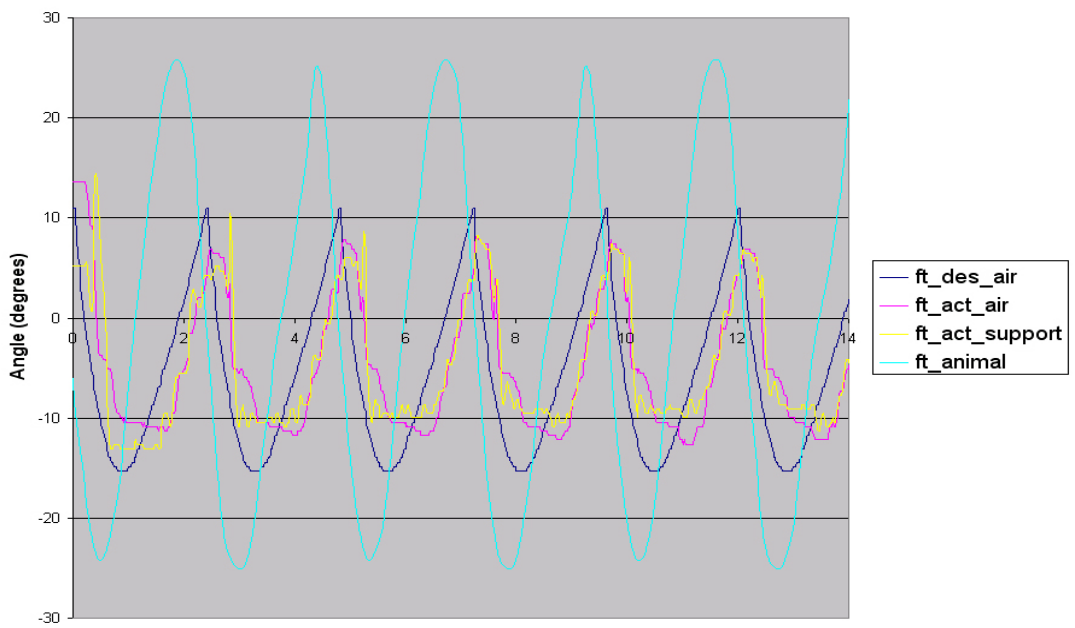


Figure 107 : Right Rear FT Joint.

9.5.3 Consequence of Outstretched Coxa Segment

The basic question about the posture is why does the insect maintain its coxa segment near the center of its body. I start with the hypothesis that the more the Coxa rotates away from the body, the more torque is necessary. To verify this hypothesis, the joint torques in the legs are predicted using dynamic simulation of the cockroach for different gamma joint angles. The simulated cockroach is standing statically on the ground. All joint desired angles are the same throughout all the tests except the Gamma joint, which is varied in each test. In these tests, the Gamma joint rotates by 0.2 radian ($\cong 11^\circ$) inside or outside. Figure 108 shows Gamma rotations in the simulated cockroach.

When the simulated cockroach is stabilized so that it is standing statically, the joint torques for all of the legs are summed. Figure 109 shows the total joint torque with varying Gamma joint angle. As the leg is rotated outside, the total joint torque increases. When the cockroach's Gamma joint is rotated inside more, the total joint torque is decreased. However, as the Gamma joint is rotated further inward, the foot position is moved in toward the body, which reduces the stability of the cockroach.

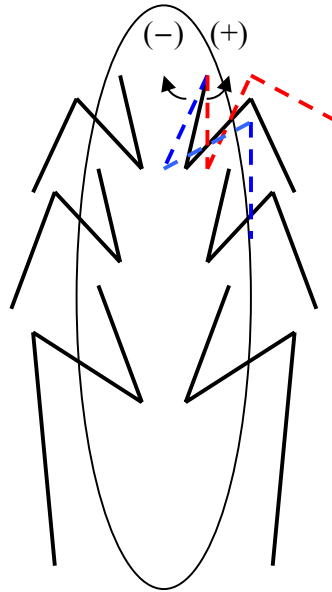


Figure 108 : Gamma Rotation Test. All joint commands are same through all tests except Gamma joint. Gamma joint is rotated inside or outside.

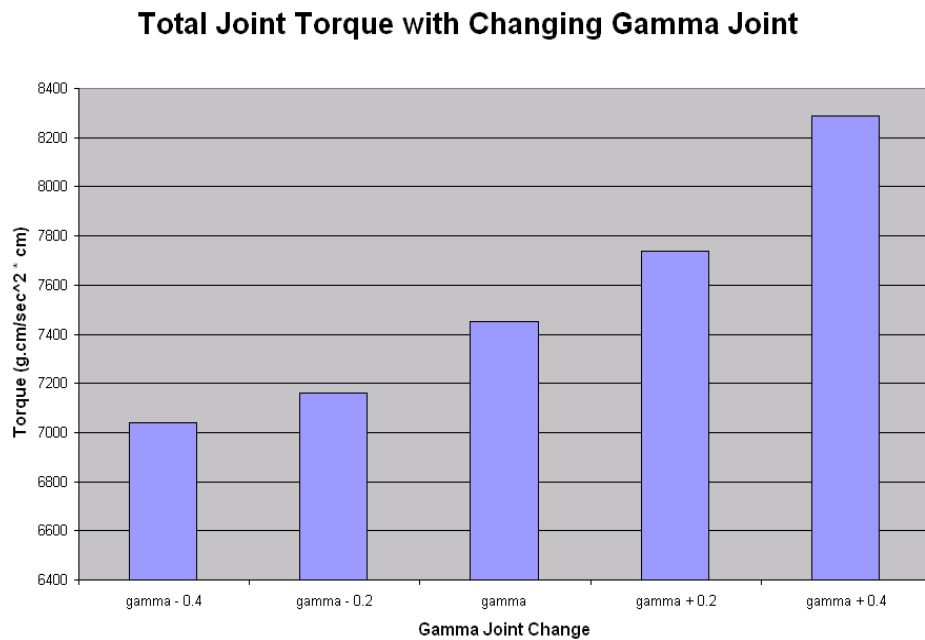


Figure 109 : Total Joint Torque Change corresponding to Gamma Joint Change. The Gamma joints are increased by 0.2.

9.6 Discussion: Can We Apply Biomechanics to Robot Directly?

Robot III and Robot V are designed differently. One major difference is in the BC joint. To reduce the joint torques, the cockroach coxa segment is rotated toward the center of the body.

What is the main mechanical design problem of Robot V? It is its BC joint design. Its Coxa segment contacts the ground when the Beta joint rotates in the negative direction. This causes the CF joint to drag and prevents Robot V from walking well. Further, the limited range of motion of its gamma joint and its inclined Coxa segment requires that Robot V use its Beta joint extensively in forward swing. The increased Beta joint motion makes the robot posture higher than that of the cockroach. Furthermore, the limited range of motion of the gamma joint does not permit the front legs to swing forward as far as they do in the cockroach.

The digitized cockroach and Robot V were compared. To put the foot nearer the body, the cockroach's Coxa segment is rotated inside the body. If this design is applied to the robot, it is possible for the Coxa segment to collide with Femur joint. This requires that the mechanical segments and actuators be packed in a small space to permit a larger range of motion.

During cockroach walking, due to its low body posture, sometimes the CF joint drags on the ground. Is this possible in Robot V? During Robot V walking, the CF joint of the middle leg touches the ground. It impedes Robot V from moving. The cockroach has a tarsus with gripping devices that gives it excellent traction so that drag from the CF

joint is easily overcome. Furthermore, the tarsus provides the cockroach body with extra elevation. A tarsus should be added to Robot V to raise its body higher and reduce CF ground contact. Regardless, the foot paths should be designed more carefully to avoid CF dragging. A three dimensional footpath should be considered rather than the current planar paths.

References

- 1 Watson, J.T. and Ritzmann, R. E. Leg kinematics and muscles activity during treadmill running in the cockroach, *Blaberus discoidalis*: I. Slow running. *J. Comp Physiol A* 182: 11-22, 1998.
- 2 Nelson, Gabriel. Modeling and Simulation of an Insect-like Hexapod, Master thesis. Case Western Reserve University, January 1995.
- 3 Dresden, D., and Nijenhuis, E. D. One the Anatomy and Mechanism of Motion of the Mesothoracic Leg of *Periplaneta Americanan*. *Proc. Kon. Ned. Akad. Wetensch. Ser. C.*, 56:39-47.
- 4 Kingsley, Daniel. A Cockroach Inspired Robot with Artificial Muscles, Ph. D. dissertation. Case Western Reserve University, January 2005.
- 5 Choi, Jongung. Dynamic Simulation of Cockroach Climbing. M.S. Thesis, CWRU. 2000

Chapter 10

Lessons Learned and Conclusions

The Robot V team included a mechanical designer, low level control engineer and high level control engineer. This is a good combination of expertise for a major robotics task. Many experiments have been done with Robot V. In this chapter, lessons learned from Robot V are explained and the conclusions and future work will be discussed.

10.1 Careful Choice of Sensors Saves Time

The flexible joint angle sensor which is used for Robot V is easy to use and mount, but this angle sensor is not reliable. The sensor was attached to steel and the steel was attached to rubber. This mount has hysteresis. An even more serious problem is that the output signal is not uniform. It tends to change during each experiment.

The nonlinear and non uniform sensor signal causes discordance between the joint sensor signal and actual joint angle. Each joint must be calibrated often to find the coefficients to convert the joint sensor digital signal to degrees.

$$\text{(sensor reading)} = b + m \times \text{(angle in degrees)} \quad (47)$$

where b and m are determined by calibration.

The first step is to measure the joint angle extreme positions and compare these numbers with the joint angle sensor's signal. When the two joint angle readings match each other, the coefficients are properly tuned. After tuning, the joint angle sensor signal is more reliable and it is matched with the actual physical joint angles.

The sensor choice is a very important part of the robot design because building a robot without confidence in its sensors is the same as building your home on the sand. A bad sensor choice means we have to spend a lot of time calibrating it.

10.2 Deep Insight into Animal Design is Essential in Bio-Inspired Design

Many researchers declare that their robot is modeled after an insect, but the making of a hexapod robot and the making of a bio-inspired hexapod robot are totally different. We are pursuing the development of a cockroach-inspired hexapod robot. We want to adopt the advantages of insect strategies for walking, climbing and turning because insects are remarkably agile. To apply these advantages in the robot, we need deep insight into insect motion, posture and its physical structure. Without this knowledge, we can not even attempt to develop a biorobot.

10.2.1 Local View of Joint Angle

The easiest metric to compare the insect and robot is a direct comparison of the joint angles. We can check the joint angles to see how similar the robot is modeled after the cockroach. Table 13 shows the original cockroach ROM and Robot V's ROM. All

angles are defined in the Robot V joint angle convention described in Chapter 8. All ROMs of cockroach are originally digitized in Ritzmann's Lab and analyzed in the Biorobotics Lab.

In terms of local joint position, the range of motion is indicative of cockroach motion and posture. For example, in Table 13, the ROM of the Gamma joint of the front leg of the cockroach is from -100 to -45 degrees. As I pointed out in Chapter 9, the Coxa segment of the cockroach is rotated inside toward the body center line. However, the Coxa segment of Robot V is directed more outward. The ROM of the Beta joint of the cockroach front leg is from -5 to 20 degrees. The cockroach can reach its legs far forward in swing with small Beta joint angles unlike Robot V, which needs larger Beta motions. The ROM of CF joint of the cockroach's right front leg is from 30 to 70 degrees, but the ROM of CF joint of Robot V's right front leg is from -20 to 30 degrees. Due to the different mechanical design of the BC joint, the ROM of CF joint of Robot V moves back and forth around the point where the CF joint angle is zero. The joint angle differences caused by the mechanical design of the BC joint are marked in magenta color in Table 11 and the differences indirectly affected by mechanical design are marked in cyan color. In Table 13, the ROM in yellow color falls short of the desired ROM.

What causes the limitation of ROM? The Festo actuator contraction ratio depends on actuator length. Because the Festo actuator can not extend, the actuator length should be decided considering coupled actuator's length. When the extensor is fully contracted, the length of the flexor can have its maximum length. Similarly, when the flexor is fully contracted, the length of the extensor can have its maximum length. If not, the ROM of the joint angle can not reach its maximum. Even though the joint can be pushed to reach

the maximum range (to its mechanical stop), the joint can not be controlled to move to that limit. The joint angle depends on the radius from the joint to the point where the actuator is attached. To get the proper range of motion, the moment arm should be carefully designed.

When we design a bio-inspired robot, the local joint positions can be the easiest metrics to compare between the animal and the robot. Understanding the ROM of the insect joints tells a lot of information about its motion and postures. When we design the bio-inspired robot, we have to test and measure the range of motion of the insect's joints.

Joint	Animal Walking	Dan's Desired ROM	Final Actual ROM		Tuned ROM
			Left Side	Right Side	
Front					
GA	-100 – -45	-52.16 – -12.16	12.16 – 52.16	-52.16 – -12.16	-52.16 – -12.16
BE	-5 – 20	-80 – -50	-75 – -35	-75 – -35	-75 – -35
AL	25 – 55	20 – 60	-45 – -20	20 – 45	20 – 45*
CF	30 – 70	-20 – 30	-25 – 25	-20 – 30	-20 – 25*
FT	-15 – 45	-30 – 45	-40 – 35	-30 – 50	-30 – 35*
Middle					
BE	-5 – 20	-80 – -50	-80 – -40	-76 – -30	-76 – -40*
AL	37 – 47	20 – 60	-35 – -23	27 – 37	27 – 35*
CF	18 – 68	-10 – 40	-45 – 0	5 – 40	5 – 40*
FT	10 – 40	-30 – 45	-45 – 40	-50 – 50	-40 – 45
Rear					
BE	-25 – -5	-40 – 0	-50 – -25	-43 – -21	-43 – -25*
CF	3 – 70	-10 – 40	-30 – 20	-8 – 34	-8 – 30*
FT	-25 – 25	-30 – 45	-35 – 40	-35 – 40	-35 – 35*

Table 13 : Revised ROM in degrees in Robot V angle definition convention

10.2.2 Global View of Body Posture

Even if we have enough information about local joint angles, if we do not consider the global body posture, we will have trouble when the robot moves. High speed

video is a good source to analyze insect motion. High speed video provides global postural information as well as local joint angle data.

I found that Robot V has a different body posture than the cockroach. The main difference between Robot V and the model cockroach is the BC joint. This small difference causes different posture and swing motion. During walking, the BC joint design difference and the inclined Coxa segment cause Beta rotations to be unlike those in the model cockroach. The CF joints contact the ground often and the posture of Robot V is higher than the cockroach.

When we design bio-inspired robots, a global view of posture is one of the important factors to be considered. We want to mimic or adopt the behavior of insects because the behavioral strategies of insects have advantages. To do so, we have to fully understand the behavior of the insect model.

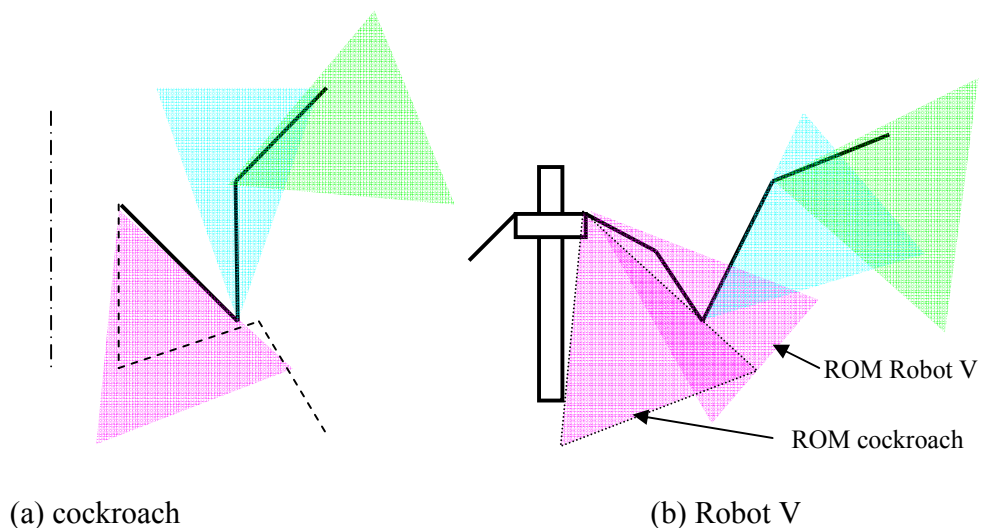


Figure 110 : Postural Strategy for Forward Swing. (a) cockroach's right front leg inside the ROMs of each leg. (b) Robot V's right front leg inside the ROMs of each leg.

10.3 Mechanical Design Differences

10.3.1 Inclined Coxa Segment

The inclined Coxa joint in Robot V causes a lot of problems. Robot V is designed based on a BC joint anatomy that remains in doubt. To get more cockroach-like motion, we have to change the BC joint. After this revision, the Gamma joint rotation should be shifted to cover the full ROM. Table 13 shows the ROM of the cockroach. The ROM of cockroach gamma at the front leg is from -100° to -45° . The Gamma joint should be revised. The ROM of CF joint at front leg is from 30° to 70° but the ROM of Robot V's CF joint is from -20° to 30° . The ROM of Robot V's CF joint is larger than that of the cockroach. The shift of ROM is enough to cover all desired CF rotation.

10.3.2 CF Joint

Several problems were found during my experiments including some in the middle legs. As mentioned in Section 9.6, the CF joint touches the ground during forward swing because of the rotational axis of the Beta joint. To protect the pneumatic actuators and to support the big actuator forces, Robot V's frames is designed as an exoskeleton. The pneumatic actuators are inside the exoskeleton frame. But the large exoskeleton structure causes the CF joint of the middle to drag. During forward swing, which

extensively uses the Beta joint, the elevation of the CF joint of the middle leg is lower than that of the foot. The Festo pneumatic actuator is strong. Sometimes the aluminum frame is deflected by the actuators. The frame should be designed with higher stiffness, but we have to check whether the structure collides with the environment or other parts in the robot.

10.3.3 Beta Joint at Front Leg

In the initial design the Beta joint of the front leg interfered with the gamma joint. After it was modified using the serial joint design, the interference problem was fixed. After revision, the beta joint produces the desired joint angles during air walking, but during ground walking, the Beta joint can not support the body's weight. The mounting positions of extensor and flexor actuators are different. The lengths of their moment arms are different. The moment arm of the extensor is too small to produce enough joint torque. This problem should be fixed by changing the position and length of the moment arm.

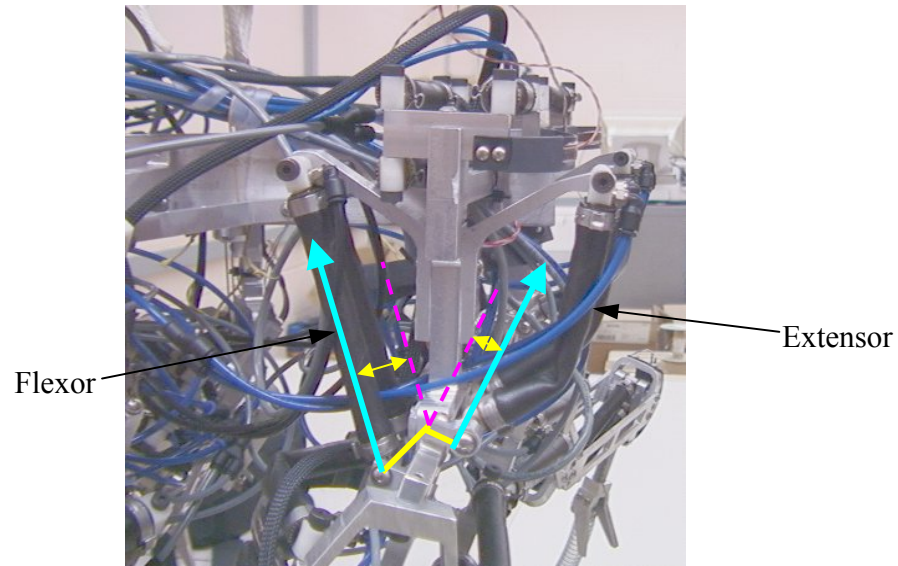


Figure 111 : Beta Joint of Right Front Leg. The Beta joint with the small moment arm can not produce enough joint torque.

10.4 Are the Actuators Strong Enough for Standing and Walking?

Robot V has walked but not efficiently and there are a lot of things to review and fix. In preliminary tests by Kingsley [1], Robot V walked using an open-loop controller. Even though a lot of work has been done and many improvements have been applied to Robot V, Robot V has not walked well. In simulation and open-loop walking, we verified that the actuators produce enough torque. Robot V's walking is actually worse using closed loop position control. What is the difference between open loop and closed loop?

In open-loop walking, most PWM commands are on (100) and off (0). The actuators were typically fully pressurized during open loop control.

In closed-loop position control, the PWM duty cycle commands depend on the joint angle errors. Joint stiffness or force was not controlled. This is a poor approach. During stance the control problem is one of force or stiffness, not joint angle tracking. The pressure sensors should be used in stance to apply force or stiffness control in feedback. The development of switching algorithms between joint tracking in swing and force control in stance needs more research in terms of robotics and biology.

10.5 Improvement of Gait Controller

The Cruse controller was implemented into the robot, but the current implemented Cruse controller is generated off-line. This means the Cruse controller could not refer to the current foot position. With the current gait controller, before the current foot position reaches the PEP, the foot starts the swing phase because the virtual foot reaches the PEP before the real foot. This is bad for stability and there is no reason to do it this way. During walking, the current foot position can be calculated using forward kinematics and then used in the gait controller.

10.6 Foot Slipping

When we design a large robot like Robot V, slipping is a considerable issue. Generally speaking, slipping causes the robot to not move forward well and to be energy inefficient. After the initial walking experiment, it was clear that Robot V has a slipping

problem. The feet of Robot V are made of the metal or a plastic tube and there are no attachment mechanisms to prevent slipping. An improved foot is essential for the robot to walk well.

10.7 Conclusions

Robot V has been developed as one of a series of five hexapod robots. Robot III could produce smooth walking motions in air. Because its valve system could not trap air, it could not walk on the ground well. Kingsley designed and constructed Robot V and he made Robot V walk in a limited way with open-loop control. The successful open-loop walking inspires us to develop more intelligent control. For closing the control loop, we added joint angles sensors and pressure sensors. The Cruse gait controller and a neural network for solving inverse kinematics were developed to coordinate the robot's joints and legs. It can perform walking motions while suspended in the air and its joints and feet can follow desired trajectories. This was the first time Festo actuators were used in the Biorobotics lab. The Festo actuator was modeled and tested in simulation to better understand its properties for use in control of the robot. All components and control schemes were tested in simulation first and then they were applied to Robot V. Many problems were found during ground walking experiments. These problems are related to mechanical design choices and control problems.

Some problems were solved by changing the mechanical design. However, Robot V needs more mechanical design fixes and force or stiffness control. The key to solving this problem is switching efficiently between trajectory-based swing to force-based

stance. Also, Robot V can not walk well without effective traction mechanisms on its feet. Furthermore, posture control should be applied to move the body appropriately.

References

- 1 Kingsley, Daniel. A Cockroach Inspired Robot with Artificial Muscles, Ph. D. dissertation. Case Western Reserve University, January 2005.

Appendix

A. Revised Cruise Controller

```
//----- Forward Kinematics for Right Front -----//
public void forward_If(YoVariable[] angle, YoVariable [] pfoot)
{
    double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [] vec_temp1 = {0.0, 0.0, 0.0},
        vec_temp2 = {0.0, 0.0, 0.0},
        vec_temp3 = {0.0, 0.0, 0.0},
        vec_temp4 = {0.0, 0.0, 0.0},
        vec_temp5 = {0.0, 0.0, 0.0},
        vec_temp6 = {0.0, 0.0, 0.0};
    double [] l1 = {-2.37, 3.769, 0};
    double [] l2 = {-3, 4.045, 0};
    double [] l3 = {0.0, 0.0, 0.0};
    double [] l4 = {0.0, 0.0, 0.0};
    double [] dirl1 = {0.0, 0.0, 0.0}, dirl2= {0.0, 0.0, 0.0}, dirl3 = {0.0, 0.0, 0.0};
    double [] position = {0.0, 0.0, 0.0};

    rz(angle[0].val, mat_temp1);
    ry(angle[1].val, mat_temp2);
    multiply_mat_mat(mat_temp1, mat_temp2, c01);
    rx(angle[2].val, c12);
    rz(angle[3].val+Math.atan2(4.045,3), c23); //cf in Yobotics_If + atan2(4.043,3) = cf in forward_If
    rz(angle[4].val-Math.PI/2, c34); //ft in Yobotics_If - PI/2 = ft in forward_rf

    direction(l2,dirl2);
    rz(-Math.PI/2,mat_temp1);
    multiply_mat_vec(mat_temp1,dirl2,dirl3);
    l3[0] = 5.864 * dirl3[0];
    l3[1] = 5.864 * dirl3[1];
    l3[2] = 5.864 * dirl3[2];
    l4[0] = 5.80 * dirl2[0];
    l4[1] = 5.80 * dirl2[1];
    l4[2] = 5.80 * dirl2[2];

    multiply_mat_vec(c34, l4, vec_temp1);
    add(l3, vec_temp1, vec_temp2);
    multiply_mat_vec(c23, vec_temp2, vec_temp3);
    add(l2, vec_temp3, vec_temp4);
    multiply_mat_vec(c12, vec_temp4, vec_temp5);
    add(l1, vec_temp5, vec_temp6);
    multiply_mat_vec(c01, vec_temp6, position);
    pfoot[0].val=position[0];
}
```

```

pfoot[1].val=position[1];
pfoot[2].val=position[2];
}

//----- Forward Kinematics of Right Front -----//
public void forward_rf(YoVariable [] angle, YoVariable [] pfoot)
{
double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [] vec_temp1 = {0.0, 0.0, 0.0},
vec_temp2 = {0.0, 0.0, 0.0},
vec_temp3 = {0.0, 0.0, 0.0},
vec_temp4 = {0.0, 0.0, 0.0},
vec_temp5 = {0.0, 0.0, 0.0},
vec_temp6 = {0.0, 0.0, 0.0};
double [] l1 = {-2.37, -3.769, 0};
double [] l2 = {-3, -4.045, 0};
double [] l3 = {0.0, 0.0, 0.0};
double [] l4 = {0.0, 0.0, 0.0};
double [] dir1 = {0.0, 0.0, 0.0}, dir2= {0.0, 0.0, 0.0}, dir3 = {0.0, 0.0, 0.0};
double [] position = {0.0, 0.0, 0.0};

rz(angle[0].val, mat_temp1);
ry(angle[1].val, mat_temp2);
multiply_mat_mat(mat_temp1, mat_temp2, c01);
rx(angle[2].val, c12);
rz(angle[3].val-Math.atan2(4.045,3), c23); //cf in Yobotics-atan2(4.043,3) = cf in forward_rf
rz(angle[4].val+Math.PI/2, c34); //ft in Yobotics+PI/2 = ft in forward_rf

direction(l2,dir2);
rz(Math.PI/2,mat_temp1);
multiply_mat_vec(mat_temp1,dir2,dir3);
l3[0] = 5.864 * dir3[0];
l3[1] = 5.864 * dir3[1];
l3[2] = 5.864 * dir3[2];
l4[0] = 5.80 * dir2[0];
l4[1] = 5.80 * dir2[1];
l4[2] = 5.80 * dir2[2];

multiply_mat_vec(c34, l4, vec_temp1);
add(l3, vec_temp1, vec_temp2);
multiply_mat_vec(c23, vec_temp2, vec_temp3);
add(l2, vec_temp3, vec_temp4);
multiply_mat_vec(c12, vec_temp4, vec_temp5);
add(l1, vec_temp5, vec_temp6);
multiply_mat_vec(c01, vec_temp6, position);
pfoot[0].val=position[0];
pfoot[1].val=position[1];
pfoot[2].val=position[2];
}

```

```

}

//----- Forward Kinematics of Left Middle -----//
public void forward_lm(YoVariable [] angle, YoVariable [] pfoot)
{
    double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [] vec_temp1 = {0.0, 0.0, 0.0},
        vec_temp2 = {0.0, 0.0, 0.0},
        vec_temp3 = {0.0, 0.0, 0.0},
        vec_temp4 = {0.0, 0.0, 0.0},
        vec_temp5 = {0.0, 0.0, 0.0},
        vec_temp6 = {0.0, 0.0, 0.0};
    double [] l1 = {-2.55, 3.275, 0};
    double [] l2 = {-3.33, 2.774, 0};
    double [] l3 = {0.0, 0.0, 0.0};
    double [] l4 = {0.0, 0.0, 0.0};
    double [] dir1 = {0.0, 0.0, 0.0}, dir2= {0.0, 0.0, 0.0}, dir3 = {0.0, 0.0, 0.0};
    double [] position = {0.0, 0.0, 0.0};

    rz(35*Math.PI/180, mat_temp1);
    ry(angle[0].val, mat_temp2);
    multiply_mat_mat(mat_temp1, mat_temp2, c01);
    rx(angle[1].val, c12);
    rz(angle[2].val + Math.atan2(2.744,3.33), c23); //cf in Yobotics_lm + atan2(2.744,3.33) = cf in
forward_lm
    rz(angle[3].val - Math.PI/2, c34); //ft in Yobotics_lm - pi/2 = ft in forward_lm

    direction(l2,dir2);
    rz(-Math.PI/2,mat_temp1);
    multiply_mat_vec(mat_temp1,dir2,dir3);
    l3[0] = 7.16 * dir3[0];
    l3[1] = 7.16 * dir3[1];
    l3[2] = 7.16 * dir3[2];
    l4[0] = 6.625 * dir2[0];
    l4[1] = 6.625 * dir2[1];
    l4[2] = 6.625 * dir2[2];

    multiply_mat_vec(c34, l4, vec_temp1);
    add(l3, vec_temp1, vec_temp2);
    multiply_mat_vec(c23, vec_temp2, vec_temp3);
    add(l2, vec_temp3, vec_temp4);
    multiply_mat_vec(c12, vec_temp4, vec_temp5);
    add(l1, vec_temp5, vec_temp6);
    multiply_mat_vec(c01, vec_temp6, position);
    pfoot[0].val=position[0];
    pfoot[1].val=position[1];
    pfoot[2].val=position[2];
}

```



```

//----- Forward Kinematics of Right Middle -----//
public void forward_rm(YoVariable[] angle, YoVariable [] pfoot)
{
    double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [] vec_temp1 = {0.0, 0.0, 0.0},
        vec_temp2 = {0.0, 0.0, 0.0},
        vec_temp3 = {0.0, 0.0, 0.0},
        vec_temp4 = {0.0, 0.0, 0.0},
        vec_temp5 = {0.0, 0.0, 0.0},
        vec_temp6 = {0.0, 0.0, 0.0};
    double [] l1 = {-2.55, -3.275, 0};
    double [] l2 = {-3.33, -2.774, 0};
    double [] l3 = {0.0, 0.0, 0.0};
    double [] l4 = {0.0, 0.0, 0.0};
    double [] dir11 = {0.0, 0.0, 0.0}, dir12= {0.0, 0.0, 0.0}, dir13 = {0.0, 0.0, 0.0};
    double [] position = {0.0, 0.0, 0.0};

    rz(-35*Math.PI/180, mat_temp1);
    ry(angle[0].val, mat_temp2);
    multiply_mat_mat(mat_temp1, mat_temp2, c01);
    rx(angle[1].val, c12);
    rz(angle[2].val - Math.atan2(2.744,3.33), c23); //cf in Yobotics_rm - atan2(2,744,3.33) = cf in
forward_rm
    rz(angle[3].val + Math.PI/2, c34); //ft in Yobotics_rm + pi/2 = ft in forward_rm

    direction(l2,dir12);
    rz(Math.PI/2,mat_temp1);
    multiply_mat_vec(mat_temp1,dir12,dir13);
    l3[0] = 7.16 * dir13[0];
    l3[1] = 7.16 * dir13[1];
    l3[2] = 7.16 * dir13[2];
    l4[0] = 6.625 * dir12[0];
    l4[1] = 6.625 * dir12[1];
    l4[2] = 6.625 * dir12[2];

    multiply_mat_vec(c34, l4, vec_temp1);
    add(l3, vec_temp1, vec_temp2);
    multiply_mat_vec(c23, vec_temp2, vec_temp3);
    add(l2, vec_temp3, vec_temp4);
    multiply_mat_vec(c12, vec_temp4, vec_temp5);
    add(l1, vec_temp5, vec_temp6);
    multiply_mat_vec(c01, vec_temp6, position);
    pfoot[0].val=position[0];
    pfoot[1].val=position[1];
    pfoot[2].val=position[2];
}

//----- Forward Kinematics of Left Rear -----//
public void forward_lr(YoVariable [] angle, YoVariable [] pfoot)
{

```

```

double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp4 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp5 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] mat_temp6 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};

double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
double [] vec_temp1 = {0.0, 0.0, 0.0},
      vec_temp2 = {0.0, 0.0, 0.0},
      vec_temp3 = {0.0, 0.0, 0.0},
      vec_temp4 = {0.0, 0.0, 0.0},
      vec_temp5 = {0.0, 0.0, 0.0},
      vec_temp6 = {0.0, 0.0, 0.0};
double [] l1 = {-1.75, 0., 0.};
double [] l2 = {-5.5, 0., 0.};
double [] l3 = {0.0, 0.0, 0.0};
double [] l4 = {0.0, 0.0, 0.0};
double [] dir1 = {0.0, 0.0, 0.0}, dir2= {0.0, 0.0, 0.0}, dir3 = {0.0, 0.0, 0.0};
double [] position = {0.0, 0.0, 0.0};

rz(35*Math.PI/180, mat_temp1);
ry(angle[0].val, mat_temp2);
multiply_mat_mat(mat_temp1, mat_temp2, c01);
rz(-35*Math.PI/180, mat_temp3);
ry(-30*Math.PI/180, mat_temp4);
multiply_mat_mat(mat_temp3, mat_temp4, c12);
rz(angle[1].val, c23);
rz(angle[2].val - Math.PI/2, c34);

direction(l2,dir2);
rz(-Math.PI/2,mat_temp1);
multiply_mat_vec(mat_temp1,dir2,dir3);
l3[0] = 7.74 * dir3[0];
l3[1] = 7.74 * dir3[1];
l3[2] = 7.74 * dir3[2];
l4[0] = 9.875 * dir2[0];
l4[1] = 9.875 * dir2[1];
l4[2] = 9.875 * dir2[2];

multiply_mat_vec(c34, l4, vec_temp1);
add(l3, vec_temp1, vec_temp2);
multiply_mat_vec(c23, vec_temp2, vec_temp3);
add(l2, vec_temp3, vec_temp4);
multiply_mat_vec(c12, vec_temp4, vec_temp5);
add(l1, vec_temp5, vec_temp6);
multiply_mat_vec(c01, vec_temp6, position);
pfoot[0].val=position[0];
pfoot[1].val=position[1];
pfoot[2].val=position[2];
}

```

//----- Forward Kinematics of Right Rear -----//

```

public void forward_rr(YoVariable [] angle, YoVariable [] pfoot)
{
    double [][] mat_temp1 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp2 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp3 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp4 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp5 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] mat_temp6 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};

    double [][] c01 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c12 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c23 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [][] c34 = {{0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0},{ 0.0, 0.0, 0.0}};
    double [] vec_temp1 = {0.0, 0.0, 0.0},
        vec_temp2 = {0.0, 0.0, 0.0},
        vec_temp3 = {0.0, 0.0, 0.0},
        vec_temp4 = {0.0, 0.0, 0.0},
        vec_temp5 = {0.0, 0.0, 0.0},
        vec_temp6 = {0.0, 0.0, 0.0};
    double [] l1 = {-1.75, 0., 0.};
    double [] l2 = {-5.5, 0., 0.};
    double [] l3 = {0.0, 0.0, 0.0};
    double [] l4 = {0.0, 0.0, 0.0};
    double [] dir1 = {0.0, 0.0, 0.0}, dir2= {0.0, 0.0, 0.0}, dir3 = {0.0, 0.0, 0.0};
    double [] position = {0.0, 0.0, 0.0};

    rz(-35*Math.PI/180, mat_temp1);
    ry(angle[0].val, mat_temp2);
    multiply_mat_mat(mat_temp1, mat_temp2, c01);
    rz(35*Math.PI/180, mat_temp3);
    ry(-30*Math.PI/180, mat_temp4);
    multiply_mat_mat(mat_temp3, mat_temp4, c12);
    rz(angle[1].val, c23);
    rz(angle[2].val + Math.PI/2, c34);

    direction(l2,dir12);
    rz(Math.PI/2,mat_temp1);
    multiply_mat_vec(mat_temp1,dir12,dir13);
    l3[0] = 7.74 * dir13[0];
    l3[1] = 7.74 * dir13[1];
    l3[2] = 7.74 * dir13[2];
    l4[0] = 9.875 * dir12[0];
    l4[1] = 9.875 * dir12[1];
    l4[2] = 9.875 * dir12[2];

    multiply_mat_vec(c34, l4, vec_temp1);
    add(l3, vec_temp1, vec_temp2);
    multiply_mat_vec(c23, vec_temp2, vec_temp3);
    add(l2, vec_temp3, vec_temp4);
    multiply_mat_vec(c12, vec_temp4, vec_temp5);
    add(l1, vec_temp5, vec_temp6);
    multiply_mat_vec(c01, vec_temp6, position);
    pfoot[0].val=position[0];
    pfoot[1].val=position[1];
    pfoot[2].val=position[2];
}

```

```

//----- Cruse Controller -----//
public void cruseControl(double [][] position)
{
    if ((stancespeed<=gaitspeed*0.975)||stancespeed>=gaitspeed*1.025))
        stancespeed = stancespeed*gaitramp;
    else
        stancespeed = gaitspeed;

    //----- Find the forward kinematic -----//
    forward_lf(q_joint_lf,pos_lf);
    forward_rf(q_joint_rf,pos_rf);
    forward_lm(q_joint_lm,pos_lm);
    forward_rm(q_joint_rm,pos_rm);
    forward_lr(q_joint_lr,pos_lr);
    forward_rr(q_joint_rr,pos_rr);

    /* each leg will get its own if statement */
    /* values are reassigned based on leg state here */

    /* -----left front section----- */

    if (leftfrontcontact == 1)                                     /* foot on ground */
    {
        leftfrontz = leftfrontground;
        leftfrontpos = leftfrontpos - stancespeed*timestep;
        leftfrontl = 0;
        if ((leftfrontpos) > (leftfrontaep - (leftfrontaep - leftfrontpep)/10))
        {
            leftfront2 = mechanism2;
            leftfrontcross2 = crossmechanism2;
        }
        else
        {
            leftfront2 = 0;
            leftfrontcross2 = 0;
        }

        if (leftfrontpos < leftfronttarget)                         /* initiate swing */
        {
            if(pos_lf_x.val < leftfronttarget)
            {
                leftfrontcontact = 0;
                leftfrontend = leftfrontpos;
            }
            else leftfrontcontact = 1;

        }
        else
        {
            leftfront5 = 0.5*mechanism5*((leftfrontaep-leftfrontpos)/(leftfrontaep-leftfrontpep));
            leftfrontcross5 = 0.5*crossmechanism5*((leftfrontaep-leftfrontpos)/(leftfrontaep-
leftfrontpep));
        }
    }
}

```

```

}
else
{
    leftfrontpos = leftfrontpos + swingspeed*timestep;
    leftfront1 = mechanism1;
    leftfront2 = 0;
    leftfrontcross2 = 0;
    leftfront5 = 0;
    leftfrontcross5 = 0;
    leftfrontz = (4*(leftfrontzmax - leftfrontground)/((leftfrontaep - leftfrontend)*(leftfrontaep -
leftfrontend))) * (leftfrontpos - leftfrontend)*(leftfrontaep - leftfrontpos) + leftfrontground;
    if (leftfrontpos > leftfrontaep)
        /* initiate stance */
        {
            leftfrontcontact = 1;
            leftfront2 = 0;
            leftfrontcross2 = 0;
            leftfrontcount = 1;
            leftfrontz = leftfrontground;
        }
}

/* -----end left front section----- */

/* -----right front section----- */
if (rightfrontcontact == 1)
{
    rightfrontz = rightfrontground;
    rightfrontpos = rightfrontpos - stancespeed*timestep;
    rightfront1 = 0;
    /* keep mechanisms clear */
    if((rightfrontpos) > (rightfrontaep - (rightfrontaep - rightfrontpep)/10))
    {
        rightfront2 = mechanism2;
        rightfrontcross2 = crossmechanism2*contbias;
    }
    else
    {
        rightfront2 = 0;
        rightfrontcross2 = 0;
    }
    if (rightfrontpos < rightfronttarget)
    {
        if(pos_rf_x.val<rightfronttarget)
        {
            rightfrontcontact = 0;
            rightfrontend = rightfrontpos;
        }
        else rightfrontcontact = 1;
    }
    else
    {
        rightfront5 = 0.5*mechanism5*((rightfrontaep - rightfrontpos)/(rightfrontaep -
rightfrontpep));
        rightfrontcross5 = 0.5*crossmechanism5*((rightfrontaep - rightfrontpos)/(rightfrontaep -
rightfrontpep))*contbias;
    }
}

```

```

    }
}
else
{
    rightfrontpos = rightfrontpos + swingspeed*timestep;
    rightfront1 = mechanism1;
    rightfront2 = 0;
    rightfrontcross2 = 0;
    rightfront5 = 0;
    rightfrontcross5 = 0;
    rightfrontz = (4*(rightfrontzmax - rightfrontground)/((rightfrontaep -
rightfrontend)*(rightfrontaep - rightfrontend))) * (rightfrontpos - rightfrontend)*(rightfrontaep -
rightfrontpos) + rightfrontground;
    if (rightfrontpos > rightfrontaep)
    {
        rightfrontcontact = 1;
        rightfront2 = 0;
        rightfrontcross2 = 0;
        rightfrontcount = 1;
        rightfrontz = rightfrontground;
    }
}
}

/* -----end right front section----- */

/* -----left mid section----- */
if (leftmidcontact == 1)
{
    leftmidz = leftmidground;
    leftmidpos = leftmidpos - stancespeed*timestep;
    leftmid1 = 0;
    /* keep mechanisms clear */
    if (leftmidpos > (leftmidaep - (leftmidaep - leftmidpep)/10))
    {
        leftmid2 = mechanism2;
        leftmidcross2 = crossmechanism2;
    }
    else
    {
        leftmid2 = 0;
        leftmidcross2 = 0;
    }
    if (leftmidpos < leftmidtarget)
    {
        if (pos_lm_x.val < leftmidtarget)
        {
            leftmidcontact = 0;
            leftmidend = leftmidpos;
        }
        else leftmidcontact = 1;
    }
    else
    {

```

```

        leftmid5 = 0.5*mechanism5*((leftmidaep - leftmidpos)/(leftmidaep - leftmidpep));
        leftmidcross5 = 0.5*crossmechanism5*((leftmidaep - leftmidpos)/(leftmidaep -
leftmidpep));
    }

}
else
{
    leftmidpos = leftmidpos + swingspeed*timestep;
    leftmid1 = mechanism1;
    leftmid2 = 0;
    leftmidcross2 = 0;
    leftmid5 = 0;
    leftmidcross5 = 0;
    leftmidz = (4*(leftmidzmax - leftmidground)/((leftmidaep - leftmidend)*(leftmidaep -
leftmidend))) * (leftmidpos - leftmidend)*(leftmidaep - leftmidpos) + leftmidground;
    if (leftmidpos > leftmidaep)
    {
        leftmidcontact = 1;
        leftmid2 = 0;
        leftmidcross2 = 0;
        leftmidcount = 1;
        leftmidz = leftmidground;
    }
}

}

/* -----end left mid section----- */

/* -----right mid section----- */
if (rightmidcontact == 1)
{
    rightmidz = rightmidground;
    rightmidpos = rightmidpos - stancespeed*timestep;
    rightmid1 = 0;
    /* keep mechanisms clear */
    if ((rightmidpos) > (rightmidaep - (rightmidaep - rightmidpep)/10))
        /* keep mechanisms clear */
    {
        rightmid2 = mechanism2;
        rightmidcross2 = crossmechanism2*contbias;
    }
    else
    {
        rightmid2 = 0;
        rightmidcross2 = 0;
    }
    if (rightmidpos < rightmidtarget)
    {
        if (pos_rm_x.val < rightmidtarget)
        {
            rightmidcontact = 0;
            rightmidend = rightmidpos;
        }
        else rightmidcontact = 1;
    }
}

```

```

else
{
    rightmid5 = 0.5*mechanism5*((rightmidaep - rightmidpos)/(rightmidaep - rightmidpep));
    rightmidcross5 = 0.5*crossmechanism5*((rightmidaep - rightmidpos)/(rightmidaep -
rightmidpep))*contbias;
}
}
else
{
    rightmidpos = rightmidpos + swingspeed*timestep;
    rightmid1 = mechanism1;
    rightmid2 = 0;
    rightmidcross2 = 0;
    rightmid5 = 0;
    rightmidcross5 = 0;
    rightmidz = (4*(rightmidzmax - rightmidground)/((rightmidaep - rightmidend)*(rightmidaep -
rightmidend))) * (rightmidpos - rightmidend)*(rightmidaep - rightmidpos) + rightmidground;
    if (rightmidpos > rightmidaep)
    {
        rightmidcontact = 1;
        rightmid2 = 0;
        rightmidcross2 = 0;
        rightmidcount = 1;
        rightmidz = rightmidground;
    }
}
}
/* -----end right mid section----- */

/* -----left rear section----- */
if (leftrearcontact == 1)
{
    leftrearz = leftrearground;
    leftrearpos = leftrearpos - stancespeed*timestep;
    if ((leftrearpos) > (leftrearaep - (leftrearaep - leftrearpep)/10))
        /* keep mechanisms clear */
    {
        leftrear2 = mechanism2;
        leftrearcross2 = crossmechanism2;
    }
    else
    {
        leftrear2 = 0;
        leftrearcross2 = 0;
    }
    leftrearcross2 = 0;
    if (leftrearpos < leftreartarget)
    {
        if(pos_lr_x.val < leftreartarget)
        {
            leftrearcontact = 0;
            leftrearend = leftrearpos;
        }
        else leftrearcontact = 1;
    }
}

```



```

    }
    else
    {
        leftrear5 = 0.5*mechanism5*((leftrearaep - leftrearpos)/(leftrearaep - leftrearpep));
        leftrearcross5 = 0.5*crossmechanism5*((leftrearaep - leftrearpos)/(leftrearaep -
leftrearpep));
    }

}
else
{
    leftrearpos = leftrearpos + swingspeed*timestep;
    leftrear1 = mechanism1;
    leftrearcross1 = crossmechanism1;
    leftrear2 = 0;
    leftrearcross2 = 0;
    leftrear5 = 0;
    leftrearcross5 = 0;
    leftrearz = (4*(leftrearzmax - leftrearground)/((leftrearaep - leftrearend)*(leftrearaep -
leftrearend))) * (leftrearpos - leftrearend)*(leftrearaep - leftrearpos) + leftrearground;
    if (leftrearpos > leftrearaep)
    {
        leftrearcontact = 1;
        leftrear2 = 0;
        leftrearcross2 = 0;
        leftrearcount = 1;
        leftrearz = leftrearground;
    }
}

}

/* -----end left rear section----- */

/* -----right rear section----- */
if (rightrearcontact == 1)
{
    rightrearz = rightrearground;
    rightrearpos = rightrearpos - stancespeed*timestep;
    rightrear1 = 0;
    /* keep mechanisms clear */
    if ((rightrearpos) > (rightrearaep - (rightrearaep - rightrearpep)/10))
        /* keep mechanisms clear */
    {
        rightrear2 = mechanism2;
        rightrearcross2 = crossmechanism2*contbias;
    }
    else
    {
        rightrear2 = 0;
        rightrearcross2 = 0;
    }
    if (rightrearpos < rightreartarget)
    {
        if(pos_rr_x.val < rightreartarget)
        {
            rightrearcontact = 0;

```

```

        rightrearend = rightrearpos;
    }
    else rightrearcontact = 1;
}
else
{
    rightrear5 = 0.5*mechanism5*((rightrearaep - rightrearpos)/(rightrearaep - rightrearpep));
    rightrearcross5 = 0.5*crossmechanism5*((rightrearaep - rightrearpos)/(rightrearaep -
rightrearpep))*contbias;
}

}
else
{
    rightrearpos = rightrearpos + swingspeed*timestep;
    rightrear1 = mechanism1;
    rightrearcross1 = crossmechanism1*contbias;
    rightrear2 = 0;
    rightrearcross2 = 0;
    rightrear5 = 0;
    rightrearcross5 = 0;
    rightrearz = (4*(rightrearzmax - rightrearground)/((rightrearaep - rightrearend)*(rightrearaep -
rightrearend))) * (rightrearpos - rightrearend)*(rightrearaep - rightrearpos) + rightrearground;
    if (rightrearpos > rightrearaep)
    {
        rightrearcontact = 1;
        rightrear2 = 0;
        rightrearcross2 = 0;
        rightrearcount = 1;
        rightrearz = rightrearground;
    }
}

}

/* -----end right rear section----- */

/* Now, reassign PEP target values with above network values */

leftfronttarget = leftfrontpep + rightfrontcross2*crossmechanismscale2 +
rightfrontcross5*crossmechanismscale5 + leftmid1*mechanismscale1 + leftmid2*mechanismscale2;

rightfronttarget = rightfrontpep + leftfrontcross2*crossmechanismscale2 +
leftfrontcross5*crossmechanismscale5 + rightmid1*mechanismscale1 + rightmid2*mechanismscale2;

leftmidtarget = leftmidpep + rightmidcross2*crossmechanismscale2 +
rightmidcross5*crossmechanismscale5 + leftfront5*mechanismscale5 + leftrear1*mechanismscale1 +
leftrear2*mechanismscale2;

rightmidtarget = rightmidpep + leftmidcross2*crossmechanismscale2 +
leftmidcross5*crossmechanismscale5 + rightfront5*mechanismscale5 + rightrear1*mechanismscale1 +
rightrear2*mechanismscale2;

leftreartarget = leftrearpep + rightrearcross1*crossmechanismscale1 +
rightrearcross2*crossmechanismscale2 + rightrearcross5*crossmechanismscale5 +
leftmid5*mechanismscale5;

```

```
rightreartarget = rightrearpep + leftrearcross1*crossmechanismscale1 +  
leftrearcross2*crossmechanismscale2 + leftrearcross5*crossmechanismscale5 +  
rightmid5*mechanismscale5;
```

```
position[0][0]=rightfrontpos;  
position[0][1]=rightfrontz;  
position[1][0]=rightmidpos;  
position[1][1]=rightmidz;  
position[2][0]=rightrearpos;  
position[2][1]=rightrearz;  
position[3][0]=leftfrontpos;  
position[3][1]=leftfrontz;  
position[4][0]=leftmidpos;  
position[4][1]=leftmidz;  
position[5][0]=leftrearpos;  
position[5][1]=leftrearz;
```

```
}
```

Bibliography

Bachmann, R. J., A Cockroach-Like Hexapod Robot fro Running and Climbing. M.S. thesis. 2000.

Berns, K., Grimminger, F., Hochholdinger, U., Kerscher, T. and Albiez, J. Design and Control of a Leg fro the Running Machine PANTER, *Proceedings of ICAR 2003, The 11th International Conference on Advanced Robotics* Coimbra, Portugal, Jung 30- July 3, 2003.

Choi, Jongung. Dynamic Simulation of Cockroach Climbing. M.S. Thesis, CWRU. 2000

Choi, J. U. , Rutter, B. L., Kingsley D. A., Ritzmann, R. E. and Quinn, R. D. A Robot with Cockroach Inspired Actuation and Control. (in Press)

Chou, Ching-Ping, Hannaford, B. Measurement and Modeling of McKibben Pneumatic Artificial Muscles, *IEEE Transactions on Robotics and Automation*, Vol. 12, Issue: 1 , pp. 90 – 102. Feb. 1996

Chou, Ching-Ping, Hannaford, B. Static and Dynamic Characteristics of Mckibben Pneumatic Artificial Muscles, *Proceedings of 1994 IEEE International Conference on Robotics and Automation*, Vol.1, pp. 281 - 286. May 1994

Clark, J., Cham, J., Bailey, S., Froehlich, E., Nahata, P., Full, R. and Cutkosky, M. Biomimetic Design and Fabrication of a Hexapedal Running Robot. *Proceedings of the 2001 IEEE International Conference on Robotics and Automation (ICRA 2001)*, Vol. 4, pp. 3643 – 3649. 2001

Colbrunn, Robb. Design and Control of a Robotic Leg with Braided Pneumatic Actuators, M.S. Thesis, CWRU. 2000

Cruse, H. What mechanisms coordinate leg movement in walking arthropods?, *Trends in Neural Science*, Vol. 13, pp. 15-21, 1990.

Dean, J., Brunn, D., Bartling, C., Cruse, H., Cymbalyuk, G., Dreifert, M. and Schmitz, J. Control of hexapod walking using artificial and real neurons. *BIONA-report 9*: 17 – 36. 1995

Delcomyn, F., Nelson, M.E. Architectures for a biomimetic hexapod robot. *Robotics and Autonomous Systems*, Vol. 30, pp. 5-15, 2000.

Dresden, D., and Nijenhuis, E. D. One the Anatomy and Mechanism of Motion of the Mesothoracic Leg of Periplaneta Americanan. *Proc. Kon. Ned. Akad. Wetensch. Ser. C.*, 56:39-47.

Espenschied, K. S. and Quinn, R. D., Biologically-Inspired Hexapod Robot Design and Simulation, *AIAA Conference on Intelligent Robots in Field, Factory, Service and Space*, Houston, Texas, March 20-24, 1994.

Espenschied, K. S., Quinn, R. D., Chiel, H. J., and Beer, R. D., Leg Coordination Mechanisms in Stick Insect Applied to Hexapod Robot Locomotion, *Adaptive Behavior*, Vol. 1, No. 4, pp. 455-468, 1993.

Gaylord, R.H., 1958, United States Patent 2,944,126.

Kerscher, T., Albiez, J., Zoellner, J. M. and Dillmann, R. AirInsect – A New Innovative Biological Inspired Six-Legged Walking Machine Driven by Fluidic Muscles, Proceedings of IAS 8, The 8th Conference on Intelligent Autonomous Systems, Amsterdam, The Netherlands, 10-13 March 2004

Kingsley, D. A., Quinn, R. D., Ritzmann, R. E. A Cockroach Inspired Robot With Artificial Muscles, *International Symposium on Adaptive Motion of Animals and Machines (AMAM 2003)*, Kyoto, Japan.

Kingsley, Daniel. A Cockroach Inspired Robot with Artificial Muscles, Ph. D. dissertation. Case Western Reserve University, January 2005.

Klute, G.K., Czerniecki, J.M. and Hannaford, B. McKibben artificial muscles: pneumatic actuators with biomechanical intelligence, *Proceedings. 1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 221 – 226, Sept. 1999

Martin-Alvarez, A., De Peuter, W., Hillebrand, J., Putz, P., Matthyssen, A. and De Weerd, J. F. Walking robots for planetary exploration missions. *Second World Automation Congress (WAC '96)*, Montpllier, France, May 27-30, 1996.

Mussa-Ivaldi, F. A. and Morasso, P. Patterns of interarticulator phasing and their relation to linguistic structure. *Biological Cybernetics*, Vol. 60, pp. 1-16, 1988.

Mussa-Ivaldi, F.A., Hogan, N. Integrable Solutions of Kinematic Redundancy via Impedance Control. *Int. J. of Robotics Research*, Vol. 10, No. 5, pp. 481-491, October, 1991.

Nelson, Gabriel. Learning about control of legged locomotion using a hexapod robot with compliant pneumatic actuators, Ph. D dissertation. May 2002.

Nelson, Gabriel. Modeling and Simulation of an Insect-like Hexapod, Master thesis. Case Western Reserve University, January 1995.

Pack, R.T., Christopher Jr, J.L. and Kawamura, K. A Rubbertuator-Based Structure-Climbing Inspection Robot, *Proceeding of the 1997 IEEE International Conference on Robotics and Automation*, pp. 1869- 1874, Alguquerque, New Mexico, April 1997.

Pfeiffer, F., Eltze, J. and Weidemann, J.-J. The TUM-walking machine. *Intelligent Automation and Soft Computing*, Vol. 1, pp. 307-323, TSI Press Series, 1995.

Pneumatic Catalog 2004 – Pneumatic Muscle, <http://www.festo.com>

Quinn, R.D., Nelson, G.M., Ritzmann, R.E., Bachmann, R.J., Kingsley, D.A., Offi, J.T. and Allen, T.J. Parallel Strategies For Implementing Biological Principles Into Mobile Robots, *Int. Journal of Robotics Research*, Vol. 22 (3) pp. 169-186. 2003

Sarnli, U., Buehler, M. and Koditschek, D. E. RHex: a Simple and highly mobile hexapod robot. *The International Journal of Robotics Research*, 20 (7) : 616-631, July 2001.

Schroer, R. T. Body Motions and Climbing Abilities of a Whegs Robot with Cockroach Comparisons, M.S. thesis. Case Western Reserve University, January 2004.

Tondu, B., Lopez, P. Modeling and Control of McKibben Artificial Muscle Robot Actuators, *IEEE Control Systems Magazine*, Vol. 20, pp. 15-38. April 2000.

Tondu, B., Boitier, V. and Lopez, P. Naturally compliant robot-arms actuated by McKibben artificial muscles, *1994 IEEE International Conference on Humans, Information and Technology*, Vol. 3, pp. 2635 – 2640. 2-5 Oct. 1994

Van der Smagt, P., Groen, F. and Schulten, K. Analysis and control of a rubber-tuator arm, *Biological Cybernetics*. 75, 433-440, 1996.

Watson, J. T., Ritzmann, R.E. Leg kinematics and muscle activity during treadmill running in the cockroach, *Blaberus discoidalis* : I. Slow running. *J. Comp. Physiol., A* 182: 11-22, 1998.

Ye, N., Scavarda, S. Betemps, M. and Jutard A. Models of a pneumatic PWM solenoid valve for engineering applications, *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, Vol. 114, pp. 680-688. Dec 1992

Yobotics! Simulation Construction Set. Yobotics, Inc. October 19, 2001