

January 2012

The Creation of a Robotics Based Human Upper Body Model for Predictive Simulation of Prostheses Performance

Derek James Lura

University of South Florida, rakinind@yahoo.com

Follow this and additional works at: <http://scholarcommons.usf.edu/etd>

 Part of the [American Studies Commons](#), [Medicine and Health Sciences Commons](#), and the [Robotics Commons](#)

Scholar Commons Citation

Lura, Derek James, "The Creation of a Robotics Based Human Upper Body Model for Predictive Simulation of Prostheses Performance" (2012). *Graduate Theses and Dissertations*.
<http://scholarcommons.usf.edu/etd/4133>

This Dissertation is brought to you for free and open access by the Graduate School at Scholar Commons. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact scholarcommons@usf.edu.

The Creation of a Robotics Based Human Upper Body Model for
Predictive Simulation of Prostheses Performance

by

Derek James Lura

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Mechanical Engineering
College of Engineering
University of South Florida

Major Professor: Rajiv Dubey, Ph.D.
William Lee, Ph.D.
Craig Lusk, Ph.D.
Kyle Reed, Ph.D.
M. Jason Highsmith, D.P.T.
Stephanie Carey, Ph.D.

Date of Approval:
March 2, 2012

Keywords: Inverse Kinematics, Compensatory Motion, Activities of Daily Living (ADL),
Range of Motion (RoM), Amputee, Motion Planning

Copyright © 2012, Derek James Lura

Dedication

I would like to dedicate this dissertation to my daughter Maia, my wife Rachel, and my parents Glenn and Lolly.

Acknowledgements

This research and development project was conducted at the University of South Florida and was made possible by a research grant that was awarded and administered by the U.S. Army Medical Research & Materiel Command (USAMRMC) and the Telemedicine & Advanced Technology Research Center (TATRC), at Fort Detrick, MD, under Contract Number: W81XWH-10-1-0601

The views, opinions and/or findings contained in this publication are those of the author and do not necessarily reflect the views of the Department of Defense and should not be construed as an official DoD/Army position, policy or decision unless so designated by other documentation.

I would like to thank my advisor Rajiv Dubey as well as my committee members William Lee, Craig Lusk, Kyle Reed, M. Jason Highsmith, and Stephanie Carey for their guidance and support. Finally, I would also like to acknowledge all of the work done by my fellow researchers at the Rehabilitation Robotics and Prosthetics Testbed (RRT), and the subject participants. Without all of their help I would not have been able to complete this dissertation.

Table of Contents

List of Tables.....	iv
List of Figures	vi
Abstract.....	ix
Chapter 1: Introduction.....	1
1.1 Performance Measures for Modern Prostheses	2
1.2 Epidemiology and Need	3
1.3 Current Upper Limb Prescription Techniques.....	5
1.3.1 No Prosthesis	6
1.3.2 Passive Function	7
1.3.3 Body-Powered Prostheses	7
1.3.4 Externally Powered Systems	8
1.3.5 Hybrid Systems.....	9
1.3.6 Activity Specific	9
1.4 Human Body Modeling	10
1.5 Functional Joint Center Modeling.....	13
1.6 Robotic Optimization Techniques for Modeling	16
1.6.1 Jacobian Based Control Algorithms	18
1.6.2 Neural Network Based Control Algorithms.....	21
1.6.3 Probability Based Control Algorithms.....	23
1.7 Previous Work by the Author in Upper Body Simulation	23
1.7.1 Brief Detail of Previous Methods	24
1.7.2 Previous Results	25
1.7.3 Limitations of Previous Study	25
1.8 Summary of the RHBM.....	26
1.9 Dissertation Overview	28
Chapter 2: Subject Motion Capture and Measurement	30
2.1 Subject Demographics.....	31
2.2 Braced Subjects.....	32
2.3 Anatomical Measurements	32
2.4 Motion Capture	33
2.5 Range of Motion Tasks	35
2.6 Activities of Daily Living.....	36
Chapter 3: Determining Functional Joint Centers and Upper Body Segments.....	37
3.1 Importing Data from Motion Capture	38
3.2 Segment Definitions and Joint Centers	40

3.2.1 Pelvis	40
3.2.2 Torso	42
3.2.3 Shoulder	43
3.2.4 Upper Arm.....	44
3.2.5 Forearm	45
3.2.6 Hand.....	46
3.3 Determining Denavit and Hartenberg Parameters and RHBM Joint Angles.....	47
3.4 Clinical Joint Angles	50
3.4.1 Rotational Conventions.....	50
3.5 Saving the Model Data	55
Chapter 4: Motion Analysis and Segment Length Results	56
4.1 Control Subjects' Range of Motion	56
4.1.1 Braced Subjects' Range of Motion.....	57
4.2 Amputee Subjects' Range of Motion	59
4.2.1 Subject R01	60
4.2.2 Subject H01	60
4.2.3 Subject H02	61
4.2.4 Subject H03.....	62
4.3 Activities of Daily Living Results and Observations.....	63
4.3.1 Brushing Hair	64
4.3.2 Drinking From a Cup.....	65
4.3.3 Eating With a Knife and Fork	66
4.3.4 Lifting a Laundry Basket	67
4.3.5 Opening a Door	68
4.4 Subject Measurements.....	69
4.5 Functional Joint Center Segment Geometry.....	70
4.6 Comparison with Vicon Plug-In Gait.....	73
Chapter 5: Methods for Predicting Human Motion.....	77
5.1 Training Data Filtering and Preprocessing	78
5.2 Defining Error.....	79
5.3 Robustness of Methods.....	80
5.4 Least Norm Solution (LN).....	81
5.5 Weighted Least Norm (WLN)	83
5.6 Probability Density Gradient Projection (GP).....	86
5.7 Artificial Neural Network (NN).....	87
5.8 Combined Methods	88
5.8.1 Neural Network with Weighted Least Norm Correction (NN+WLN)	89
5.8.2 Global Weighted Least Norm with Probability Density Correction (GP+WLN)	89
Chapter 6: Motion Prediction Results and Analysis of Error	90
6.1 Analysis of Least Norm Solution Error.....	91
6.1.1 Brushing Hair	92

6.1.2 Drinking From a Cup	93
6.1.3 Eating With a Knife and Fork	94
6.1.4 Lifting a Laundry Basket	95
6.1.5 Opening a Door	96
6.2 Weighted Least Norm	97
6.2.1 WLN Robustness	99
6.3 Probability Density Gradient Projection (GP)	99
6.3.1 GP Robustness	101
6.4 Neural Network.....	102
6.4.1 NN Robustness	103
6.5 Neural Network with Weighted Least Norm Correction	104
6.6 Global Weighted Least Norm with Probability Density Correction	105
6.7 Braced Subject Testing	106
6.7.1 Braced Weighted Least Norm Testing.....	107
6.7.2 Braced Neural Network Testing	108
6.7.3 Braced Probability Density Gradient Projection Testing.....	109
6.8 Analysis of Distribution of Error	111
6.8.1 Joint Angle Distribution of Error.....	111
6.8.2 Task Based Comparison of Methods	111
Chapter 7: Discussion and Future Work.....	113
7.1 Discussion.....	114
7.1.1 Contributions to the State of the Science	116
7.1.2 Significance of Errors	117
7.1.3 Limitations	118
7.2 Future Work.....	119
7.2.1 Integration and Verification with Additional Amputee Subject Data	119
7.2.2 Dynamic Analysis.....	119
7.2.3 Residual Limb Interface.....	119
7.2.4 3D Visualization	120
7.2.5 Graphical User Interface	120
References	121
Appendices.....	130
Appendix A: Data Collection Documents.....	131
Appendix B: Matlab Code.....	133

List of Tables

Table 1: Upper extremity prosthesis rejection rates for adults, reproduced from [1]	2
Table 2: Motions of the 15 DoF upper limb model [15-17]	24
Table 3: Limitations of previous studies and solutions	26
Table 4: Segment and joint definitions of RHBM	27
Table 5: Subject demographic data	31
Table 6: Anthropometric measurement names	32
Table 7: Residual limb measurements	33
Table 8: Marker descriptions	34
Table 9: Subject Instructions for RoM tasks.....	35
Table 10: Description of ADLs.....	36
Table 11: Description of Denavit and Hartenberg parameters.....	48
Table 12: Denavit and Hartenburg parameters	49
Table 13: Conversion between joint angle conventions (radians)	54
Table 14: Range of motion for control subjects (degrees).....	57
Table 15: Range of motion of braced control subjects (degrees).....	58
Table 16: Control subject anthropometric measurements (cm)	69
Table 17: Amputee subject residual limb measurements (cm)	70
Table 18: Segment geometry parameters from function joint centers (cm)	70
Table 19: R ² correlations for segment lengths	71
Table 20: Average difference between joint angle conventions (degrees).....	74

Table 21: Variation in Plug-in Gait segment lengths for RoM tasks (mm).....	75
Table 22: Data distribution for robustness testing.....	80
Table 23: Brief summary of primary methods and results	90
Table 24: Right arm RMS subject error for LN solution (degrees)	91
Table 25: Right arm RMS task error for LN solution (degrees)	92
Table 26: RMS error by subject for optimized weights (degrees)	97
Table 27: RMS error by task for optimized weights (degrees).....	98
Table 28: WLN RMS subject error for braced and un-braced subjects (degrees)	107
Table 29: Global control and braced inverse weights for the dominant arm.....	108
Table 30: NN RMS subject error for braced and un-braced subjects (degrees)	108
Table 31: GP RMS subject error for braced and un-braced subjects (degrees).....	110
Table 32: Comparison of methods task RMS error (degrees)	112
Table 33: Predicted number of subjects for convergence.....	112
Table 34: Review of tested methods	115

List of Figures

Figure 1: Hosmer silicon gloves.....	7
Figure 2: Hosmer body-powered hook and elbow.	8
Figure 3: Diagram of Utah 3 prosthetic arm.....	8
Figure 4: Ideal functional joint centers: circle fit method (left), and instant center of rotation (right)	15
Figure 5: A two DoF robotic manipulator	16
Figure 6: Example NN with one hidden layer.	21
Figure 7: Diagram of the RHBM kinematics (axes top, lengths bottom).....	28
Figure 8: Diagram of the data flow during development of the RHBM	28
Figure 9: RHBM file directory setup.....	39
Figure 10: Diagram of the pelvis definitions	41
Figure 11: Diagram of torso segment definitions.....	42
Figure 12: Diagram left and right shoulder segment definitions	44
Figure 13: Diagram of left and right the upper arm segments	45
Figure 14: Diagram of the forearm segments	46
Figure 15: Diagram of the hand segments	47
Figure 16: Matlab plot of robot [90] object for subject C03.....	48
Figure 17: Impace of bracing on range of motion.....	59
Figure 18: RoM of subject RH01 (blue) superimposed over control RoM (red).....	60
Figure 19: RoM of subject H01 (blue) superimposed over control RoM (red)	61
Figure 20: RoM of subject H02 (blue) superimposed over control RoM (red)	62
Figure 21: RoM of subject H03 (blue) superimposed over control RoM (red)	63

Figure 22: Impact of bracing on dominant arm for brushing task	64
Figure 23: Impact of bracing on dominant arm for drinking task.....	65
Figure 24: Impact of bracing on dominant and non-dominant arm for eating task	66
Figure 25: Impact of bracing on dominant and non-dominant arm for lifting task	67
Figure 26: Impact of bracing on dominant arm for opening task.....	68
Figure 27: Left elbow flexion for functional joint center and Plug-in Gait.....	75
Figure 28: Plug-in Gait abnormality and associated variation in segment length	76
Figure 29: Neural network diagram	88
Figure 30: Upper arm rotation (left) and torso flexion (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz) for recorded data and least norm solution for brushing hair task, subject C04	92
Figure 31: Upper arm rotation (left) and torso flexion (right) joint angles (top) and rotational velocity (bottom), recorded data and least norm solution, drinking task, subject C01	93
Figure 32: Elbow flexion (left) and forearm pronation (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, eating task, subject C05	94
Figure 33: Torso flexion (left) and upper arm flexion (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, lifting task, subject C05	95
Figure 34: Torso flexion (left) and upper arm rotation (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, opening task, subject C02	96
Figure 35: Density function for joint 1 (torso flexion).....	100
Figure 36: Inverse density and gradient function for joint 1 (torso flexion)	100
Figure 37: GP accuracy vs. division of end effector space.....	101
Figure 38: Robustness of the GP method	102
Figure 39: Effect of network size on bilateral NN performance.....	103

Figure 40: Robustness of the NN approximation.....	104
Figure 41: Robustness of NN+WLN method	105
Figure 42: Robustness of the GP+WLN method	106
Figure 43: RMS error of each joint, C01-C05 included, C06-C10 excluded	111
Figure 44: Diagram of upper body prosthesis simulation tool.....	113
Figure 45: Diagram of simulation function	114

Abstract

This work focuses on the use of 3D motion capture data to create and optimize a robotic human body model (RHBM) to predict the inverse kinematics of the upper body. The RHBM is a 25 degrees of freedom (DoFs) upper body model with subject specific kinematic parameters. The model was developed to predict the inverse kinematics of the upper body in the simulation of a virtual person, including persons with functional limitations such as a transradial or transhumeral amputation. Motion data were collected from 14 subjects: 10 non-amputees control subjects, 1 person with a transradial amputation, and 3 persons with a transhumeral amputation, in the University of South Florida's (USF) motion analysis laboratory.

Motion capture for each subject consisted of the repetition of a series of range of motion (RoM) tasks and activities of daily living (ADLs), which were recorded using an eight camera Vicon (Oxford, UK) motion analysis system. The control subjects were also asked to repeat the motions while wearing a brace on their dominant arm. The RoM tasks consisted of elbow flexion & extension, forearm pronation & supination, shoulder flexion & extension, shoulder abduction & adduction, shoulder rotation, torso flexion & extension, torso lateral flexion, and torso rotation. The ADLs evaluated were brushing one's hair, drinking from a cup, eating with a knife and fork, lifting a laundry basket, and opening a door. The impact of bracing and prosthetic devices on the subjects' RoM, and their motion during ADLs was analyzed.

The segment geometries of the subjects' upper body were extracted directly from the motion analysis data using a functional joint center method. With this method there are no conventional or segment length differences between recorded data segments and the RHBM. This ensures the accuracy of the RHBM when reconstructing a recorded task, as the model has the same geometry as the recorded data. A detailed investigation of the weighted least norm, probability density gradient projection method, artificial neural networks was performed to optimize the redundancy RHBM inverse kinematics. The selected control algorithm consisted of a combination of the weighted least norm method and the gradient projection of the null space, minimizing the inverse of the probability density function. This method increases the accuracy of the RHBM while being suitable for a wide range of tasks and observing the required subject constraint inputs.

Chapter 1: Introduction

The objective of this study was to develop the RHBM into a kinematically accurate model of the upper body, with the ability to predict the subjects' pose during activities of daily living. The RHBM must also be suitable for use in simulating the motion of persons with limited functional capabilities, specifically persons with transhumeral or transradial amputations. This model can then be used in a simulation of prostheses performance to prospectively determine patient outcomes, evaluate the performance of different devices, design new prosthetic devices, and better train patients to use their prostheses. To facilitate this work the following research objectives were identified:

1. Evaluation of the range of motion and task performance of persons wearing braces and amputees using prosthetic devices.
2. Creation of database of subject upper body poses during activities of daily living.
3. Development of subject specific parameters to create a highly accurate model of the upper body.
4. Development and investigation of a variety of inverse kinematic control algorithms, and their application in the field of human motion prediction.

By modeling the upper body and applying that model to the field of prosthetics the performance of devices can be quantitatively and objectively measured. Quantitative measures of prosthesis performance will help the prescription, evaluation, design, and training associated with these devices. Improvement in each of these areas would lead to more independence and a better quality of life for prosthesis users.

1.1 Performance Measures for Modern Prostheses

In prosthetic research there is currently a gap in the ability to predict the prospective outcome of an amputee's ability to become fully proficient with and regularly use a prosthetic device. Additionally, rejection and non-wear rates of upper extremity prostheses are high, as shown in Table 1, and there is need for further study to determine the "comprehensive understanding of the factors affecting prosthesis use and abandonment" [1]. Recent review of prosthetic outcomes measures [2, 3] found that of the existing measures the Assessment of Capacity for Myoelectric Control (ACMC) [4], the Orthotics and Prosthetics Users' Survey (OPUS) [5], and the Trinity Amputation and Prosthesis Experience Scales (TAPES) [6], were recommended when measuring outcomes of an adult amputee population. These tools will help to evaluate the efficacy of prosthetic devices; however incorporation of simulation can lead to better prediction and optimization of prosthetics outcomes and can be quickly applied to clinical knowledge.

Table 1: Upper extremity prosthesis rejection rates for adults, reproduced from [1]

	# of Studies	Mean (%)	Range (%)	S.D. (%)
Passive	1	38	-	-
Body-Powered	3	45	36-66	17
Electric	12	32	12-75	19
No Prosthesis	7	16	6-34	11

Currently a wide body of literature exists on tracking and modeling the human body [7-14]. The development of tools for simulating the efficacy of prosthetic devices can be achieved using techniques developed for robotics and biomechanics [15-17]. This work seeks to contribute to that body of knowledge by developing an upper body model suitable for predicting patient outcomes through simulation, to improve the efficacy of upper extremity prostheses. The implementation of the RHBM into simulation software

will be completed as part of the ongoing research project “Development of a Simulation Tool for Upper Extremity Prostheses” at the University of South Florida funded by the U.S. Army Medical Research & Materiel Command (USAMRMC) and the Telemedicine & Advanced Technology Research Center (TATRC). This simulation will be used to evaluate the efficacy of different devices based on predictions of a subject’s task performance relative to healthy persons without an amputation. This information can then be used to assist in the determination of which prosthesis is best for a particular individual (prescription), which prosthesis is optimal for specific tasks (evaluation), determine the efficacy of potential prosthetic components and capabilities (design), and effective strategies for prosthesis use (training).

1.2 Epidemiology and Need

Of the estimated 1.6 million persons with amputation in the United States in 2005, 35% are living with loss or deficiency of the upper extremity [18]. The number of amputees is expected to increase to 2.2 million by 2020. According to data from the Joint Theater Trauma Registry and Military Amputee Research program, there have been 423 service members who have suffered one or more major limb amputation in the period between October 2001 and June 2006. Of those, 105 have had an upper extremity amputation “at or proximal to the wrist” [19]. A 2010 article cited that more than 950 soldiers have sustained combat-related amputation during the current conflicts [20]. In 1993 Silcox reported prosthesis rejection rates for upper extremity myoelectric prostheses of up to 50% and that only about 25% would rate themselves as excellent prosthesis users [21]. Due to the wide variety of prosthetic types, amputation levels, and user preferences, reported use and abandonment vary widely [1]. Richard Sherman studied traumatic

amputees in the VA and found that 22% said the prosthesis was “not useful for anything” and only 32% reported the prosthesis was up to half as effective as the original limb [22], although the rates for the upper limb specifically were not identified. In addition to those that reject the use of a prosthetic device, there is a group that chooses to wear the device but only use it passively [1]. Upper limb amputees are also less likely to use a prosthesis than lower-limb amputees [23]. A 2007 survey of prosthesis users in Sweden and the UK found high levels of satisfaction from users of upper limb cosmetic and electric prostheses, but did not account for non-users [24]. An online survey found that users with a myoelectric prosthetic hand use their prosthesis more for work than recreation, but generally reported high levels of use [25]. Clearly, while improvements are being made in use and satisfaction with prosthetic devices, the current generation of powered upper limb prostheses is not serving the population as effectively as possible. Emerging prosthetic devices offer increased capabilities, but are also increasingly complex, and the costs of these devices are increasing exponentially. Methods for maximizing the capabilities of devices, and determining the advantages and the disadvantages of additional components, will become increasingly important to ensure the efficacy of these devices. Increased efficacy in the development, prescription, and utilization of new devices will lead to greater patient satisfaction and renewed desire for continued development.

It has been shown that a variety of different solutions are required for individuals with upper extremity amputations depending on their perceptions and goals [26]. The role of the amputee in selecting the device and the timeliness of delivery are significant factors in prosthesis acceptance [1]. Even a small change in the artificial limb can have

significant impact on the overall body movements, [27] and ultimately lead to a reduction in the rate of use of the intact arm and body, possibly reducing overuse injuries. Limited function of upper limb prostheses may cause awkward aberrant movements not normally experienced by non-amputees, called compensatory motion [28, 29]. These aberrant motions have been cited as one of the factors influencing the discontinuation of prosthetic use [21]. Quantification and predictions of compensatory motions can help assess design changes and patient-training methods for the upper limb prosthesis in a functional context. Quantifying the underlying aspects of prosthesis performance can also lead to significant improvement in prosthesis selection and design.

1.3 Current Upper Limb Prescription Techniques

Contemporary prescription and selection of components for upper extremity prostheses have limited objective quantitative aspects. Prescription of prostheses commonly relies on the qualitative knowledge and experience of the prosthetist. For instance, if a person with an upper extremity amputation has extensive periscapular muscular impairment coupled with severe postural defects, then limited range of motion would suggest that a body-powered shoulder harness prosthesis would be a poor option. Similarly, prescription of a two site myoelectric prosthesis with co-contraction switching for a patient who is unable to activate the radial nerve distal to the elbow would likely be viewed as over-prescription, as their ability to properly control the device would likely be limited. The latter example has further implications in terms of surgical decisions regarding limb length. Battlefield surgical decisions for residual limb length may at times include component considerations without knowledge of potential patient satisfaction and function, which could potentially lead to future device abandonment. Abandonment in

this particular case may be due to the patient's perception of a poor functioning prosthesis. However, this may not be an issue of poor prosthetic function, but rather one of an inappropriate prosthetic prescription.

Current prosthetic prescription practices are based largely on a practitioner's clinical experience and their experience with commercially available components. The commercial sector impact from manufacturer marketing likely influences component prescription. This is plausible because prosthetists' perceptions of component function may be based on marketing claims. Implementation of this research could help prosthetists validate the function of devices from the commercial sector and develop opinions of performance independent of the component's marketing information. Upper limb prostheses are generally subdivided and selected from the following major categories; no prosthesis, passive, body-powered, externally powered, hybrid, or activity specific [30]:

1.3.1 No Prosthesis

Patients who feel that the prosthesis impairs function, does not provide sufficient function, or lacks cosmetic appeal are likely to not use a prosthesis. Additionally patients may not use a prosthesis if they lack the motor skills or cognitive ability to do so, or if the use of the device presents a risk of injury. Many users will choose not to use a prosthesis during specific activities such as: sleeping, bathing, or even recreational or work activities for which their prosthesis is not useful. While choosing to not use a prosthesis provides no additional functionality to the residual limb it also allows the full range of motion of the proximal joints, which patients may be able to utilize for functional performance.

1.3.2 Passive Function

Cosmetic and passive devices are often considered when pre-posing the terminal device is sufficient, or if psychosocial domains may benefit by restoring shoulder and extremity symmetry. They are also considered if the visibility of a high quality cosmetically replicated hand increases satisfaction, and social/societal reintegration. Passive devices do not offer additional active DoFs, however they can be used to extend the residual limb and act as support when performing tasks. Poseable passive devices, ones with inactive DoFs, may also be used to carry or hold objects. Passive devices may be desirable in tasks that require high levels of stability.

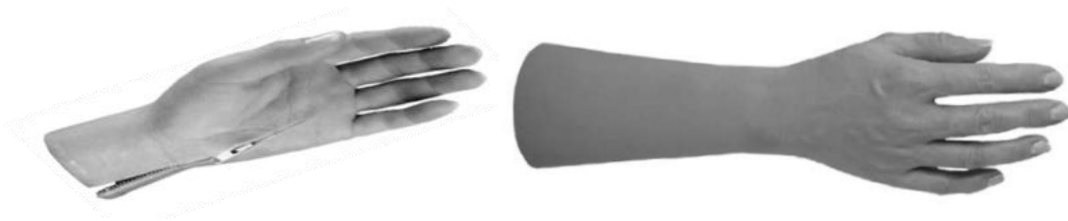


Figure 1: Hosmer silicon gloves

1.3.3 Body-Powered Prostheses

Body-powered prostheses are most commonly cable driven and generally require moderate scapular and shoulder muscle force production coupled with considerable scapular and humeral excursion. These prostheses should be considered if an individual's functional tasks create situations that are potentially damaging to the electronics associated with externally powered componentry such as vocation and recreation in oceanic environments, welding, and others. Most body-powered devices offer an active elbow and/or end effector, often used in combination with a hook. Passive joints for rotation of the end effector can also be included in the prosthesis.



Figure 2: Hosmer body-powered hook and elbow.

1.3.4 Externally Powered Systems

Incorporating external power commonly requires myoelectric signaling. Therefore a minimal amount of peripheral nerve activation is required in order to operate even the most simplistic (e.g. single channel “cookie crusher”) myoelectric prostheses. The increased control capability of the user (i.e. co-contraction, isolation, proportional control, etc.) enables a greater number of DoFs and separate functions that are available for the user. Nerve function, fatigue, added mass, battery life, maintenance, cost, compliance with instruction, environmental conditions, and gadget tolerance are also commonly considered. Externally powered systems have the most versatile range of available DoF, components exist to mimic almost all anatomical joints. Recent advances in robotic prosthetics have led to prosthetic arms with nearly the same capabilities of an anatomical arm. However, the mechanisms for control of these devices have not matured and traditional myoelectric control often only allows for a few control sites.

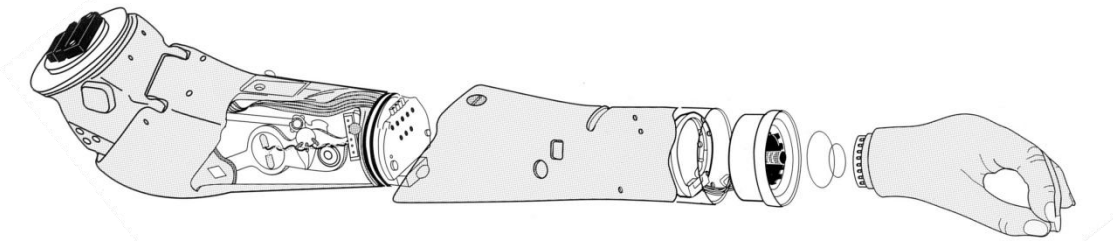


Figure 3: Diagram of Utah 3 prosthetic arm

1.3.5 Hybrid Systems

Hybrid systems offer combined control strategies and functions from both body-powered and externally powered systems. This is considered when maximal function is not attainable from a single activation system alone, often because of a patient's unique dysfunction and residual anatomy. Hybrid prostheses may combine passive, body-powered, and externally powered components to offer a device specific to the needs of an individual. This level of components selection is one of the potential areas of application for the prosthesis simulation tool.

1.3.6 Activity Specific

Activity specific prostheses are designed for performing a single specific task. They are commonly used in recreational settings but may also be used in occupational or other settings. Making a prosthesis activity specific may be as simple as exchanging an all-purpose terminal device for a highly specialized single task terminal device. Examples include terminal devices specific for: eating, hygiene, gardening, weightlifting, kayaking and more [28].

As observed above, the background structure for clinical device selection is largely based on subjective experience instead of guidelines or algorithms based on scientific evidence. Once one of the aforementioned general categories of prostheses has been prescribed, there is little data to confirm the success of the prescription. The successful prescription of a prosthesis should be confirmed by objective outcome measures such as higher function, increased satisfaction, decreased compensatory movement, decreased prosthetic abandonment rates, and decreased secondary complications (i.e. overuse syndromes) in the long term. Work is currently being done on the development of upper limb prosthetic

outcomes and standardization of outcome measures [2]. A paradigm for clinical decision making for orthoses has been developed [31]. A prescription criterion for lower limb prostheses is often based on Medicare Functional Classification Level, or other insurance guidelines. However, comparative analysis of lower limb function and outcomes for prosthetic knees have been explored [32, 33], but little is currently known about the prescription success and function of upper limb prostheses.

By developing a system to test the functional capacity of subjects fitted with a variety of components the simulation tool for upper extremity prosthesis will evaluate the impact of a variety of prosthetic components, by translating the components into kinematic parameters that the RHBM can then use to predict subject performance. The desired effect of which will give prosthetist an objective measure of predicted patient outcomes that they can use in conjunction with their professional experience to maximize the compatibility of patients and the prescribed devices.

1.4 Human Body Modeling

Quantitatively analyzing the performance of prosthetic devices starts with the creation of a model of the human body. Many models have been used in the recent development of lower limb prostheses and orthoses. A dynamic musculoskeletal model was used to predict gait in rehabilitation [34]. A simple two-dimensional model has been used to predict the effect of ankle joint misalignment on calf band movement in ankle-foot orthoses. This model was able to predict these effects for a range of ankle angles without human testing [35]. Crabtree et al. developed a tunable ankle-foot orthosis model to predict torque from ankle angle and velocity and to identify plausible changes in muscle excitation and function in a walking simulation [36]. A spring-mass model has been used

in conjunction with a symmetry index to observe the effect of varying prosthetic height and stiffness on running biomechanics [37]. This method of using a model and symmetry index is a tool that evaluates the effects of changes in lower limb prosthetic prescriptions. A model has also been used to predict the effects of variations in prosthetic sagittal-plane alignment, mass distribution and foot selection [38]. While modeling has been very successful in lower limb prosthetics, there have not been as many attempts to apply similar methods to the upper limb. This is likely due to the increased complexity of the upper limb, relative to the lower limb, which requires complex modeling techniques and control methods.

Although upper body models have been rarely used in the field of prosthetics, the development of a human body model that behaves like a person has been studied in a wide variety of fields, from computer graphics [39] to rehabilitation [40]. These models differ greatly in their degree of complexity and configuration depending on their scope and application. Maurel developed a 3D kinematic and dynamic model of the upper body and detailed the scapular thoracic joint, modeling the scapula position as being constrained by a series of points on a surface approximating the thorax [41]. These constraints led to a biomechanically accurate depiction of scapular movement, but are difficult to decompose into a series of single DoF joints. De Groot and Brand developed a regression for predicting scapular movement based on the angle of the humerus relative to the torso [13], which has been used in biomechanical simulation by Holzbaur [42]. This reduces the complexity of their upper body simulation. However, in the prosthetic population, as well as other populations with dysfunction of the upper extremity, scapular movement is an important control and compensation strategy and should not be coupled

to humeral motion. Most human body models simplify anatomical joints into a combination of single DoF revolute and prismatic joints that are commonly used to represent serial robotic manipulators [15-17, 43-45], which increases the ease of applying robotics based control algorithms. For instance, the shoulder is often simplified as three revolute joints that have intersecting orthogonal axes. More detailed models are often used in biomechanics to simulate muscle action, and have articulations that resemble anatomical movement with greater accuracy, but these models require detailed knowledge of the path of the motion or the individual muscle forces [12, 46-48], and therefore are not useful for prediction. Most models of the upper body have some degree of redundancy, and use various methods to optimize their pose; however the level of redundancy is usually low. The use of an upper body model to predict human movements has been studied by Abdel-Malek et al. [43], but focused on predicting the path of the arm given a number of waypoints. The variety of models of the upper body leads to confusion about different conventions and joint configurations. The International Society of Biomechanics has attempted to generate standard conventions [8], and the SIMM [48] and openSIM.tk [47] projects have been adopted by a number of biomechanics researchers and have led to somewhat standardized practices, however there is yet to be an established gold standard.

Study of the upper limb, when movement of the torso and scapular are excluded, has been much more extensive [40, 44, 49-53] than study of the upper body. Upper limb models typically have up to seven DoF, and are generally considered grounded to the shoulder (glenohumeral joint center) [51]. Upper limb models for the analysis of task performance and development of prostheses were developed by Troncossi [45], but the

model was not verified with recorded data. An example of design methodology for the determination of the optimal prosthesis architecture for a unilateral shoulder disarticulation amputee was applied [44]. Another common solution to the upper limb inverse kinematic problem is to resolve the redundancy by adding a constraint to the model reducing the 7 DoF model to a 6 DoF model, this allows for a purely analytical solution of the 7 DoF arm. This has been done by optimizing the ‘swivel angle’ of the elbow [52], and by minimizing the upper arm elevation [53]. The limitation of most of these models is that they do not predict the motion of the entire upper body. Therefore they are not well suited for use in prediction of task performance when the torso and shoulder complex are likely to contribute to user motion.

Coupling modeling with motion analysis enables the verification and optimization of the model results. There are many methods and programs for tracking human motion [50, 54-57], and many for modeling human motion as discussed above. To ensure accurate results the motion analysis and modeling conventions must be closely linked. In this study the use of functional joint centers [58, 59], and a robotic as well as clinical joint angle convention, ensure compatibility between motion analysis and the RHBM.

1.5 Functional Joint Center Modeling

The analysis of human upper body kinematics is complicated by its large number of joints, and its range of movement. Complex biomechanical analysis of the human body relies on detailed geometric and musculoskeletal modeling, similar to the work of Lee et al. [46]. However, in modeling the human upper body for analysis in interactive and real time simulation, like those developed by Hauschild et al. [60], or while recording upper body or whole body motions, it is often necessary to limit the number and complexity of

joints used to model the human body. In these cases, simplifications of complex joint structures are often made. Segments are often assumed to be rigid, and have joint centers with fixed position in the coordinate systems of the proximal segment [61]. Commonly used motion analysis techniques, such as the Vicon Plug-in Gait [54], rely on the regression of joint centers based on approximated distances from anatomical landmarks. These regressive methods often use mean anthropometric measurements, such as those provided by Drills [62] or Winter [63], in combination with subject anthropometric measurements taken manually by a researcher to approximate joint center locations. These locations are subject to error from subject measurements, marker placement, and variations in subject skeletal geometry. They can also be difficult to validate and compare with other models.

Functional methods, [59] those relying on the path data from motion analysis of a subject for determining the location of joints within a system, have several advantages over traditional regressive methods. A functional joint center is the center of rotation of a body in space relative to another body. In the case that the bodies are only rotating relative to each other, this is also the position on the reference body where the distance from any point on the rotating body remains constant, as shown in Figure 4. The primary advantages of functional joint center methods are that they do not rely on pre-existing knowledge of a body's anthropometry, and markers can be placed anywhere on a rigid segment. Marker artifacts and skin movement will decrease the accuracy of the functional joint center calculation, but only in relation to the rest of the movement. If the volitional movement is much larger than the noise, the skin movement, and the other sources of error, the impact on the functional joint center location will be minimal. Whereas noise

and other sources of error will translate directly into movement and/or rotation of the segment in regressive models, such as the Plug-in Gait. Functional methods are therefore less susceptible to measurement error, marker placement error, and deviation in subject's relative limb lengths.

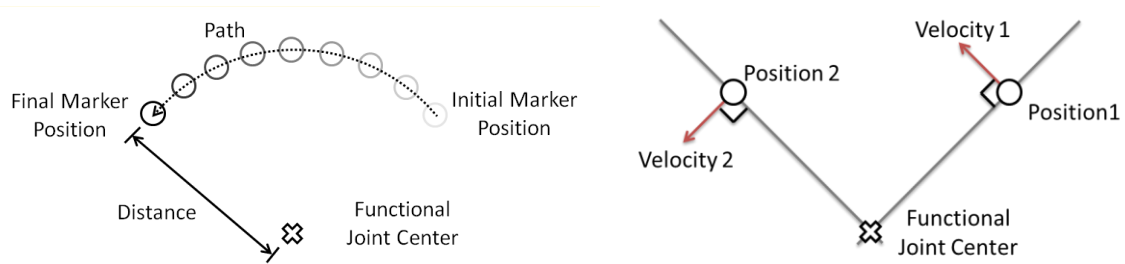


Figure 4: Ideal functional joint centers circle fit method (left), and instant center of rotation (right)

However, since the human body is not constructed of ideal hinges, no position exists on a segment of the upper body that will remain at a truly constant distance relative to all points on a distal or proximal segment. Therefore, it is necessary to find the position where the distance is nearly constant, and a sufficient amount of movement is required to discriminate relative segment motion from sources of error such as noise, segment deformation, and others. Several methods have been developed to predict a joint's center given a set of recorded position data. A least squares method has been developed [64], which provides computationally efficient solutions. An optimization algorithm for finding the joint center of the hip was developed [56]. A generalized gradient based optimization was also developed for automatic skeleton generation from motion analysis data [58]. These methods were tested for accuracy and noise tolerance, and the generalized gradient based optimization was selected for use with the RHBM.

1.6 Robotic Optimization Techniques for Modeling

The use of robotic methods to model the human body has been applied for various purposes, including 3D graphics, human engineering, biomechanics, and others. Robotic methods generally refer to the decomposition of a kinematic system into a series of single DoF joints, that can be used to calculate the forward and inverse kinematics of a system. For instance in Figure 5, a two DoF manipulator is presented. The forward kinematic equation, $fkine$, calculates the position of the end of the manipulator as a function of its joint angles, θ_1 and θ_2 . The inverse kinematic equation is the opposite of the forward kinematics where the joint angles are a function of the Cartesian position of the end of the manipulator, x and y .

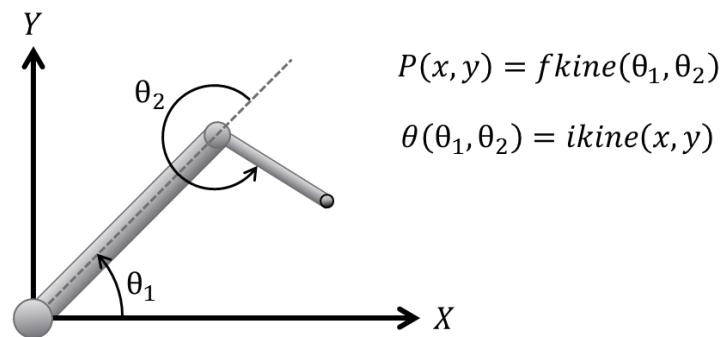


Figure 5: A two DoF robotic manipulator

Despite a great deal of research, the methodology of human movement has remained elusive. This is partially due to the fact that the human upper body is highly redundant. Redundancy is when the number of joints exceeds the number of controlled coordinates in the workspace, and the conventional inverse kinematics for a close-form solution is no longer applicable. The process of solving the redundancy of human poses remains a prominent topic of research. The use of the Jacobian, a mapping between joint angle and end effector velocity, for inverse kinematic control of redundant manipulators has been well studied [65-68], and the weighted least norm solution has been used in simulating

movement of the human upper body [15-17]. Additionally, Guez and Ahmad have shown that neural networks can be used in inverse kinematics problems for redundant robotics [69], and Kiguchi and Quan have used a fuzzy neural network for controlling an upper limb power assist exoskeleton [70].

The use of robotic methods to describe upper body kinematics was developed to facilitate the use of various control algorithms from robotics literature for the RHBM. The robotics literature contains many methods for controlling serial manipulators. Since the ideal control methodology was unknown, a wide variety of methods were considered. When controlling a robotic device, it is essential to compare the workspace capability of the robot and the task space required in operation. In general, a minimum of six DoFs are required in a robot in order to accomplish total manipulation control of objects in the workspace. Each side of the upper body model in the RHBM has 14 DoFs. Redundancy resolution and optimization has been the subject of a great deal of research, where the use of the extra joints is employed to execute additional tasks and optimize the motion based on certain performance criteria. Yang et al. developed a framework for multivariable optimization of a human model [71], where they minimized functions for joint displacement, changes in potential energy, and discomfort. However they did not use recorded data to optimize their cost equations for the reproduction of recorded motion, or test the realism of their generated poses.

In the RHBM, the redundancy of the model was used to minimize the difference between the model's predicted motion and the motion analysis data of persons performing ADLs. In this project several methods for optimizing the redundancy were tested. Control methods were divided into three categories for analysis. Jacobian based methods

compose the first category, of which the weighted least norm and null space projection methods were considered. Neural network based methods compose the second category, of which there are a wide variety of potential inputs and outputs. Finally the last category consists of probability based methods, primarily Gaussian processes, which provide a mapping between data sets. The final method developed was a combination of the weighted least norm solution with a null-space correction based on the gradient of probability density of the joint angles to predict joint movements that are preferable to human subjects.

1.6.1 Jacobian Based Control Algorithms

This section reviews several of the Jacobian based methods for controlling and optimizing redundancy that were explored during this study. These methods are generally extensions and applications of optimization of redundancy using Jacobian methods as outlined by Nakamura [67]. The Jacobian describes the mapping between joint angle velocity and end effector velocity and can be used to find methods for inverse kinematics and dynamics.

Chang [65] proposed a closed-form solution for inverse kinematics of redundant manipulators using the Lagrange multiplier method. He proposed an additional set of equations to resolve the redundancy at the inverse kinematic level in such a way that a given criteria function may be minimized or maximized. The additional equations were set in a similar way to the homogeneous solution term of the resolved rate method, which uses the null space to resolve the redundancy. He used the manipulability index [72] as the criteria function, but any criteria function can be used as long as the function can be reduced to an expression in terms of joint variables only.

Khadem et al. [66] used a global optimization scheme to avoid round obstacles using the resolved rate method and the null space of the Jacobian. Their simulation of a three-revolute-joint planar robotic arm has shown good performance in following a path while the specified robot link was avoiding a specified obstacle throughout the simulation.

Chan et al. [73] proposed a new method to resolve the redundancy and optimize for joint limit avoidance. They were able to control a 7-DoF robotic arm using a symmetric positive definite weight matrix that carries different weights for each joint of the redundant robot included in the least-norm solution. The weighted-least norm solution was implemented, and was able to reach the goal with the specified trajectory accurately and avoid the joint limits of the robotic arm. McGhee et al. [74] later used the weight matrix to avoid joint limits, singularities, and obstacles using the probability-based weighting of the performance criteria.

Beiner et al. [75] improved the velocity norms and the kinetic energy of their planar 3-DoF robotic crane with hydraulic actuators by using an improved pseudoinverse solution control scheme based on the weighted least norm methods. They used the initial manipulator configuration as an optimization parameter, and were able to reduce the actuator velocities obtained by a pseudoinverse solution and simultaneously avoid the actuators limits.

Zergeroglu et al. [76] designed a model-based nonlinear controller that achieved exponential link position and subtask tracking. Their control strategy used the pseudoinverse of the manipulator Jacobian and did not require the computation of the positional inverse kinematics. Their control strategy did not place any restriction on the

self-motion of the manipulator, and hence, the extra DoFs were available for their manipulability maximization, obstacle avoidance, and joint limits subtasks.

Kwon et al. [77] introduced a new method to optimize and resolve redundancy considering joint-limit constraint functions. Their dual quadratically constrained quadratic programming (QCQP) method used quadratic inequality constraints to approximate linear inequality constraints to represent joint position, velocity and torque bounds using the null space of the Jacobian. They were able to reduce the size of the problem by reducing the number of constraints and variables. They formulated the quadratic objective function and then converted the problem into two problems by eliminating linear equality constraints and by applying the duality theory. This method was used in their simulation of a 4-joint planar robotic arm, and they were able to reduce the computation time to about a tenth of that when the problem was not reduced.

Ellekilde et al. [78] created a new scheme for controlling robots in visual servoing applications. They employed quadratic optimization techniques to solve the inverse kinematics problem and explicitly handle both joint position, velocity and acceleration limits by incorporating these as constraints in the optimization process. Contrary to other techniques that use the redundant DoF to avoid joint limits, in their method they incorporated the dynamic properties of the manipulator directly into the control system to use redundancy to avoid joint velocity and acceleration limits. They used the joint position limits, velocity limits and acceleration limits by converting them into the velocity domain and chose the case of these limits that satisfied other limits as well for every time step within optimization function. The algorithm was tested by having a robot track a car that moved in a circle in the playing area. The quadratic programming control

system was robust with respect to singularities which enables the robot to track the car as “good as possible” even when it was out of reach.

The weighted least norm and gradient projection methods were combined to control a wheelchair mounted robotic arm [79]. This allows for the simultaneous control of the drive system and the robotic arm while optimizing for ADLs and overcoming workspace limitations. These methods can also be used to optimize the path of the wheelchair separately from the path of the end effector [80].

1.6.2 Neural Network Based Control Algorithms

An artificial neural network (NN) is a series of many simple functions that can be used to approximate a complex function. Networks are divided into layers with an input layer and output layer, and at least one hidden layer. The weighted sum of the previous layer becomes the input to one of the functions of the hidden layer. Typically the same function is used throughout a layer, referred to as the transfer function. The parameters of each equation of the functions within the network, called neurons, are tuned to optimize the performance of the network given a set of training data.

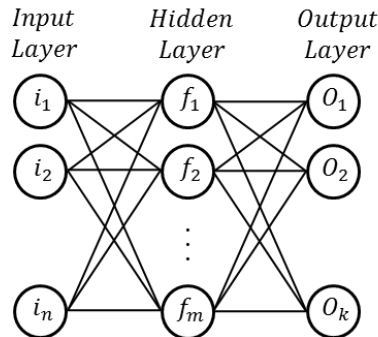


Figure 6: Example NN with one hidden layer.

Guez and Ahmad proposed to find a solution to robotic inverse kinematics using a neural network [69]. They found that the neural network produced adequate results and was

computationally efficient after training. Guez also notes that neural networks can be used to find solutions to inverse kinematics problems with no closed form solutions, including those of redundant manipulators. Josin et al. proposed the addition of a neural network to compensate for errors in an existing control algorithm by training the neural network with desired end effector positions and controller angle output, relative to the true angles required to achieve the desired positions [81].

Xia et al. have developed a parallel one layer neural network that they call the dual neural network, for the inverse kinematic control of redundant manipulators [82]. They have also further expanded this method to observe joint angle and velocity limits while minimizing complexity without needing to perform matrix inversion [83]. This method provides a computationally efficient and robust solution to the inverse kinematic equation that is also stable in all configurations.

In upper body research Kiguchi et al. have used a neuro-fuzzy network to optimize the weights of a weighted Jacobian torque controller for a robotic upper limb exoskeleton [70]. Kundu et al. have used a neural network to classify upper limb ADLs [84]. This method help the device to determine the user's intentions to determine the force the exoskeleton should apply to assist the user.

Inohira and Yokoi developed a neural network control of a prosthesis for bimanual manipulation tasks, solving for joint velocity of the prosthesis given the position of the contralateral arm and of the prostheses [85]. Ramirez-Garcia et al. used a neural network to control an upper arm prosthetic device by mapping desired joint angles to actuator lengths [86]. In these works the neural networks directly control the prosthetic device.

1.6.3 Probability Based Control Algorithms

Rasmussen and Williams [87] detail the advantages of Gaussian processes for machine learning. This is a somewhat newer methodology in the field of robotics and motion simulation but has been rapidly adopted. Gaussian processes can be used to create generic mappings between correlated variables, for instance; mapping of joint positions, velocities, and accelerations of a robotic arm to torques, and then using that mapping to calculate the torques required to move along a specified path.

Lee et al. [88] developed an algorithm for interactive control of avatars moving through a variety of terrains. They used principle component analysis to reduce the complexity of the motion in joint space, and a Markov chain to control the transitions between motions based on collected motion analysis data. Transitions between activities were then blended to ensure smooth movement.

Wei et al. [89] developed a physically constrained human model for animation. The model was developed using a Gaussian process to find a force vector field. This allowed for the addition of constraints in the force domain, and ensures the validity of the model when different segment masses were adapted. The techniques were then demonstrated by showing the model results when: walking with a heavy foot, running with forward resistance, walking on a slippery surface, and walking in a low gravity environment.

1.7 Previous Work by the Author in Upper Body Simulation

Although this study was built from the ground up, it was not the first attempt to make an upper body simulation for use in the evaluation of upper limb prostheses. In previous studies [15-17], the movement of the upper body while performing the tasks of opening a

door, drinking from a cup, turning a steering wheel, and lifting a box were evaluated using a 15 DoF robotic model. By applying various constraints to the model, it was shown that compensatory motions could be simulated in a virtual environment for unilateral [17] and bilateral [16] tasks. Work was also done to compare the simulated results to recorded trials [15]. This study was completed in Matlab and utilized the robotics toolkit developed by Peter Corke.

1.7.1 Brief Detail of Previous Methods

Previous development of an upper body simulation was completed in Matlab using the robotics toolkit [90]. Control over the range of motion of the model was performed by the use of a weighted inverse kinematic method, where the function of each joint can be controlled by a weighting parameter. Tasks were defined by the use of discrete end-effector positions and orientations along a path to form the desired motion. The 15 DoF model included the movements described in Table 2.

Table 2: Motions of the 15 DoF upper limb model [15-17]

Joint	Description
J1	Translation of the hip joint in the Z direction
J2	Translation of the hip joint in the Y direction
J3	Translation of the hip joint in the X direction
J4	Torso Bending Backward (+) / Forward (-)
J5	Torso Sideways Bending Right (+) / Left (-)
J6	Torso Rotation Left (+) / Right (-)
J7	Shoulder Complex Retraction (+) / Protraction (-)
J8	Shoulder Complex Depression (+) / Elevation (-)
J9	Upper Arm Adduction (+) / Abduction (-)
J10	Upper Arm Extension (+) / Flexion (-)
J11	Upper Arm Medial Rotation Inward (+)/Outward (-)
J12	Elbow Extension (+) / Flexion (-)
J13	Forearm Pronation (+) / Supination (-)
J14	Wrist Flexion (+) / Extension (-)
J15	Wrist Adduction (+) / Abduction (-)

Three configurations of the model were tested: an anatomical configuration, with all of the joints intact; a prosthesis with wrist rotation configuration where joints J14 and J15 were restricted from movement; and a prosthesis configuration where J13, J14, and J15 were restricted from movement.

1.7.2 Previous Results

The accuracy of the previous study was evaluated using joint angles calculated using Vicon Plug-In Gait and was found to have an average joint error of 7.35° and 5.22° for the right and left arm respectively when reconstructing control subject motion with task based weighted least norm control and no joint limit constraints. Implementation of the previous model was able to simulate the compensations of the upper body but resulted in over-exaggerated motions. While the model was able to predict compensatory motion the results were considered unrealistic. It was determined that to develop a clinically acceptable predictive model a large scale detailed analysis of upper body motion, and investigation of various control and constraint algorithms would need to be performed.

1.7.3 Limitations of Previous Study

Some of the following limitations were considered to be less significant, and were not addressed in this study. All segments were considered rigid bodies. This approximation was made because the relative motion of the joints with respect to deformation in the segment lengths was very large. Anatomical joints were approximated by constant centers of rotation, and segments with a large number of articulations were reduced into generalized movements with approximated joint centers. The functional joint centers have shown high accuracy when modeling the motions of the spine and shoulder complex, and the motions of the anatomical joints within these complexes are highly

coupled for most movement. Limitations of the previous studies [15-17], that are addressed in this study are given in Table 3.

Table 3: Limitations of previous studies and solutions

Limitation	Solution
A Limited number of tasks were analyzed.	Additional tasks were analyzed. The interface will help facilitate the addition of future tasks.
Some anatomical features were omitted; the model excluded the carrying angle of the elbow, and did not include any motions of the head.	Verification of the model with the Vicon motion analysis system was performed. The functional joint center model of the subjects provided nearly exact reconstruction of the recorded motion. Motion of the head does not affect the position of the hand and was omitted.
Each task was tested with only one gripping angle (the angle of the hand relative to the object being grasped). Changing the gripping angle will change the resulting compensatory motion.	Each task was analyzed on a subject basis and the performance was evaluated based on the movement of the subject. The gripping angle used by the subject was the angle at which the RHBM was tested. In simulation any gripping angle can be used within the task input parameters.
Each task was only performed with one trajectory; there are an infinite number of trajectories that can perform a similar task. Carey et al. [29] have shown that the trajectory used by a person with prosthesis varies from that of non-prosthesis users.	The RHBM was tested using multiple task trajectories from the recorded subject data. The most probable joint configuration for each trajectory can be estimated by the RHBM, which will allow future work to optimize task trajectories for potential training and therapy.
Joint limit functions were omitted based on results from simulated tasks due to the decreased correlation between recorded and simulated trials.	The recorded optimal poses from the control provide a stricter constraint than joint limits, ensuring that all joint remain within joint limits.
No functions for collision avoidance were developed or tested.	The new control method has inherent self-avoidance via the pose estimation algorithm.
The weighting factors for each task were determined by trial and error.	Weighting and other control parameters were optimized in Matlab, to maintain optimum values based on pose and task requirements.

1.8 Summary of the RHBM

The RHBM is a 25 DoF bilateral upper body model with subject specific kinematic and control parameters. The segment, or link, parameters of the RHBM are determined from

the RoM data by the functional joint center methods, detailed in Chapter 3:. The segment parameters can also be calculated from a linear regression of common anthropometric measurements of the upper body, which are given in Section 2.3. Each link corresponds to a rotational DoF; all joints in the model have three DoFs, except the hand which has only 2 due to the constraints at the wrist. The descriptions of each joint of the RHBM are given in Table 4.

Table 4: Segment and joint definitions of RHBM

Segment	Joint	Right Arm Convention	Left Arm Convention	
Torso	1	Torso Extension		
Torso	2	Lateral Torso Flexion		
Torso	3	Torso Rotation		
Shoulder	R4	Protraction	L4	Retraction
Shoulder	R5	Depression	L5	Depression
Shoulder	R6	External Rotation	L6	Internal Rotation
Upper Arm	R7	Flexion (transverse)	L7	Extension (transverse)
Upper Arm	R8	Elevation (coronal)	L8	Elevation (coronal)
Upper Arm	R9	Axial Rotation (external)	L9	Axial Rotation (internal)
Forearm	R10	Flexion	L10	Extension
Forearm	R11	Carrying Angle	L11	Carrying Angle
Forearm	R12	Pronation	L12	Supination
Hand	R13	Flexion	L13	Extension
Hand	R14	Abduction	L14	Abduction

The joints for the torso (1-3) are common across the left and right arm. The description of each joint is in terms of the convention used by the robotic model, and therefore equivalent joints on the right and left arm do not always move in the same direction. In the clinical convention, Section 3.4, the direction joint rotation is the same on both sides and is equal to the positive directions of the right arm. A diagram showing the axes of rotation and the lengths of each segment is given in Figure 7.

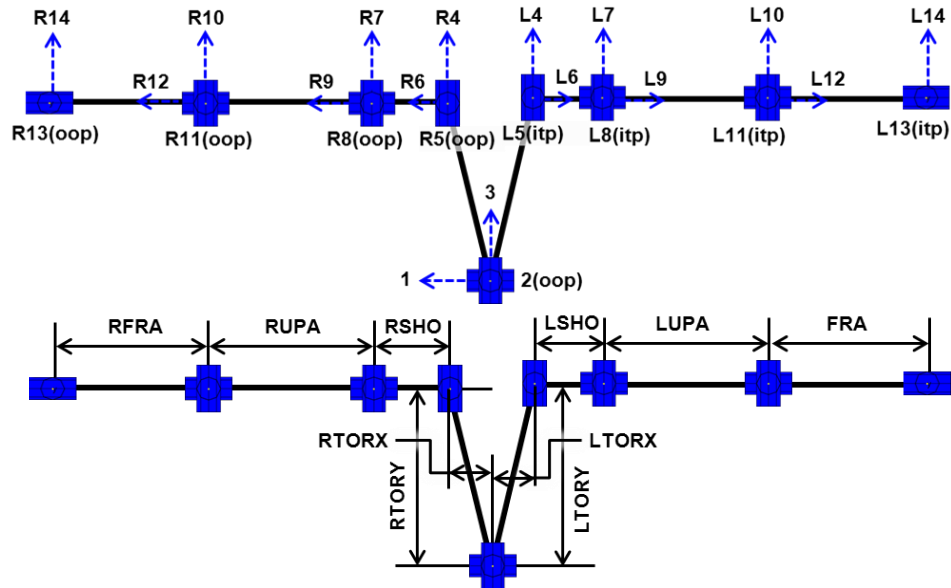


Figure 7: Diagram of the RHBM kinematics (axes top, lengths bottom)

The selected control of the RHBM inverse kinematics was based on the weighted least norm solution with a null space correction based on the probability density function. The flow of data for to the development of the RHBM is shown in Figure 8.

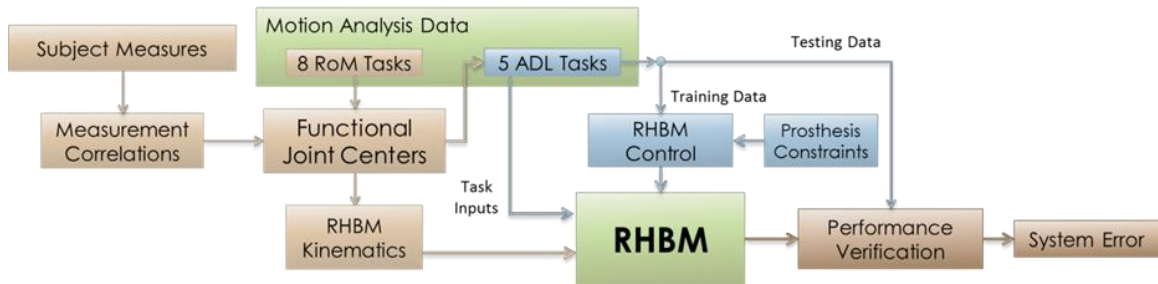


Figure 8: Diagram of the data flow during development of the RHBM

1.9 Dissertation Overview

This dissertation is split into seven chapters based on the approximate chronology of work performed in the study. This first chapter covered the objectives, motivation, background, previous work, and a brief preview of the final RHBM. The second chapter describes the data collection methods, which is then used in the following chapters.

Chapter Three covers the methods for development of the segment parameters and joint angles, or kinematics, of the RHBM. Chapter Four covers the kinematic results from the

motion analysis data, as well as the results from the joint center calculations and segment definitions. Chapter Five covers the development of methods for the various control algorithms tested. Chapter Six describes the results of the control algorithm testing, and compares the various methods. Finally, Chapter Seven discusses the final RHBM, other significant findings, and future work. Each chapter has been written to stand alone, but occasionally reference to preceding or proceeding chapters or sections are necessary to provide relevant information without being repetitive. In these cases links to the appropriate sections are provided.

Chapter 2: Subject Motion Capture and Measurement

Human motion is a well-studied field of research. Since the goal was to accurately reproduce and predict human motion it makes sense to start by observing and quantifying human motion. An eight camera Vicon (OMG plc., Oxford, UK) motion analysis system was used to collect data from 14 subjects performing RoM and ADL trials. Of the subjects, 10 were non-amputee controls, one subject used a transradial myoelectric prosthesis, one subject was a bilateral transhumeral amputee with two body-powered prostheses, one subject was a unilateral transhumeral amputee with a body-powered prosthesis, and one subject was a unilateral transhumeral amputee with myoelectric prosthesis. One of the control subjects had a congenital limb deficiency, missing digits 4 (ring finger) and 5 (digiti minimi) of their right hand, but showed no functional limitations. A marker set was developed for use with the proceeding methods; and consisted of up to 31 passive reflective markers, depending on the level of amputation. These markers were used to track the segment locations during the various tasks, or to act as redundant tracking points in the case of marker dropout.

The subjects were asked to perform 13 tasks during the motion analysis data collection. These tasks were divided into two categories: 8 RoM tasks and 5 ADLs. The data collected during RoM tasks were used to calculate the segment functional joint centers of the upper body, and analyze differences in range of motion between groups. The functional joint centers and marker positions were then used to define the segment coordinate frames. The segment coordinate frames were arranged into a kinematic chain,

and used to extract the parameters and joint angles of the RHBM. Data collected from ADLs were used to train the various control algorithms and to analyze the compensatory movements of the prostheses users and the braced control subjects.

2.1 Subject Demographics

The demographic information for the 14 individuals that participated in this study is given in Table 5. Anthropometric measurements were taken of each subject according to the measurement form in Appendix A.1. These measurements were tested for correlations to the upper body segment geometry extracted from the RoM data. This will allow clinicians to accurately reproduce the subject kinematics based on measurements that are taken as part of a routine patient evaluation. Information on each subject's prosthesis was recorded and used in creating the component dependent parameters for motion prediction with different prosthetic devices.

Table 5: Subject demographic data

Subject #	Age (yr)	Sex	Height (m)	Body Mass (kg)	Dom. Hand	Amp. Side	RLL (cm)	Socket Type	Pros. Mass (kg)	Pros. Type
C01	21	M	173	62.5	R	-	-	-	-	-
C02	25	M	180	79.8	R	-	-	-	-	-
C03	20	M	181	83.5	L	-	-	-	-	-
C04	20	M	180	70.5	R	-	-	-	-	-
C05	24	M	186	100.5	R	-	-	-	-	-
C06	35	M	184	102.5	L	-	-	-	-	-
C07	38	F	160	62.0	R	-	-	-	-	-
C08	41	M	177	73.2	R	-	-	-	-	-
C09	58	M	174	90.5	R	-	-	-	-	-
C10	54	F	166	65	R	-	-	-	-	-
H01	61	M	175	90.3	-	Bi	17	TR	-	Hook
H02	41	M	175	73.5	L	R	26	SS	1.9	Hook
H03	61	M	174	73	R	L	11.5	Utah	2.2	Utah
R01	48	M	174	88	R	R	23.2	i-limb	1.3	Pulse

C = Control Subject, H = Transhumeral Subject, R = Transradial Subject

2.2 Braced Subjects

Control subjects were asked to complete all tasks with and without a brace on their dominant arm. The brace restricts pronation / supination of the forearm, as well as flexion / extension, and abduction / adduction of the wrist. The inclusion of braced testing for control subjects allows for a potential reduction of subject range of motion that is similar to that seen in amputees, although the magnitude of compensatory motions of braced subjects is generally less than that of amputee subjects [29]. Additionally, studies have also shown compensatory motions in object manipulation, [91] citing the potential for shoulder injury in assembly workers wearing splints due to increased upper arm elevation and axial rotation. This helps to compensate for the limited number of amputee subjects in order to test the control algorithms, by increasing the amount of data available for training and testing.

2.3 Anatomical Measurements

The list of manually recorded subject measurements for control subjects is given in Table 6, and are based on measurements by Gordon et al. [92]. All measurements were recorded using a standard cloth measuring tape.

Table 6: Anthropometric measurement names

ID	Description
CC	Chest circumference
UCP	Upper arm circumference at axilla
UCD	Upper arm circumference superior to elbow
FC	Forearm circumference distal to the elbow
SC	Wrist circumference at styloid process
A2E	Acromion to lateral humeral epicondyle
X2E	Axilla to medial humeral epicondyle
E2S	Lateral humeral epicondyle to radial styloid process (wrist pronated)
E2T	Lateral humeral epicondyle to thumb tip (wrist pronated)
S2T	Radial styloid process to thumb tip

Standard measurements for the residual limb of the amputee subjects were also recorded. Residual limb length measurements were taken from the reference landmark to the end of the residual limb with the tissue compressed. The list of measurements is given in Table 7.

Table 7: Residual limb measurements

ID	Description
PRLC	Residual limb circumference at the axilla
DRLC	Distal residual limb circumference
A2RL	Acromion to residual limb end
X2RL	Axilla to residual limb end
E2RL	Lateral epicondyle to residual limb end

2.4 Motion Capture

Motion analysis is the process of quantitatively evaluating specific aspects of the movement of bodies. This is done by taking images of tracking points or markers from multiple views and triangulating the 3D position of each marker from the intersection of the projection of the 2D images. The Vicon system used in this study had 8 infrared cameras that tracked the positions of passive reflective markers placed on the upper body of the subjects. The markers used in this study are given in Table 8. The total number of markers and their descriptions is referred to as a marker set. The marker set used for each subject was dependent on their level of amputation. Non-amputees did not use the residual limb or socket markers (RSLA, RSLP, SCKTA, SCKTP). If socket trim lines were very near the shoulder or elbow markers the residual limb markers (RSLA & RSLP) are neglected. If the socket covered the elbow of a transradial prosthesis user the socket markers (SCKTA & SCKTP) replace the elbow markers (ELB & ELBM), in the position of the elbow markers. These changes allow the use of the same starting marker set for a combination of amputee levels, and for both left and right arm amputees. The tracking

markers included in the marker set provide additional points for the automatic labeling algorithm in Vicon Workstation, increasing the ease of the labeling process. The tracking markers can also be used to reconstruct the position of other markers in the case of marker dropout. This was done using the marker cluster algorithm [55], and can regenerate the position of a missing marker provided three markers on the same body segment are still visible.

Table 8: Marker descriptions

Name	Placement
T1	Spinous process; 1 st thoracic vertebrae
*T10	Spinous process; 10 th thoracic vertebrae
CLAV	Jugular notch
*STRN	Xiphoid process
*LBAK	Middle of left Scapula (asymmetrical)
R/LASI	Right / Left anterior superior iliac spine
R/LPSI	Right / Left posterior superior iliac spine
*R/LIC	Right / Left iliac crest
R/LSHOA	Anterior portion of right / left acromion
R/LSHOP	Posterior portion of right / left acromion
*R/LUPA	Right / Left lateral upper arm
R/LELB	Right / Left lateral epicondyle
R/LELBM	Right / Left medial epicondyle
*R/LFRA	Right / Left lateral forearm
R/LWRA	Right / Left wrist radial styloid
R/LWRB	Right / Left wrist ulnar styloid
R/LFIN	Dorsum of right hand just proximal to 3 rd metacarpal head
¹RSLA	Anterior or lateral residual limb above trim line
¹RSLP	Posterior or medial residual limb above trim line
²SCKTA	Anterior or lateral portion of the socket in line with SHO or ELB markers
²SCKTP	Posterior or medial portion of the socket in line with SHO or ELB markers

**Markers used for tracking and redundancy only, these markers are less sensitive to placement as they are not used in segment definition.*

¹For subject where the socket trim line was very near the shoulder for transhumeral subjects or the elbow for the transradial the residual limb markers (RSLA & RSLP) were neglected.

²The socket covered the elbow of the transradial subject therefore the socket markers (SCKTA & SCKTP) replaced the elbow markers (ELB & ELBM), in the position of the elbow markers.

2.5 Range of Motion Tasks

This section describes each RoM task as described to the subjects, Table 9. Subjects were asked to start with enough clearance between their arms and sides to prevent obstruction of the cameras' view of the markers. All movements were performed without assistance, and can be considered active, patient-initiated, RoMs. Each trial was completed three times to collect an average RoM for each subject.

Table 9: Subject Instructions for RoM tasks

Elbow Flexion / Extension	Start with your elbows extended, palms facing body, thumbs forward, flex your elbows until maximum flexion is reached. Hold that position briefly, and then extend your elbows back to terminal extension.
Forearm Pronation / Supination	Start with your elbows flexed to 90° (subject approximated), arms near the body, palms facing inward, rotate your forearms inwards toward body to as far as you can, and flex wrist downward. After a brief pause rotate the forearm outward (supinate) while continuing to point hands down (extending the wrist). Pause briefly then return to the starting position.
Shoulder Flexion / Extension	Starting with your arms extended towards the floor, palms facing your body, raise your arms, reaching forward, then up, then backward as far as you can (maximum shoulder flexion). After a brief pause return arms by stretching, up, forward, down, and then backward (maximum extension). Pause briefly before returning to starting position.
Shoulder Abduction / Adduction	Starting with your arms extended toward the floor, palms facing your body, thumbs forward, abduct arms with elbows straight to maximum, then pause briefly. Adduct arms back down crossing arms in front of the chest, and then return to the starting position.
Shoulder Rotation	Starting with elbows flexed to 90° (subject approximated) and arms abducted until parallel with floor, palms facing down. While keeping your upper arms parallel to floor rotate the forearm arms downward as far as you can. Pause briefly then rotate your arms upward to maximum position. Pause again before returning to the starting position
Torso Flexion / Extension	Starting from a vertical standing position, flex the torso as far forward as possible without needing to take a step, focusing on bending your spine. Pause briefly then extend torso backwards as far as you can. Pause again then return to the starting position.
Torso Lateral Flexion	Starting from a vertical standing position, lean as far to the right as possible bending your torso. Pause briefly then lean to the left as far as possible. Pause again then return to the starting position.
Torso Rotation	Starting from a vertical standing position, keeping your torso upright, rotate to the right as far as possible. Pause briefly then rotate to the left as far as possible. Pause again then return to starting position.

For the RoM tasks the subjects were led by a researcher to ensure that they were moving their joints through the proper range of movement associated with each task. The speed the subjects perform each task, and the duration of all pauses was selected by the subjects. Additionally subjects were asked at the start of the collection to not over-exert themselves, to reduce the risk of injury.

2.6 Activities of Daily Living

The ADLs as they were presented to the subjects are given in Table 10. Similar to the RoM tasks, subjects were asked to start with enough clearance between their arms and sides to prevent obstruction of the cameras' view of the markers. All ADLs were performed without assistance. All subjects were able to complete the specified tasks. Each activity was completed three times for intra-subject comparison. Unilateral tasks were completed with the dominant, braced, or prosthetic arm. No instructions were given for the pose or movement for the uninvolved arm during unilateral tasks.

Table 10: Description of ADLs

Bushing Hair	Stand with your arms at your side facing the table. Pick-up a brush from the table, 'Brush' your hair (subject selected duration), return brush to the table, and return to the starting position.
Drinking from a Cup	Stand with your arm at your side with the elbow flexed to approximately 90° holding the cup. Raise the cup to your mouth to 'drink', lower the cup back to the original position.
Eating with Knife and Fork	In a seated position, start with your arms on either side of the place setting. Grasp the knife and fork, mime cutting a piece of steak, mime eating, then set down knife and fork, and return to starting position.
Lifting a Laundry Basket	Starting from a comfortable standing position, pick the basket (10 lb) up from the ground, raise and place the basket on the table (height: 82 cm), release basket and return to a comfortable standing position. Pick the basket up from the table, return the basket to the original position on the ground, and then return to starting position. (Lifting the basket and returning it to the floor is considered one trial).
Opening a Door	Stand with your arm at your side facing the door. Open the door, and then return to the starting position. Closing the door is not included in the recorded data.

Chapter 3: Determining Functional Joint Centers and Upper Body Segments

To generate a geometrically accurate model of the upper body, without increasing the complexity of the model, functional joint center calculations were used to define the model segment. The use of functional joint centers for upper body modeling has not been published; however several algorithms have been published for general use, and for use in the lower limb. Specifically a least squares sphere fit method [64], an optimization algorithm for finding the joint center of the hip by Piazza et al. [56], and a gradient based optimization for automatic skeleton generation by Schönauer [58], have been developed. To test the different algorithms, a field of 3 random points was generated in Matlab and rotated about a known constant center. Each algorithm was then used to find the joint center given different levels of noise. The error between the calculated joint centers and the known center of rotation was then evaluated. Each method was also tested in generating the location of the glenohumeral joint center given data with varying RoM [93]. The least squares method was very accurate without noise but quickly became unstable when noise was introduced. The method developed by Piazza had a consistently higher average error than the gradient method; however, it was less susceptible to noise than the least squares method. The gradient method developed by Schönauer was found to be the most resilient method, with its greatest limitation being that high errors occurred in instances where the initial guess was poor, which resulted in error even in the case where no noise was introduced [94].

Since a reasonable initial guess can be found for anatomical joints by using the relative position of markers, the gradient method was chosen for use in this study. The functional joint center was calculated by optimizing the cost function which penalizes the variation in distance between each point and the distal segment and potential joint center. The cost function is given in Eq. 1 and the function for average distance between the tested point and a point on the distal segment is given in Eq. 2. The cost function increases as the sum of the variance of the distance between the position (x, y, z) and all points in an m by 3 by n array increases, where m is the number of samples, and n is the number of markers. Px_k^i is the x position of point i at time (or sample) k . The point Px_k^i was the element $P(k, I, i)$. The minimum of the cost function is the position where the distance between (x, y, z) and all points of P is constant. This assumes that the body was undergoing primarily rotation, and that translation was relatively small within the reference frame.

$$\text{Eq. 1 } C(x, y, z) = \sum_{i=1}^n \sum_{k=1}^m \left[\sqrt{(Px_k^i - x)^2 + (Py_k^i - y)^2 + (Pz_k^i - z)^2} - \text{Ravg}^i \right]^2$$

$$\text{Eq. 2 } \text{Ravg}^i(x, y, z) = \frac{\sum_{k=1}^m \sqrt{(Px_k^i - x)^2 + (Py_k^i - y)^2 + (Pz_k^i - z)^2}}{m}$$

The initial guess for the joint center was the average of marker positions placed on the body near the joint center. This method has proven to be effective where a sufficient RoM was present. The RoM tasks, Section 2.5, in this study provide the necessary data to ensure accurate joint centers using this method.

3.1 Importing Data from Motion Capture

All of the kinematic and joint center calculations were performed as a batch process in the *CreateUBM.m*, Appendix B.1, Matlab file on a subject basis. Data collected in Vicon

Workstation were saved into the *.c3d format which contains the marker position data. Data were imported from the motion analysis files into Matlab matrices using the c3d server application developed by Walker and Rainbow [95]. A data structure was created for the RoM data, the subject was defined as a field in the RoM field, each trial was a field within each subject, and marker data were stored as variables inside the task field. The data were loaded automatically by reading the subject data directly and loading the *.c3d files into fields based on the folder names, trial names, and the desired subject number specified by the user. Figure 9 shows the configuration of the file structures required for the programs to operate correctly.

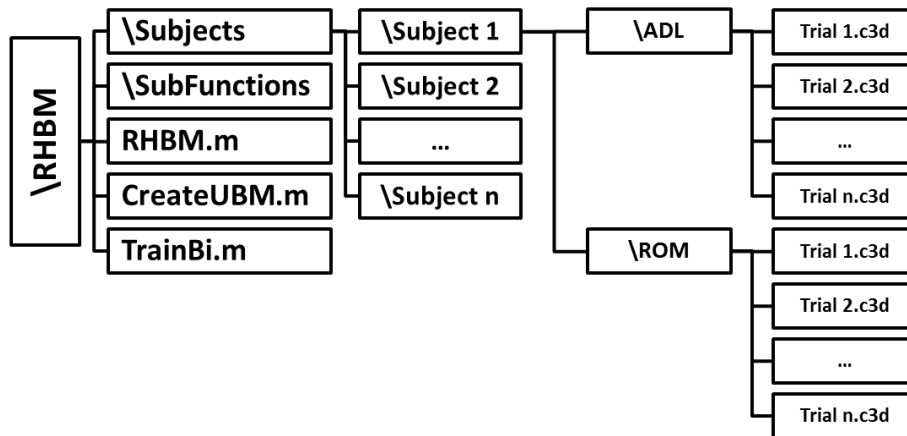


Figure 9: RHBM file directory setup

Any spaces in trial names are removed with the *removewhite.m*, Appendix B.2, function, as spaces are not allowed in Matlab field names. After all of the trials have been loaded, the marker position data were filtered using a low pass filter. The *WMAfilter.m*, Appendix B.3, function was used to filter the data. The function creates a linear weighted moving average with the width specified in the first input. An 11 point width filter was used to filter the raw position data to remove noise.

3.2 Segment Definitions and Joint Centers

Each segment was defined by an origin and two defining lines using *createSegment.m*, Appendix B.4. Each segment in the RHBM was centered at the origin. The unit vector parallel to the first defining line becomes the first axis of the segment. The unit vector parallel to the cross product of the first and second line becomes the second axis. Finally the cross product of the first two axes becomes the final axis. The order of the axis names was set in the model using a string, for instance if the first, second, and third axes were X, Y, and Z, then the string would have been defined as 'xyz.' In order to maintain the right hand rule, the direction of the third axis depends on the order specified, for instance in the case of 'yxz' the negative of the cross product of the Y and X axes becomes the Z axis. The 4 by 4 homogeneous transformations for each point in time, as well as the direction of each axis, were saved as fields in the segment structure. The segment structure was saved into a field for each task. Point data were described in the segment frame by adding the point to the segment structure by calling the *addPoint2.m*, Appendix B.5, and *addDistalPoint.m*, Appendix B.6, where the latter was used to define the points used for the functional joint center calculation, to find the next segment origin.

3.2.1 Pelvis

The pelvis segment was the primary reference frame for all upper body markers and was used to describe the relative location of objective positions in end effector space. Because the RASI and LASI markers were prone to being obscured when subjects bent over, a reconstruction algorithm was created. If no additional tracking markers were used then the *reconstruct.m* Appendix B.7 was used, which can find the position of missing markers as long as only one was missing at a time. If the tracking markers RIC and LIC

were used then *clusterReconstruct.m*, Appendix B.8 was used and can regenerate the pelvis markers if up to three markers were missing from the pelvis. If more than three of the pelvis markers are missing it was impossible to generate the pelvis frame. The ISB recommendations for the pelvis are included in the lower body definitions [9]. The Z-axis was defined as parallel to the line connecting the right and left ASI markers, pointing right. The X-axis was defined as the line orthogonal to the Z-axis lying in the plane defined by RASI, LASI, and the midpoint of the LPSI and RPSI (MPSI). The Y-axis was defined perpendicular to the X and Z axes, maintaining the right hand rule. The segment was defined with the MPSI as the origin, because the segment was used for movement relative to the torso, and not the thigh as in the ISB lower body recommendations. The first defining line was defined from LASI to RASI, and the second is defined from MPSI to RASI, with the convention 'zyx.' The orientation of the frame relative to the pelvis markers is shown in Figure 10.

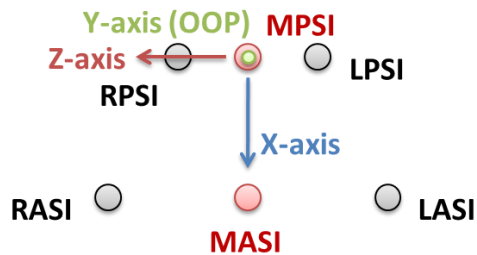


Figure 10: Diagram of the pelvis definitions

The T1 and CLAV marker were then defined in the pelvis segment and added to the pelvis structure. All of the positions of the T1 and CLAV for all of the RoM tasks for each subject was concatenated into a single array, *pelvisCompiled*, and sent to the *MLOptim.m*, Appendix B.9, function to calculate the functional joint center of the torso segment in the pelvis frame.

3.2.2 Torso

The torso segment is defined in the ISB recommendations with the Y-axis parallel to the line from the midpoint between the xiphoid process and 8th thoracic vertebra (T8) to the midpoint of the jugular notch (CLAV), and 7th cervical vertebra (C7). They define the Z-axis as the line perpendicular to the plane formed by the CLAV, C7, and the midpoint of the xiphoid process and T8, positive to the right. The X-axis is defined as the line perpendicular to the Z and Y axes. In our model we use the functional joint center of the torso instead of the midpoint of the xiphoid process and 8th thoracic vertebra, allowing us to eliminate markers. The T1 marker is used instead of the C7 to help eliminate soft tissue movement of the neck. The origin is set to the functional joint center. The first defining line is defined from the torso joint center to the average of the CLAV and T1 markers. The second defining line is defined from CLAV to T1, with the convention 'yzy.' The orientation of the frame relative to the torso markers is shown in Figure 11.

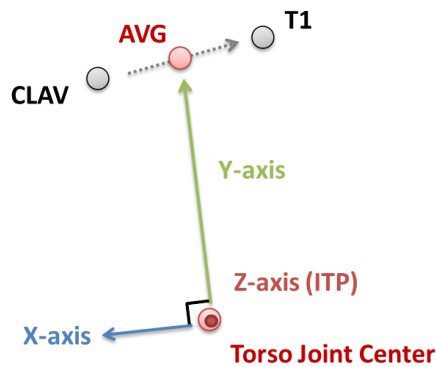


Figure 11: Diagram of torso segment definitions

The rotational order between the torso and the pelvis was 'zxy' which represents torso flexion, lateral flexion, and rotation. Since the torso segment and all distal segments after it follow a similar convention, the processing was performed in the *autoSegments.m* function, Appendix B.10. This function creates the segment as defined above, calculates

the joint center of the next segment, and then re-defines the segment by replacing the average of the two segment markers (CLAV and T1 for the torso) with the joint center of the distal segments as the second point on the first defining line. This ensures that the distance between centers is described in the Z-axis of the proximal segment.

3.2.3 Shoulder

The shoulder is the segment that connects the torso and the upper arm and approximates the movement of the clavicle and the scapula. The ISB recommendations separate the clavicle and scapular movement and have individual segment definitions for each system. However, tracking scapular movement with skin markers is difficult due to the large displacement of bone relative to the skin over the scapula. Due to this error, and the relatively small movement between the glenohumeral joint and the acromioclavicular joint the motion of the scapula and the clavicle are approximated as a single segment, which is referred to as the shoulder segment.

The origin of the shoulder segment was defined as the functional joint center of the shoulder complex. The first defining line was defined from the functional joint center of the shoulder complex to the functional joint center of the upper arm. However since we need a segment definition to find the functional joint center of the upper arm, the average position of the anterior and posterior shoulder markers are used temporarily. This process was repeated with all segments distal to the torso. The second defining line is the line from the posterior to anterior shoulder marker on the right, and anterior to posterior on the left. The segment axis order is 'zyx,' making the segment orientation similar to the ISB definitions. The 'yxz' rotational order is used between the shoulder and the torso.

The Y axis represents the protraction of the shoulder segment on the right, and retraction

on the left. Rotation around the X axis represents rotation depression of the shoulder on the right and left. Rotation about Z represents the roll or sagittal rotation of the shoulder segment, and is internally positive on the right and negative on the left. The orientation of the frame relative to the shoulder markers is shown in Figure 12.

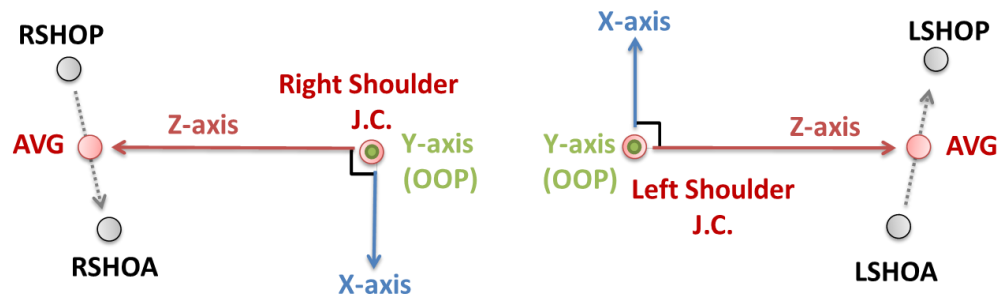


Figure 12: Diagram left and right shoulder segment definitions

The shoulder is also the first segment where there exists a right and left pair. Since there is no assumed symmetry in the model, each side is calculated separately. Because we would like the right and left sides to be as consistent as possible, the same segment definitions were used for the creation of the segments on the right and left side. This necessitates modification of the raw segment rotation into clinically relevant joint angles, Section 3.4, since the direction of the segment axes varies and the segment definitions must obey the right hand rule. The segment orientations for the left and right side are shown in Figure 12. Positive rotation of the X-axis on the right side is depression of the shoulder, and on the left it is elevation. Positive rotation of the Y-axis is protraction of the shoulder on the right and left side. Rotation of the Z-axis is best described as axial rotation of the clavicle, and is also in the same direction on both sides.

3.2.4 Upper Arm

The upper arm and forearm segment definitions are very similar to the shoulder definition. The first defining line was defined from the upper arm joint center to forearm

joint center, with the average of the medial and lateral elbow markers serving as the temporary joint center. The second defining line was defined from the lateral to medial elbow marker on both right and left sides. Both sides use the ‘zyx’ axis definitions. The axes represent flexion, abduction, and rotation of the upper arm about the glenohumeral joint center. The orientation of the frames relative to the elbow markers is shown in Figure 13.

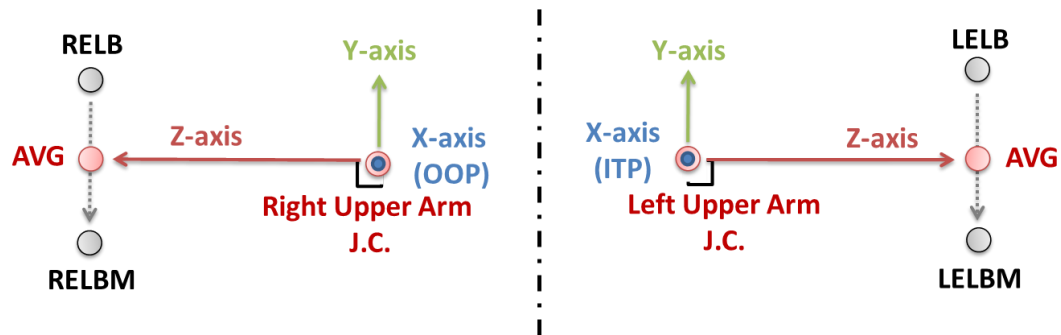


Figure 13: Diagram of left and right the upper arm segments

The ‘yxz’ free axis rotational order between the shoulder and upper arm segments is used to find the joint angles. The Y-axis represents flexion (or plane of elevation) in the transverse plane of the shoulder complex. The X-axis represents abduction (elevation) in the frontal plane of the shoulder complex. The Z-axis represents axial rotation of the upper arm about the glenohumeral joint center.

3.2.5 Forearm

The motions of the forearm segment include flexion, carrying angle, and pronation about the center of rotation, which is located at the elbow. The first defining line was defined from the forearm joint center to the average of the wrist markers. The second defining line was defined from the ulnar to radial marker on the right and from the radial to ulnar wrist marker on the left. The ‘yxz’ order was used to define segments on both the right

and left sides. The orientation of the frames relative to the wrist markers is shown in Figure 14.

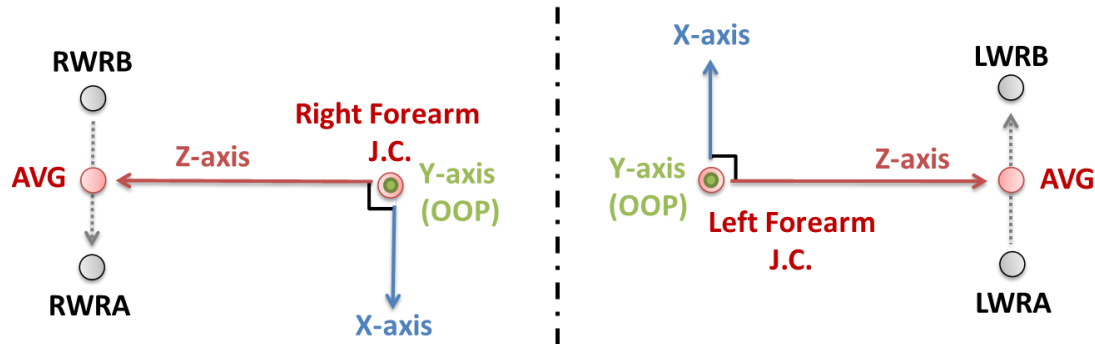


Figure 14: Diagram of the forearm segments

The rotational order 'yxz' was used to find the free axis rotational angles between the forearm and upper arm. Rotation about the Y-axis represents flexion of the elbow in the sagittal plane of the upper arm, rotation about the X-axis represents the carrying angle of the arm, and rotation about the Z-axis represents pronation and supination of the forearm. The carrying angle [96] is extracted from the rotation about the X-axis. The carrying angle is nearly constant for each subject but varies between subjects and has potential as a design variable for optimizing performance of prosthetics.

3.2.6 Hand

The hand was defined using the wrist markers, the marker on the third metacarpal head, and the joint center of the hand. The first defining line goes from the joint center to the metacarpal head, and the second line was defined from the ulnar to radial marker on the right and from the radial to ulnar wrist marker on the left. The 'zyx' axis definition order was used on both sides. The rotational order for the hand relative to the forearm was 'xyz'. The X-axis rotation of the hand is the flexion / extension of the wrist and the Y-axis is abduction / adduction. Because the X-axis of the forearm was used in the

definition of the hand segment, it only has two DoFs and the Z-axis rotation of the hand was always zero. The orientation of the frames relative to the wrist and hand markers is shown in Figure 15.

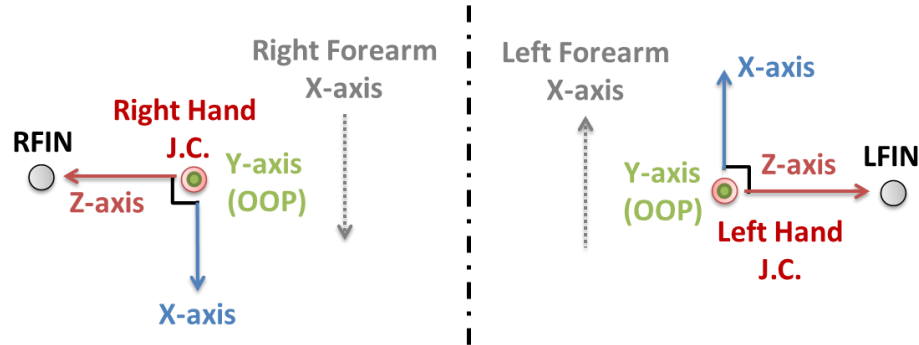


Figure 15: Diagram of the hand segments

3.3 Determining Denavit and Hartenberg Parameters and RHBM Joint Angles

After all of the segments have been defined, and the joint centers have been calculated, they are redefined using the distal joint center in place of the average of the distal markers for all segments except the torso and the hands. This redefinition makes the distance between segments lie entirely on the Z-axis, which simplifies the calculation of the Denavit and Hartenberg parameters as described in the convention established by Craig [97]. This redefinition does not change the location of the joint centers in space, but the orientation of each segment. The distance between the joint centers also remains the same, and equal to the square root of the sum of the squares of the position elements in the temporary frames as given in the tables of the preceding section. Joint angles are calculated from the segment homogeneous transforms using the *autoFindTheta.m* Appendix B.11, and the *findTheta.m* Appendix B.12, functions. *findTheta.m* calculates the Euler angles given a 3 by 3 rotation matrix and a given convention, and *autoFindTheta.m* calculates the rotation matrix for all points of all trials for all subjects and then calls *findTheta.m* to find the joint angles. The rotational order ‘zxy’ was used for

the torso, 'xyz' was used for the hands, and 'yxz' was used for all other segments. The joint angles for the RHBM required the addition of offsets to match the existing conventions, and maintain orthogonal joint axes. The angular offsets, as well as the other Denavit and Hartenberg parameters, are defined in *createRobot.m*, Appendix B.13. Descriptions of the parameters used in the RHBM are given in Figure 16. The full lists of parameters as they are used to create the links of the RHBM are given in Table 12. A graphical representation of the upper body model using the parameters from subject C03 is shown in Figure 16.

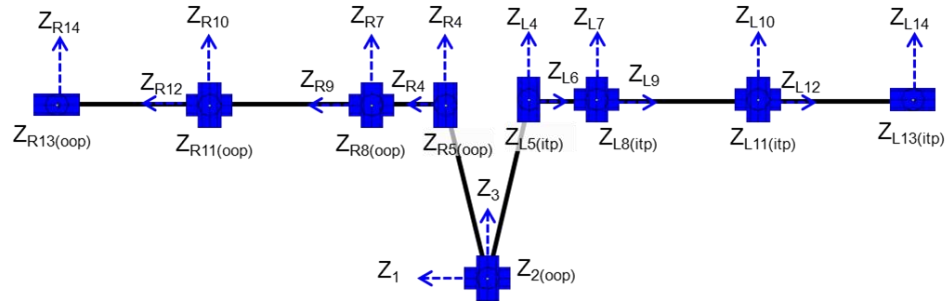


Figure 16: Matlab plot of robot [90] object for subject C03

Table 11: Description of Denavit and Hartenberg parameters

Name	Description
A	Link Length: the distance along the line normal to both axes
α	Link Twist: the angle between the current link axis and the next link axis
D	Link Offset: the distance between the center of the current link and the next along the link axis.
Θ	Joint Offset: the initial rotation of the link about its axis
R1-14	Links of the right arm model
L1-14	Links of the left arm model
RSJC_{x,y,z}	X, Y, Z position of the right shoulder joint center.
RUAJC_z	Z position of the right upper arm joint center (shoulder segment length).
RFJC_z	Z position of the right forearm joint center (upper arm segment length).
RHJC_z	Z position of the right hand joint center (forearm segment length).
LSJC_{x,y,z}	X, Y, Z position of the left shoulder joint center.
LUAJC_z	Z position of the left upper arm joint center (shoulder segment length).
LFJC_z	Z position of the left forearm joint center (upper arm segment length).
LHJC_z	Z position of the left hand joint center (forearm segment length).

Table 12: Denavit and Hartenburg parameters

Link	α	A	Θ	D	Segment	Axis	Positive Convention
R1	0	0	0	0	Torso	Z	Extension
R2	$\pi/2$	0	$-\pi/2$	0	Torso	X	Right Lateral Flexion
R3	$-\pi/2$	0	$-\pi/2 - atan2(RSJC_z, RSJC_x)$	0	Torso	Y	Left Rotation
R4	0	$\sqrt{RSJC_x^2 + RSJC_z^2}$	$-atan2(RSJC_x, RSJC_z)$	$RSJC_y$	Right Shoulder	Y	Protraction
R5	$-\pi/2$	0	$-\pi/2$	0	Right Shoulder	X	Depression
R6	$-\pi/2$		$-\pi/2$	$RUAJC_z$	Right Shoulder	Z	External Rotation
R7	$-\pi/2$	0	$-\pi/2$	0	Right Upper Arm	Y	Flexion
R8	$-\pi/2$	0	$-\pi/2$	0	Right Upper Arm	X	Adduction
R9	$-\pi/2$	0	$-\pi/2$	$RFJC_z$	Right Upper Arm	Z	External Rotation
R10	$-\pi/2$	0	$-\pi/2$	0	Right Forearm	Y	Flexion
R11	$-\pi/2$	0	$-\pi/2$	0	Right Forearm	X	Adduction
R12	$-\pi/2$	0	0	$RHJC_z$	Right Forearm	Z	Supination
R13	$\pi/2$	0	$\pi/2$	0	Right Hand	Y	Flexion
R14	$\pi/2$	0	0	0	Right Hand	X	Adduction
L1	0	0	0	0	Torso	Z	Extension
L2	$\pi/2$	0	$-\pi/2$	0	Torso	X	Right Lateral Flexion
L3	$-\pi/2$	0	$-\pi/2 - atan2(LSJC_z, LSJC_x)$	0	Torso	Y	Left Rotation
L4	0	$\sqrt{LSJC_x^2 + RSJC_z^2}$	$\pi - atan2(LSJC_x, LSJC_z)$	$LSJC_y$	Left Shoulder	Y	Retraction
L5	$\pi/2$	0	$\pi/2$	0	Left Shoulder	X	Depression
L6	$-\pi/2$	0	$-\pi/2$	$LUAJC_z$	Left Shoulder	Z	Internal Rotation
L7	$-\pi/2$	0	$-\pi/2$	0	Left Upper Arm	Y	Extension
L8	$-\pi/2$	0	$-\pi/2$	0	Left Upper Arm	X	Adduction
L9	$-\pi/2$	0	$-\pi/2$	$LFJC_z$	Left Upper Arm	Z	Internal Rotation
L10	$-\pi/2$	0	$-\pi/2$	0	Left Forearm	Y	Extension
L11	$-\pi/2$	0	$-\pi/2$	0	Left Forearm	X	Adduction
L12	$-\pi/2$	0	0	$LHJC_z$	Left Forearm	Z	Pronation
L13	$\pi/2$	0	$\pi/2$	0	Left Hand	Y	Extension
L14	$\pi/2$	0	0	0	Left Hand	X	Adduction

3.4 Clinical Joint Angles

The direct rotations of segments are used in the kinematics calculations. However, due to the complexity and conventional requirements of the model, these joint angles can be difficult to interpret. The Euler angle rotations of the shoulder can also result in gimbal lock, where the axes of rotation become aligned, resulting in reduced manipulability of the joint and high joint angle velocities become necessary for small movements. To increase the ease of clinical analysis of joint angles, the raw joint angles are re-computed in a more intelligible context. This section describes the conventions used for the clinical joint angles, and how they are calculated. The free axis rotational, orders ‘zxy’ for the torso, ‘xyz’ for the hands, and ‘yxz’ for the other segments were used in the robot angle calculations. The robotic convention for joint angles also includes the angular offsets required to manipulate the robotic model, which are not included in the clinical angles.

3.4.1 Rotational Conventions

The rotation between two segments can be described by the projection of the distal frame axes Rd_x , Rd_y , and Rd_z onto the proximal frame. Where Rd_x is a 3 by 1 vector, $[R11, R21, R31]^T$, of the projection of the distal X axis onto the X, Y, and Z, axes of the proximal frame, and Rd_y and Rd_z are the projections for the distal Y and Z axes respectively. This creates the 3 by 3 rotational matrix, R , that describes the rotation between the segments, as shown in Eq. 3.

$$\text{Eq. 3} \quad R = [Rd_x \quad Rd_y \quad Rd_z] = \begin{bmatrix} R11 & R12 & R13 \\ R21 & R22 & R23 \\ R31 & R32 & R33 \end{bmatrix}$$

The rotation between segments can also be described by rotations about a series of axes.

The rotation between frames, ${}^A_B R$, can be achieved by rotating about the segment axes by

angles γ , β , and α either in the proximal or fixed frame Eq. 4, or about the rotating or free frame Eq. 5. In these cases, R_X , R_Y , and R_Z represent the rotation about the X, Y, and Z axes respectively. The free axis rotations are also referred to as the Euler angles.

$$\text{Eq. 4} \quad {}^A_B R_{fixed}(\gamma, \beta, \alpha) = R_Z(\alpha) * R_Y(\beta) * R_X(\gamma)$$

$$\text{Eq. 5} \quad {}^A_B R_{free}(\gamma, \beta, \alpha) = R_X(\gamma) * R_Y(\beta) * R_Z(\alpha)$$

In the kinematics calculations of the RHBM, the free, or Euler angle rotations are used. A combination of fixed and free rotation can be used to better describe the motion of each joint. The first two rotations can be considered to be about the fixed axis of the proximal segment by switching their order of rotation. For instance the rotations of the torso are calculated as the free axis rotations ‘zxy’ which is torso flexion about the torso Z axis, lateral flexion about the rotated X axis, and rotation about the rotated Y axis. In anatomical terms we can also describe this rotation as rotation about the fixed pelvis X axis, then the fixed pelvis Z axis, and the rotated torso Y axis. This does not change the joint angles but makes the rotation easier to visualize.

$$\text{Eq. 6} \quad {}^A_B R_{mixed}(\beta_{fixed}, \gamma_{fixed}, \alpha_{free}) = R_X(\gamma) * R_Y(\beta) * R_Z(\alpha)$$

This allows the clinical description of the Euler angles, but does not address the problems with gimbal lock of the shoulder. The clinical shoulder joint angles did not follow the ISB recommendations [8], as they have been shown to be prone to gimbal lock. In fact, investigations of Euler rotations for the shoulder found no rotational sequence was clinically interpretable for all movements [98]. Therefore a new convention for clinical shoulder angles was developed. Shoulder flexion, $\theta_{Flexion}$, and abduction, $\theta_{Abduction}$, were described as the arcsine and arccosine of the projection of the axis of the humerus, or upperarm Z-axis, onto the anterior / posterior, and superior / inferior axes of

the shoulder segment, which are the shoulder X and Y-axes respectively. The calculation of shoulder flexion and abduction from the rotation matrix elements is given in Eq. 7 and Eq. 8 respectively.

$$\text{Eq. 7} \quad \theta_{Flexion} = \text{asin}(R13_{Upperarm})$$

$$\text{Eq. 8} \quad \theta_{Abduction} = \text{acos}(R23_{Upperarm})$$

Calculation of the upper arm rotation in a clinical context is more difficult. The definition of internal and external rotation of the upperarm for varying levels of flexion and abduction are not well defined in a clinical context. For this study the orientation of the upperarm segment that maximizes the sum of the projections of the upper arm segment X and Y-axes onto the shoulder segment X and Z-axes, while maintaining the Z-axis orientation as described by the flexion and abduction angles. This minimizes the difference between upperarm segment orientation, and the standard orientation used when clinically evaluating shoulder range of motion. The derivation of the upper arm rotation angle is given in Eq. 9 through Eq. 20. Where $R_{Upperarm}$ is the rotation of the upper arm relative to the shoulder, R_p , is the rotation associated with flexion and abduction to the point of neutral rotation, $R_{Rotation}$, is the Z axis rotation of the upper arm relative to the neutral axis, and $\theta_{rotation}$, is the angle of upper arm rotation. First, R_p , is found in terms of $R_{Upperarm}$ and $R_{Rotation}$, by multipluing both sides of the euation by the transpose of $R_{Rotation}$, as shown in Eq. 9 through Eq. 11.

$$\text{Eq. 9} \quad R_p * R_{Rotation} = R_{Upperarm}$$

$$\text{Eq. 10} \quad R_p * R_{Rotation} * R_{Rotation}^T = R_{Upperarm} * R_{Rotation}^T$$

$$\text{Eq. 11} \quad R_p = R_{Upperarm} * R_{Rotation}^T$$

Then by substituting the elements of the rotational matrices the values relating to the projections of the upper arm segment X and Y-axes onto the shoulder segment X and Z-axes can be found, **Eq. 12** through **Eq. 17**.

$$\text{Eq. 12} \quad \mathbf{R}_{\text{Upperarm}} = \begin{bmatrix} \mathbf{R11} & \mathbf{R12} & \mathbf{R13} \\ \mathbf{R21} & \mathbf{R22} & \mathbf{R23} \\ \mathbf{R31} & \mathbf{R32} & \mathbf{R33} \end{bmatrix}$$

$$\text{Eq. 13} \quad \mathbf{R}_{\text{Rotation}}^T = \begin{bmatrix} \cos(\theta_{\text{rotation}}) & -\sin(\theta_{\text{rotation}}) & 0 \\ \sin(\theta_{\text{rotation}}) & \cos(\theta_{\text{rotation}}) & 0 \\ 0 & 0 & 1 \end{bmatrix}^T$$

$$\text{Eq. 14} \quad \mathbf{R}_{\text{Rotation}}^T = \begin{bmatrix} \mathbf{cR} & \mathbf{sR} & 0 \\ -\mathbf{sR} & \mathbf{cR} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{Eq. 15} \quad \mathbf{Rp} = \begin{bmatrix} \mathbf{R11} & \mathbf{R12} & \mathbf{R13} \\ \mathbf{R21} & \mathbf{R22} & \mathbf{R23} \\ \mathbf{R31} & \mathbf{R32} & \mathbf{R33} \end{bmatrix} * \begin{bmatrix} \mathbf{cR} & \mathbf{sR} & 0 \\ -\mathbf{sR} & \mathbf{cR} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{Eq. 16} \quad \mathbf{Rp} = \begin{bmatrix} \mathbf{R11} * \mathbf{cR} - \mathbf{R12} * \mathbf{sR} & \mathbf{R11} * \mathbf{sR} + \mathbf{R12} * \mathbf{cR} & \mathbf{R13} \\ \mathbf{R21} * \mathbf{cR} - \mathbf{R22} * \mathbf{sR} & \mathbf{R21} * \mathbf{sR} + \mathbf{R22} * \mathbf{cR} & \mathbf{R23} \\ \mathbf{R31} * \mathbf{cR} - \mathbf{R32} * \mathbf{sR} & \mathbf{R31} * \mathbf{sR} + \mathbf{R32} * \mathbf{cR} & \mathbf{R33} \end{bmatrix}$$

$$\text{Eq. 17} \quad \text{maximize}((\mathbf{R31} - \mathbf{R12}) * \mathbf{sR} + (\mathbf{R32} + \mathbf{R11}) * \mathbf{cR})$$

Finally by setting the derivative of **Eq. 17** relative to θ_{rotation} the upper arm rotation can be solved, as shown in **Eq. 18** through **Eq. 20**.

$$\text{Eq. 18} \quad (\mathbf{R31} - \mathbf{R12}) * \mathbf{cR} - (\mathbf{R32} + \mathbf{R11}) * \mathbf{sR} = 0$$

$$\text{Eq. 19} \quad (\mathbf{R31} - \mathbf{R12}) * \mathbf{cR} = (\mathbf{R32} + \mathbf{R11}) * \mathbf{sR}$$

$$\text{Eq. 20} \quad \theta_{\text{rotation}} = \text{atan2}((\mathbf{R31} - \mathbf{R12}), (\mathbf{R32} + \mathbf{R11}))$$

Additionally, to maintain the right hand rule and allow for control of the RHBM, the joint angles of the segments on the right and left hand of the model do not share the same rotational conventions. To fix this problem the raw joint angles are inverted for select joints on the left arm to allow the left and right clinical joint angles to describe the same direction of rotation. The rotation from the torso to shoulder segments requires a 180

degree rotation about the torso Y axis, so an offset is added to the L4 joint angle to maintain the same initial angle. Table 13 shows the conversions required to calculate the robotic and clinical joint angles given the raw joint angle data.

Table 13: Conversion between joint angle conventions (radians)

Raw	Robotic	Clinical
1	R1	1
2	R2 - $\pi/2$	2
3	$R3 + \frac{-\pi}{2} - \text{atan}\left(\frac{RSJC_z}{RSJC_x}\right)$	3
R4	$R4 - \text{atan}\left(\frac{RSJC_x}{RSJC_z}\right)$	R4
R5	R5 - $\pi/2$	R5
R6	R6 - $\pi/2$	R6
R7	R7 - $\pi/2$	$\text{asin}(R(1,3)_{UA})$
R8	R8 - $\pi/2$	$\text{acos}(R(2,3)_{UA})$
R9	R9 - $\pi/2$	$\text{atan}\left(\frac{(R(3,1)_{UA} - R(1,2)_{UA})}{(R(3,2)_{UA} + R(1,1)_{UA})}\right)$
R10	R10 - $\pi/2$	R10
R11	R11 - $\pi/2$	R11
R12	R12	R12
R13	R13 + $\pi/2$	R13
R14	R14	R14
L4	$L4 + \pi - \text{atan}\left(\frac{LSJC_x}{LSJC_z}\right)$	-L4 + π
L5	L5 + $\pi/2$	L5
L6	L6 - $\pi/2$	-L6
L7	L7 - $\pi/2$	$-\text{asin}(R(1,3)_{UA})$
L8	L8 - $\pi/2$	$\text{acos}(R(2,3)_{UA})$
L9	L9 - $\pi/2$	$-\text{atan}\left(\frac{(R(3,1)_{UA} - R(1,2)_{UA})}{(R(3,2)_{UA} + R(1,1)_{UA})}\right)$
L10	L10 - $\pi/2$	-L10
L11	L11 - $\pi/2$	L11
L12	L12	-L12
L13	L13 + $\pi/2$	-L13
L14	L14	L14

Raw joint angles are calculated from the segment rotations by *autoFindTheta.m*, the robotic joint angles are calculated in *CreateUBM.m* using the raw angles and the Denavit and Hartenburg parameters, and the clinical joint angles are calculated by *ROMtest.m*, Appendix B.14, at the same time the range of motion for each subject is calculated.

3.5 Saving the Model Data

The final model uses the Denavit and Hartenberg parameters defined in Table 12 and the robotic joint angles as described in Section 3.3. These variables are saved into the *Train* structure as *Train.(subjectID).RUpperbody*, *Train.(subjectID).LUpperbody*, *Train.(subjectID).(trialname).RTheta*, and *Train.(subjectID).(trialname).LTheta*, in a Matlab file *(subjectID)UpperBodyModel.mat*. The training and testing functions for the control are able to run using only these variables, and all other variables are stored into *(subjectID)Data.mat*. The workspace is then cleared before running the process for the next subject. This process minimizes the amount of data in the workspace at any given time and stores all of the data for reference if needed. Since some of the training algorithms are memory intensive, preserving the memory available is crucial.

Chapter 4: Motion Analysis and Segment Length Results

This chapter presents the results from the motion capture, subject measurements, and functional joint center calculations. The clinical joint angles of the un-braced control subjects were compared to the braced control subjects, and the amputee subjects. The subject anthropometric measurements were correlated to the segment lengths as calculated by the functional joint center method. Significant differences were determined by analysis of variance and multiple comparison tests in Matlab using the *anovan.m* and *multcompare.m* function with a 95% confidence interval.

4.1 Control Subjects' Range of Motion

The RoM of each joint is an indication of that joint's health and ability to add to the workspace of the upper body. In this study the RoM of each joint of the upper body was analyzed for several reasons. The RoM relative to averages of the control subjects indicated the impedance / capability of the prosthesis and socket, which was then be used to control the capability of the model in the control algorithms. The angles given in this section follow the conventions of the clinical joint angles, as given in Section 3.4, which allow for the left and right arm to be analyzed as dominant or sound side, versus non-dominant or prosthetic side. The average and standard deviation of the minimum, maximum, and RoM of the un-braced control subjects are given in Table 14. For this section all motions were evaluated relative to the dominant (D) or non-dominant (N) arm, rather than the right (R) or left (L). No significant difference ($p < 0.05$) was found between dominant and non-dominant joint RoM for un-braced control subjects.

Table 14: Range of motion for control subjects (degrees)

Segment	Description	Joint	Min		Max		RoM	
			Avg.	S.D.	Avg.	S.D.	Avg.	S.D.
Torso	Flexion	1	-43	15	34	13	76	23
	Lateral Flexion	2	-33	9	33	8	66	17
	Rotation	3	-45	12	43	10	88	20
Shoulder	Protraction	D4	-32	8	40	14	72	15
		N4	-30	6	42	13	72	13
	Depression	D5	-54	9	38	9	92	10
		N5	-53	9	45	9	99	10
	Rotation	D6	-26	9	66	17	92	18
		N6	-25	9	61	17	86	14
Upper Arm	Flexion	D7	-48	14	78	13	126	21
		N7	-44	11	75	20	120	26
	Elevation	D8	-8	7	72	7	80	6
		N8	-13	9	66	8	79	10
	Rotation	D9	-73	13	63	28	136	31
		N9	-69	14	53	18	122	23
Forearm	Flexion	D10	12	7	149	5	137	7
		N10	9	7	149	5	140	6
	Carrying Angle	D11	-14	3	10	6	24	6
		N11	-14	4	9	6	23	4
	Pronation	D12	-74	24	78	28	152	37
		N12	-66	14	63	8	130	16
Hand	Flexion	D13	-69	14	58	14	126	11
		N13	-55	15	71	12	126	13
	Abduction	D14	-30	12	7	11	36	5
		N14	-12	17	24	10	37	16

4.1.1 Braced Subjects' Range of Motion

For the braced trials an arm brace was attached to the subjects' dominant arm. The subjects were instructed not to force the brace movement by overpowering the brace material, but rather to move through any slack in the brace, until they felt moderate resistance. The brace was a Restorative Care of America Incorporated (St. Petersburg, FL) wrist and elbow brace, where the elbow was not restricted. This configuration restricts the movement of forearm pronation, and wrist flexion and extension. Significant differences ($p < 0.05$) in the subject range of motion between braced and un-braced subjects were found between braced arm joints D8, upper arm abduction, D10, elbow

flexion, D12-14, forearm pronation, wrist flexion, wrist abduction of the braced arm, and N10, un-braced arm elbow flexion. This implies that the brace had a significant impact on the braced arm. Additionally there was a significant difference between the braced and un-braced arms when wearing the brace for joints 8, upper arm abduction, 10, elbow flexion, and 12-14, forearm pronation, wrist flexion, and wrist abduction respectively. RoM results for braced subjects are shown in Table 15.

Table 15: Range of motion of braced control subjects (degrees)

Segment	Description	Joint	Min		Max		RoM	
			Avg.	S.D.	Avg.	S.D.	Avg.	S.D.
Torso	Flexion	1	-35	23	32	15	67	31
	Lateral Flexion	2	-31	11	30	16	61	26
	Rotation	3	-42	21	43	15	85	35
Shoulder	Protraction	D4	-33	7	32	20	65	23
		N4	-27	12	40	13	67	21
	Depression	D5	-46	17	32	11	78	23
		N5	-47	18	39	13	86	26
	Rotation	D6	-24	6	60	24	84	25
		N6	-24	6	53	24	76	24
Upper Arm	Flexion	D7	-31	30	76	16	107	40
		N7	-40	20	74	20	114	37
	Elevation	D8	-5	13	59	9	65	11
		N8	-12	11	64	9	76	13
	Rotation	D9	-61	32	54	16	115	38
		N9	-69	24	52	15	121	28
Forearm	Flexion	D10	24	13	132	9	108	8
		N10	14	8	145	8	131	11
	Carrying Angle	D11	-8	8	13	14	21	9
		N11	-14	6	7	8	21	7
	Pronation	D12	-13	26	21	27	34	14
Hand	Flexion	N12	-67	11	62	14	130	22
		D13	-18	52	10	47	28	31
	Abduction	N13	-51	13	66	13	117	15
		D14	-9	27	7	22	16	10
		N14	-11	20	20	15	30	10

With the exception of forearm pronation of the non-braced limb the average RoM for all joints of the braced subject trials was less than the average RoM of the non-braced subjects. Figure 17 also shows the impact of bracing on RoM in terms of the average

maximum and minimum joint angles with standard deviation. Variation in the braced position of the forearm and hand between subjects contributes to the high standard deviation in the maximum and minimum joint angles for the braced forearm and wrist joints (D12-D14). The standard deviation of the RoM of the braced joints was less than the standard deviation of the maximum and minimum joint angles.

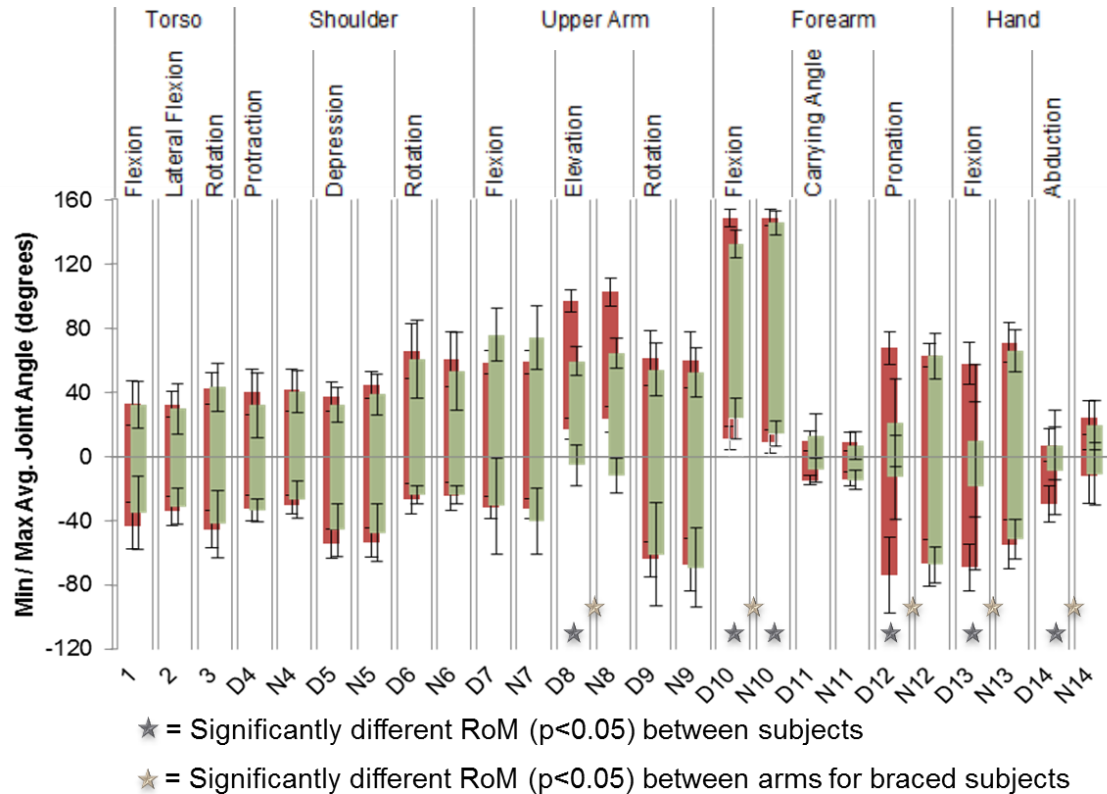


Figure 17: Impact of bracing on range of motion

4.2 Amputee Subjects' Range of Motion

This section compiles all of the results for the amputee subjects in the sample. Due to the limited number of amputees included, these data are largely observational and may not be widely generalizable at this time. A larger sample is recommended for future work.

Amputee subjects exhibited a decrease in RoM of the prosthesis relative to the control subjects, on their prosthetic side. In this section each joint number is listed as the dominant (D) or prosthetic (P) side.

4.2.1 Subject R01

This subject was the only transradial amputee to complete the study. His RoM was very similar to the control subjects' with exceptions to the wrist and forearm of his prosthetic arm, as shown in Figure 18.

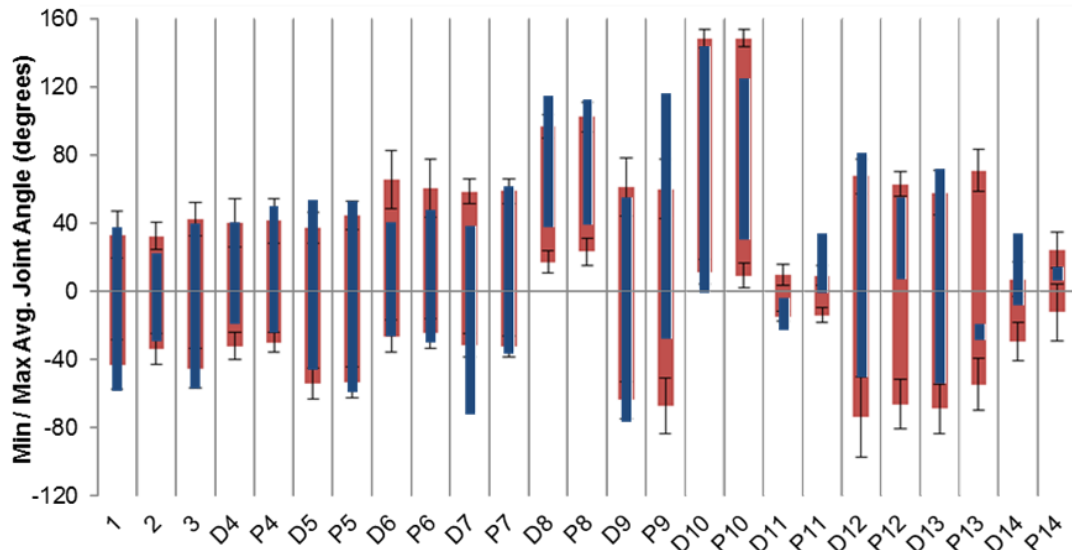


Figure 18: RoM of subject RH01 (blue) superimposed over control RoM (red)

The angle of shoulder rotation (P9) was elevated above the control range, with the min and max both above the standard deviation of the control subjects. This may be caused by the alignment of the prosthesis relative to the anatomical elbow, or potentially contributed to misplacement of the markers due to the inability to palpate the epicondyle of the elbow, as they were covered by the socket. The motion of the wrist of the prosthesis was primarily passive and actuated by the contralateral limb between trials.

4.2.2 Subject H01

This subject was the only bilateral amputee in the tested group. The extreme reduction in RoM of the distal limb joints, with the exception of the elbow, can be seen in Figure 19.

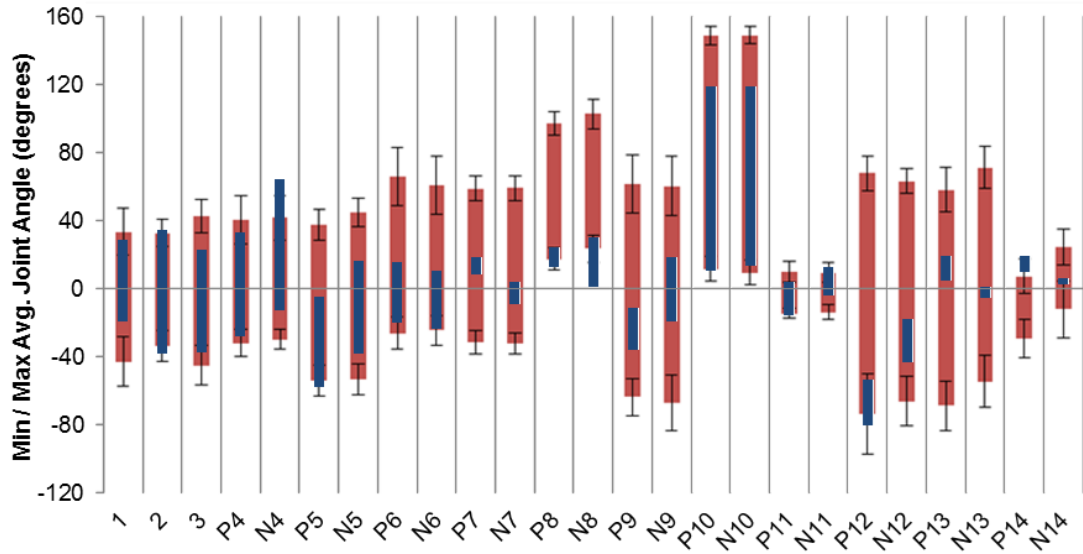


Figure 19: RoM of subject H01 (blue) superimposed over control RoM (red)

The range of motion of the ADL tasks shows the limitation of the prosthesis, with little or no motion available at the wrist, and restricted motion of the shoulder. Impressively, this subject was able to complete all of the ADL tasks, with what seemed to be less difficulty than some of the other amputee subjects. This may be due to the fact that, because he was a bilateral amputee, he has been forced to use his prostheses for all of the tasks in his daily life. The unilateral amputee subjects have the option and likely elect to use their intact contralateral limb for most activities in their daily life.

4.2.3 Subject H02

This subject had a unilateral transhumeral amputation, and used a body-powered prosthesis. His RoM was reduced, but not nearly as drastically as subject H01. Subject H02 had a large range of motion of the Torso (1-3, on the higher end relative to the control subjects) and some decreased motion of the scapular complexes (4-6), but maintained a moderate range of motion of the upper arm about the glenohumeral joints (7-9). Motion from upper arm rotation and about the wrist (P9, P12-14) came mostly

from passive joints moving under gravity, and from actuation of the hook. The RoM of subject H02 relative to control averages is given in Figure 20.

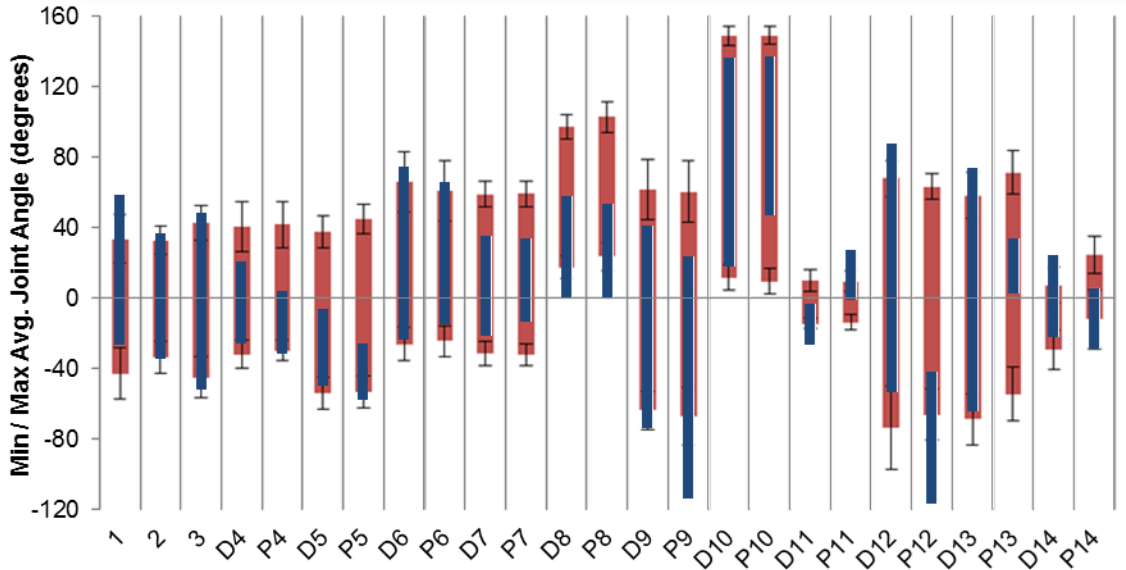


Figure 20: RoM of subject H02 (blue) superimposed over control RoM (red)

Additionally, subject H02 was in very good health. He had a highly muscular upper body, and reported performing 300 push-ups 5 days a week using a push-up rig that he designed and built himself. Despite being well conditioned, he did have some difficulty with the unilateral ADL tasks, which the protocol required each subject to complete with the prosthetic side.

4.2.4 Subject H03

This subject was a unilateral transhumeral amputee with a myoelectric prosthesis. This subject had a reduced range of motion for joints primarily on his prosthetic side. Shoulder protraction and elevation (P4-5), upper arm flexion, abduction, and rotation (P7-9), as well as wrist flexion and abduction (P13-14), all had decreased range of motion relative to controls and the contralateral side. Forearm rotation of the prosthesis had continuous motion; therefore there was no limit on the RoM of joint P12. However, forearm rotation

was the only means of positioning the gripper relative to the forearm. The RoM of subject H03 relative to control averages is given in Figure 21.

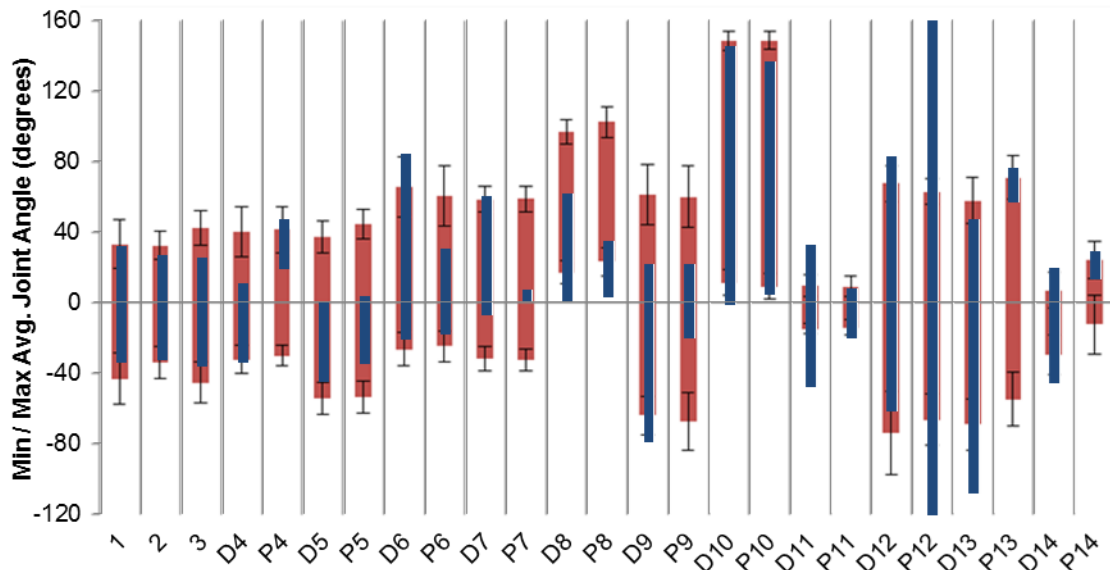


Figure 21: RoM of subject H03 (blue) superimposed over control RoM (red)

4.3 Activities of Daily Living Results and Observations

The range of motion and qualitative observations of subjects performing the activities of daily living is discussed in this section. Difficulties and solutions to obstacles associated with each task are also presented. The compensatory motion is defined as the excessive motion of a proximal joint to compensate for the limited motion of a distal joint. The use of motion analysis for the detection of compensatory motions has been established for the upper body [29, 99]. Compensatory motion is categorized by a significant increase ($p < 0.05$) in RoM of the proximal limb, and a significant decrease ($p < 0.05$) in RoM of the distal limb. Compensatory motion can be seen in all of the ADLs evaluated in this study, except the lifting the laundry basket task.

4.3.1 Brushing Hair

The braced subjects had a significantly increased RoM for scapular rotation (joint D6) and a significantly decreased RoM for elbow flexion, forearm pronation, and wrist flexion (joints D10, D12, and D13). For amputee subjects, the most frequently observed difficulty with the grooming task involved the acquisition of the brush. Most amputee subjects had to start with the brush in hand or transfer the brush to the prosthesis with their contralateral limb. Some subjects, primarily within the transhumeral group, had difficulty abducting their arm sufficiently to raise the brush to the top and back of their head. Primary compensation strategy for amputees seems to involve increased motion of scapular elevation and protraction. Figure 22 show the range of motion of the un-braced and braced control subjects respectively.

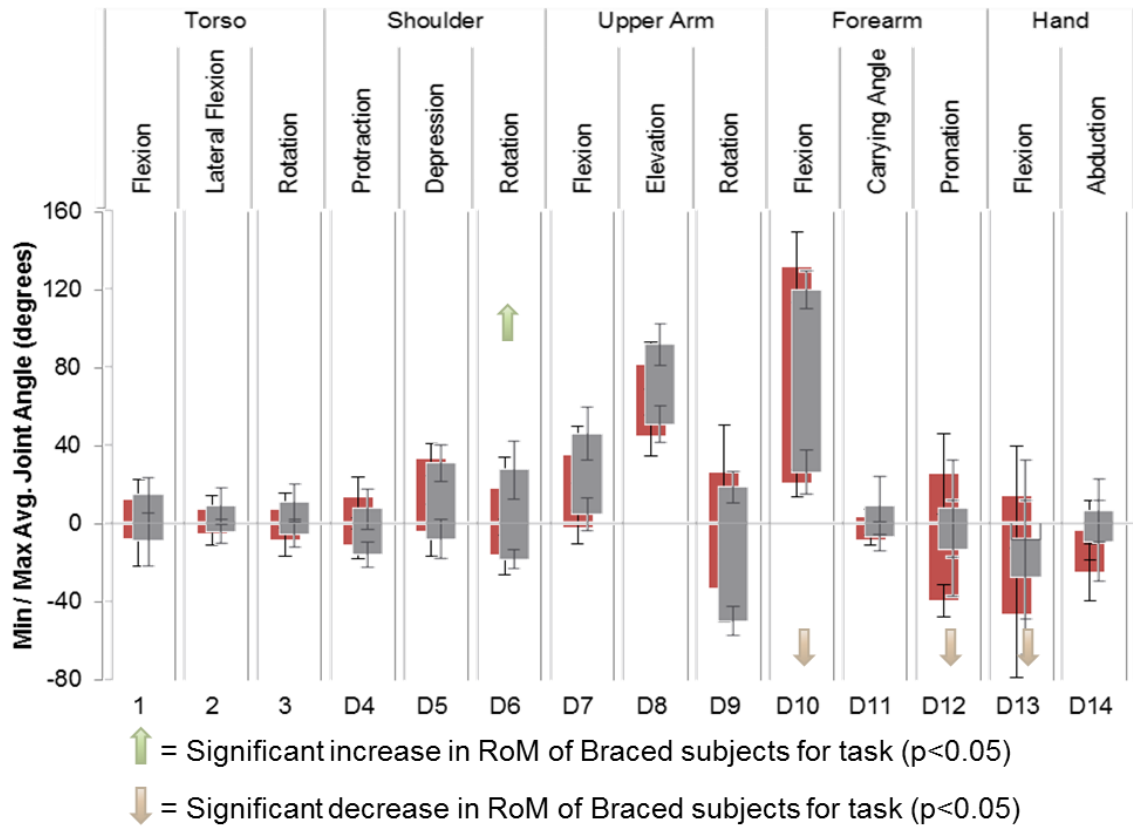


Figure 22: Impact of bracing on dominant arm for brushing task

4.3.2 Drinking From a Cup

For the drinking task, the braced subjects showed a significant increase in torso rotation, scapular rotation, and upperarm rotation (joints 3, D6, and D9). However, the range of motion of the torso for both braced and un-braced subjects is small for this task. There was also a significant decrease in the RoM of forearm pronation, wrist flexion, and wrist abduction (joints D12-14). This task was easily completed by the majority of the subjects. However some subjects did not bring the cup entirely to the mouth. Subjects with high level transhumeral amputations were the most likely to have difficulty with this task. Since an empty cup was used there is potential for the subjects to be able to complete the task in the lab while still having difficulty in everyday situations. The RoMs of the braced and un-braced control subjects for drinking are given in Figure 23.

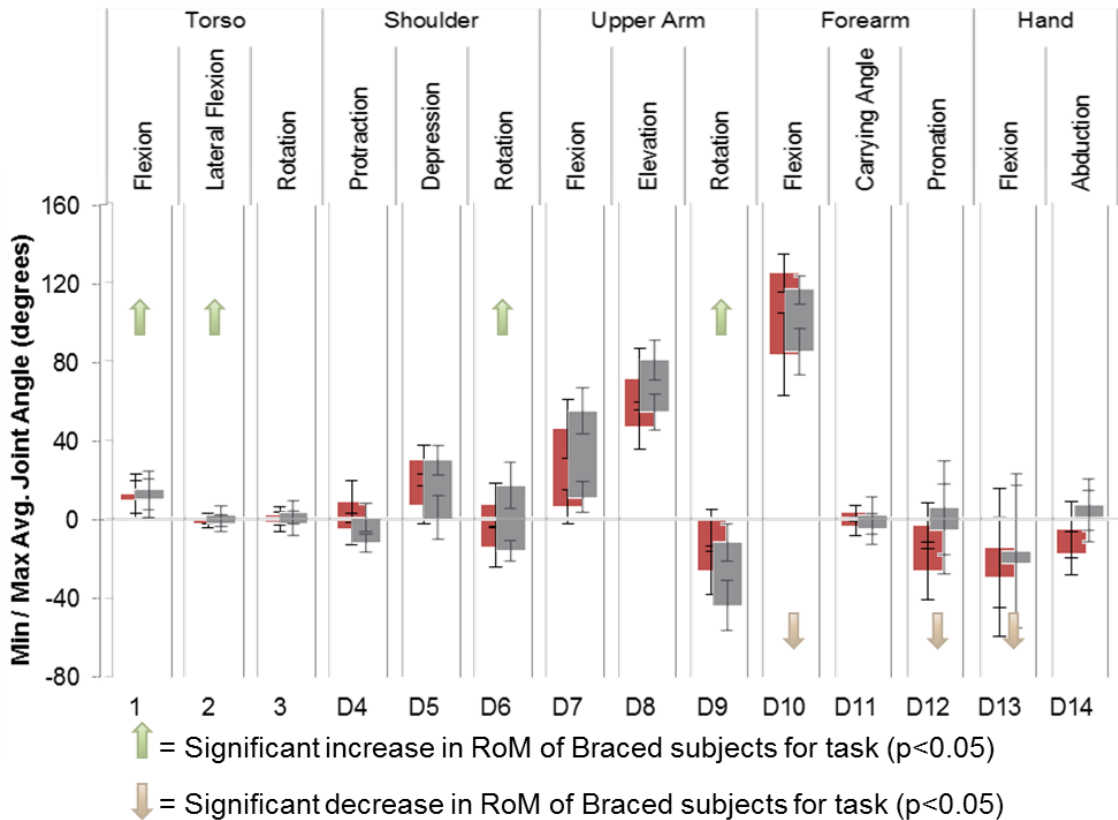


Figure 23: Impact of bracing on dominant arm for drinking task

4.3.3 Eating With a Knife and Fork

For this task there was a significant increase in the RoM for torso rotation, scapular abduction, scapular elevation, scapular rotation, upperarm abduction, and upperarm rotation (3, D4-6, D8-9) of the dominant / braced side, and in elbow flexion (N10) of the non-dominant/un-braced side. The braced forearm pronation, wrist flexion and wrist abduction (D12-14) showed a significant decrease in RoM. Similar to the brushing hair task, the eating task often required the pre-positioning of the utensil prior to the subject being able to complete the task. Unilateral amputees were able to position the utensils using their contralateral limb; however the bilateral amputee received help primarily to preserve time between task collections. Since this was a bilateral task the range of motion of all joints is given in Figure 24 for braced and un-braced subjects.

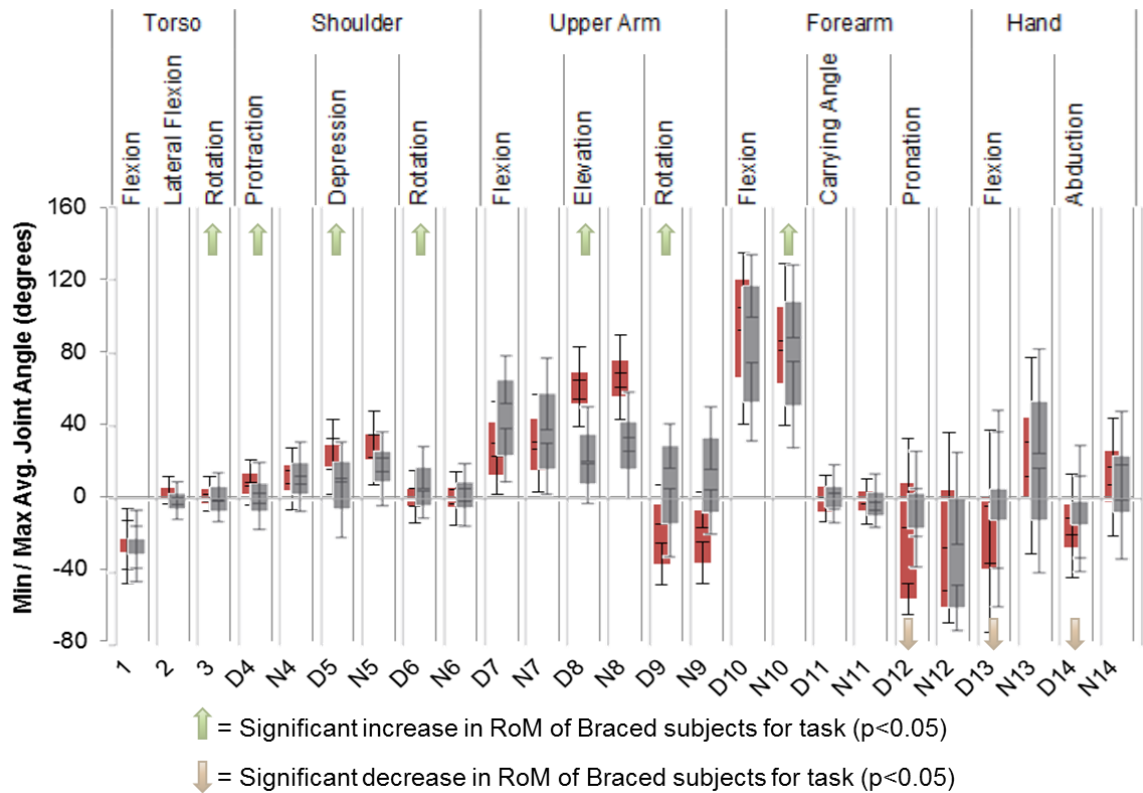


Figure 24: Impact of bracing on dominant and non-dominant arm for eating task

4.3.4 Lifting a Laundry Basket

There was no significant increase in RoM for braced subjects performing the lifting task. A significant decrease in upper arm abduction, elbow flexion, forearm pronation, wrist flexion, and wrist abduction for the braced / dominant arm (D8, D10, and D12-14) was observed. The laundry basket lifting task presented a greater challenge to users fitted with electrically controlled prosthesis. They tended to have to open and close the prosthesis after positioning their hand near the handles of the basket, and in one case had great difficulty controlling the prosthesis while bent over. This is possibly due to the control sensor not contacting the subject's arm properly in that position. Body-powered prosthesis users would pre-position their terminal device before performing the task and would either simply hooked the handles or were able to open their gripper while bending to grab the basket. This task required the greatest sum of joint angle RoM to complete. The RoM of un-braced and braced subjects for the lifting task is given in Figure 25.

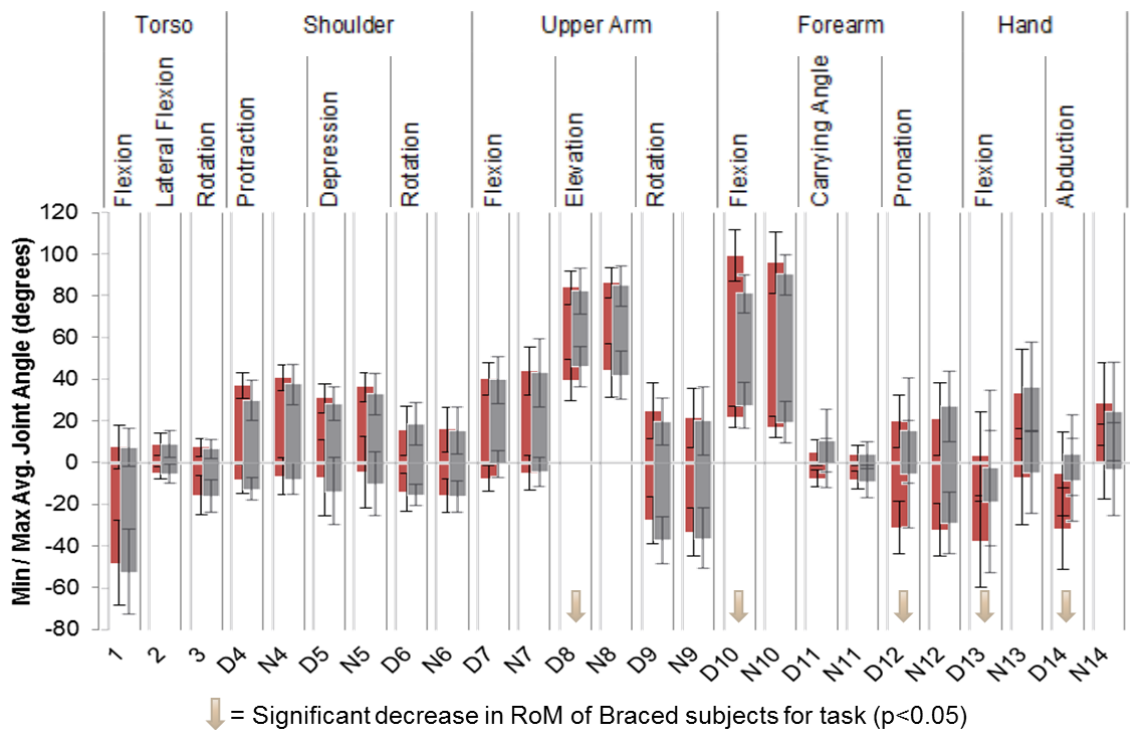


Figure 25: Impact of bracing on dominant and non-dominant arm for lifting task

4.3.5 Opening a Door

Significant increases in the RoM of torso flexion, torso lateral flexion, scapular rotation, and upper arm rotation (joints 1, 2, D6, and D9) were observed for braced subjects during the door opening task. Significant decreases in elbow flexion, forearm pronation, and wrist flexion (joints D10, and D12-13) were also observed for the braced subjects. The positioning of the door made recording the task somewhat difficult and marker dropout was common. To increase visibility the superior section of the door was removed just above the second hinge. For subjects who were unable to open the door with a traditional round knob, a secondary lever handle was prepared. Only one subject required the lever handle to open the door. The RoM of un-braced and braced subjects performing the opening task is given in Figure 26.

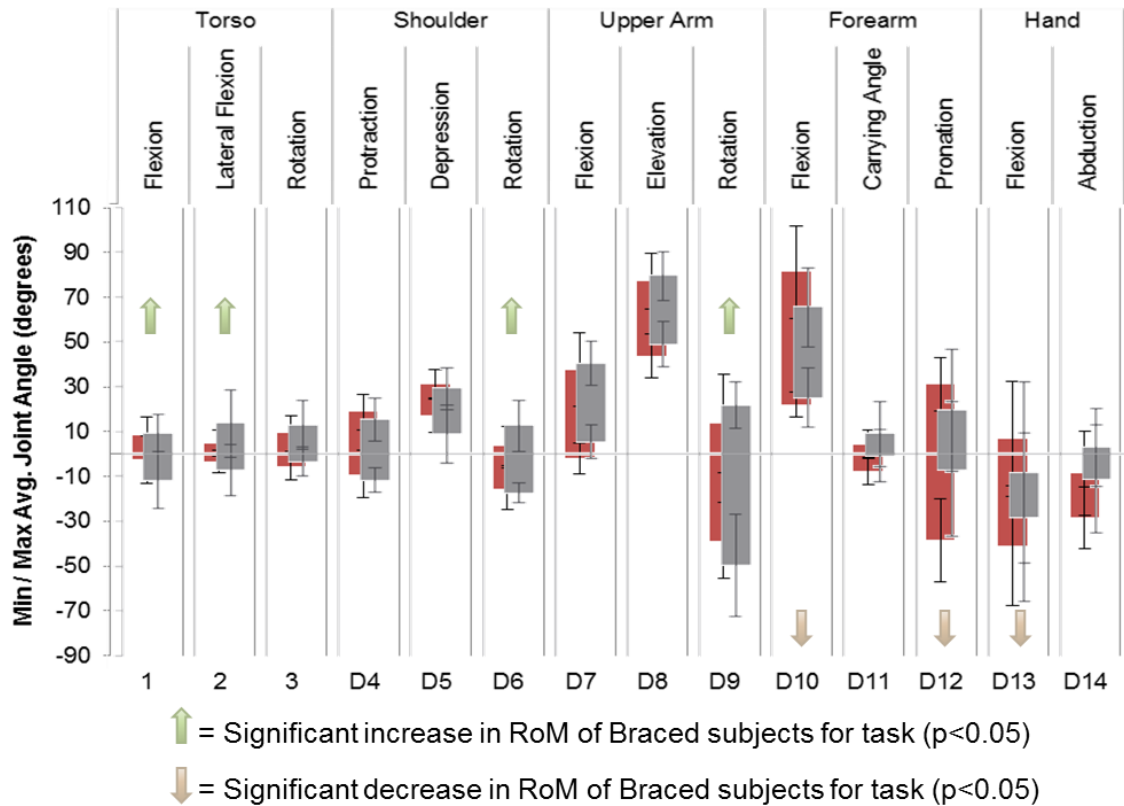


Figure 26: Impact of bracing on dominant arm for opening task

4.4 Subject Measurements

Recorded measurements for control subjects are given in Table 16. For this study, only small variances in the upper arm lengths were observed between subjects. Since these measurements were recorded manually there was a ± 1 cm margin of error, which may account for left-right asymmetry.

Table 16: Control subject anthropometric measurements (cm)

Subject	CC	UCP	UCD	FC	SC	A2E	X2E	E2S	E2T	S2T	
C01	Right	90	28	22	20	16	29	21	26	39	13
	Left		27	22	22	16	31	22	26	39	13
C02	Right	94	34	27	26	17	32	24	27	40	14
	Left		30	27	25	16	33	23	27	40	14
C03	Right	99	39	29	30	18	30	23	28	40	13
	Left		35	30	28	19	29	26	27	39	12
C04	Right	97	35	27	27	17	32	24	27	40	14
	Left		32	26	27	16	32	22	27	41	15
C05	Right	112	40	34	33	19	31	24	27	41	13
	Left		41	35	31	19	33	22	28	41	14
C06	Right	108	36	30	30	19	32	25	27	40	12
	Left		35	30	29	19	34	23	28	41	13
C07	Right	99	31	26	25	16	28	18	23	33	11
	Left		32	26	23	15	28	19	23	34	12
C08	Right	99	31	30	28	17	31	22	26	39	14
	Left		32	29	26	17	31	23	26	39	14
C09	Right	107	38	30	29	18	32	23	28	39	14
	Left		39	29	28	17	31	23	27	38	14
C10	Right	94	29	26	23	16	31	21	25	37	14
	Left		31	25	22	15	31	21	26	37	13
Avg.	100	34	28	26	17	31	22	26	39	13	
S.D.	7.0	4.1	3.2	3.3	1.3	1.7	1.9	1.4	2.2	0.9	

**Descriptions for anatomical measurements are given in Table 6*

The amputees included in this study varied considerably in residual limb anthropometry.

Residual limb measurements collected from the amputee subjects are given in Table 17.

Subject H01 was a bilateral amputee, so measurement for the right and left residual limb

are included. Subject H03's residual limb was so short that only one practical

measurement of residual limb circumference could be obtained.

Table 17: Amputee subject residual limb measurements (cm)

Subject		PRLC	DRLC	A2RL	X2RL	E2RL
R01	Right	-	-	-	-	23.2
H01	Right	32	28	13	7	-
	Left	31	26.5	17	8	-
H02	Right	31	20	26	18	-
H03	Right	26	-	12	4	-

4.5 Functional Joint Center Segment Geometry

The torso joint center is given relative to the pelvis segment. The shoulder joint center is given relative to the torso segment. The shoulder, upper arm, and forearm segment lengths are the distance between joint centers, since the joint centers are defined along the Z-axis of the proximal segments. The values for the segment parameters are given in Table 18 and were found with the functional joint center method, described in Chapter 3:.

Table 18: Segment geometry parameters from function joint centers (cm)

		Torso Joint Center			Shoulder Joint Center			Segment Length		
		X	Y	Z	X	Y	Z	SHO	UA	FA
C01	Right	10	12	0	1	29	7	12	27	26
	Left	10	12	0	0	30	7	11	27	26
C02	Right	10	11	1	0	31	9	12	26	26
	Left	10	11	1	0	33	9	11	27	26
C03	Right	10	12	0	1	30	7	14	27	27
	Left	10	12	0	1	29	7	15	27	27
C04	Right	13	11	-1	1	34	7	13	26	27
	Left	13	11	-1	1	33	6	15	25	27
C05	Right	8	16	-1	2	25	7	15	26	29
	Left	8	16	-1	0	25	8	16	25	29
C06	Right	11	12	1	0	32	8	14	26	28
	Left	11	12	1	0	32	8	14	25	26
C07	Right	6	7	-1	1	30	7	10	22	22
	Left	6	7	-1	1	30	7	12	21	22
C08	Right	7	15	-3	1	21	7	12	26	27
	Left	7	15	-3	2	21	6	13	26	26
C09	Right	14	9	0	0	30	8	13	26	28
	Left	14	9	0	-1	30	9	13	24	27
C10	Right	12	1	0	1	32	6	12	26	24
	Left	12	1	0	1	33	6	12	25	24
	Avg.	10	11	0	1	30	7	13	25	26
	S.D.	3	4	1	1	4	1	1	2	2

In order to facilitate the implementation of the RHBM for subjects who have not completed the RoM motion capture, the segment lengths and joint center locations were correlated to the measured subject's limb lengths using the Pearson product moment correlation, or R^2 value in Microsoft Excel. Data from the right and left side were used in a single correlation since the relations between anatomical measures and segment lengths were assumed to be symmetrical. The correlations found are given in Table 19. Most anatomical measures had a low correlation relative to the calculated segment lengths.

Table 19: R^2 correlations for segment lengths

	Torso Center			Shoulder Center			Segment Lengths		
	X	Y	Z	X	Y	Z	Sho.	UPA	FA
Height	0.18	0.74*	0.23	0.00	-0.17	0.39	0.76*	0.62	0.91*
CC	0.00	0.37	0.14	-0.18	-0.27	0.36	0.61	-0.19	0.51
UCP	0.14	0.36	0.21	-0.19	-0.15	0.42	0.68	-0.02	0.62
UCD	-0.10	0.45	-0.03	0.04	-0.43*	0.26	0.69	0.00	0.61
FC	0.02	0.54	0.10	-0.08	-0.27	0.34	0.69	0.10	0.72
SC	0.09	0.58	0.31	-0.04	-0.25	0.34	0.75	0.40	0.75
A2E	0.47	0.32	0.33	-0.27*	0.15	0.48*	0.43	0.39	0.64
X2E	0.36	0.46	0.34	-0.07	0.00	0.34	0.56	0.70	0.78
E2S	0.58*	0.41	0.43*	-0.19	0.10	0.41	0.65	0.68	0.88
E2T	0.36	0.64	0.22	0.01	-0.06	0.31	0.70	0.71*	0.89
S2T	0.41	0.22	-0.18	0.02	-0.05	0.06	0.31	0.37	0.51

**Values represent highest correlation for the given model length.*

To increase the accuracy and reliability for use in future studies, the measured lengths were then used in a multivariable linear regression in order to more accurately determine the segment lengths in relation to manual measurements. The regression was also forced to a zero intercept to increase the stability of the solution given the inclusion / exclusion of subjects. The subject height and chest circumference, CC, were used to estimate the torso center, shoulder center, and shoulder length. The distance from the acromion to lateral epicondyle of the humerus, A2E, and the distance from the axilla to the elbow to the medial humeral epicondyle, X2E, was used to generate the upper arm length, UPA.

The distance from the lateral epicondyle to the radial styloid process, E2S, and to the thumb, E2T, was used to generate the forearm length, FA. The RHBM parameters were obtained from the functional joint center methods, but can also be entered manually from calculations based on the height of the subject or from the anthropometric correlations given in Eq. 21 through Eq. 30. Any units can be used in the following equations; however, the same units must be used for all measurements. The torso joint center in Z-axis direction and shoulder joint center in X-axis direction were set to zero because the subject variation was larger than the average value.

$$\text{Eq. 21} \quad \mathbf{Torso\,JC(X) = -0.0389 * CC + 0.0791 * Height}$$

$$\text{Eq. 22} \quad \mathbf{Torso\,JC(Y) = -0.0602 * CC + 0.0949 * Height}$$

$$\text{Eq. 23} \quad \mathbf{Torso\,JC(Z) = 0}$$

$$\text{Eq. 24} \quad \mathbf{Shoulder\,JC(X) = 0}$$

$$\text{Eq. 25} \quad \mathbf{Shoulder\,JC(Y) = -0.0632 * CC + 0.2029 * Height}$$

$$\text{Eq. 26} \quad \mathbf{Shoulder\,JC(Z)_{Right} = -0.241 * CC + 0.540 * Height}$$

$$\text{Eq. 27} \quad \mathbf{Shoulder\,JC(Z)_{Left} = 0.241 * CC - 0.540 * Height}$$

$$\text{Eq. 28} \quad \mathbf{Shoulder\,Length = 0.0479 * CC + 0.0457 * Height}$$

$$\text{Eq. 29} \quad \mathbf{Upper\,Arm\,Length = 0.654 * X2E + 0.350 * A2E}$$

$$\text{Eq. 30} \quad \mathbf{Forearm\,Length = 0.434 * E2T + 0.365 * E2S}$$

The accuracy of the RHBM reconstruction with RoM data relative to the recorded segment locations, using the functional joint centers as segment origins, is very high with the average end effector reconstruction error of less than 1mm. Using the anthropometric correlations, the model accuracy decreases to an average error of 26 mm for the tested subjects. Using literature average segment length relative to height, for a 50th percentile

male [63], results in an average error of approximately 164 mm for the tested subjects. The reconstruction error using literature averages for the height ratio is somewhat exaggerated. Since RHBM was designed to use the functional joint center data, which orients the segments based to align the joint center, and the literature data were given relative to surface landmarks.

4.6 Comparison with Vicon Plug-In Gait

To help validate the clinical relevance of the joint angles calculated by the functional joint center based model, the joint angles were compared to the joint angles calculated using the Vicon Plug-in Gait [54]. The Plug-in Gait is a commonly used program for motion analysis studies. Therefore, using similar conventions will allow for comparison of the RHBM outputs to existing studies. To facilitate the comparison, subject C01 was fitted with a 29 marker upper body marker set that contained the standard marker set for the upper body portion of the Plug-in Gait and the markers required for the functional joint center algorithm. Anthropometric measurements required for the Plug-in Gait were recorded by hand using a standard tape measure prior to motion analysis. The subject completed the same eight RoM tasks as specified in Section 2.5. The raw position data were filtered with a weighted moving average digital filter. The Plug-in Gait algorithm was used to find torso, shoulder, elbow, and wrist angles within the Vicon Bodybuilder software. Matlab was used to find the functional joint centers and to define the upper body segments based on joint center and marker positions, as defined in Section 3.2. The rotational conventions defined in the Plug-in Gait manual were then used to find the joint angles given the RHBM segments. The difference in joint angles was a function of the difference between the segment definitions in the Plug-in Gait and functional joint center

methods. After analyzing the Plug-in Gait coordinate systems, the conventions of the RHBM segments were adapted to match by re-defining the axes and rotational orders. The transformation of joint angles from the RHBM convention to the Plug-in Gait convention was achieved by using the same joint rotation conventions as established in the Plug-in Gait as a post-hoc analysis of the segment rotational matrices after all of the segments were defined. The distances between the segment origins of the Plug-in Gait were also analyzed. Variation of the distances between segments, or segment lengths, leads to error between motion reconstructions and recorded data when implementing the data in a rigid body model such as the RHBM.

The functional joint center algorithm was able to generate accurate joint centers for the upper body segments of all 10 control subjects, resulting in average position reconstruction error of less than 1mm between the forward kinematics of the RHBM and the hand segment locations [100]. The average difference between the joint angles of the Plug-in Gait and the functional joint center methods for each joint is presented in Table 20. The angles calculated from the functional joint center method closely matched the Plug-in Gait for all joints except for the wrist. The hand and forearm segments were defined differently between the two models, primarily due to the conventional differences caused by assumptions for elbow motion. The average difference for all joints except the wrist was $6.0 \pm 3.1^\circ$. The wrist had a much larger average difference of 39.9° .

Table 20: Average difference between joint angle conventions (degrees)

Torso			Left Shoulder			Right Shoulder			Elbow Flex		Wrist Pron	
Flex	LatF	Rota	Flex	Abdu	Rota	Flex	Abdu	Rota	Left	Right	Left	Right
3.3	1.4	3.8	9.3	11.8	10.1	8.4	5.2	4.5	6.8	5.1	2.8	5.3

Figure 27 shows left elbow flexion for the elbow flexion task using the functional joint center and the Plug-in Gait methods. The component rotations of a joint were coupled,

therefore a difference in one rotation (i.e. shoulder flexion) will result in differences for all rotations associated with that segment (i.e. shoulder abduction & rotation). This is the typical form of the difference between methods which is caused by the difference in segment orientation. The axes were similar in orientation but not exact since they used different markers in the segment definition. The error was somewhat systemic, usually consisting of an offset as a function of the joint angles of the associated segments.

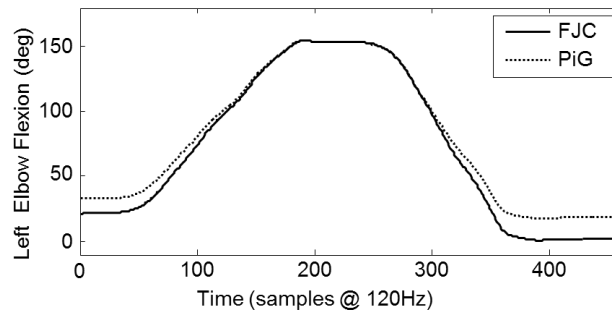


Figure 27: Left elbow flexion for functional joint center and Plug-in Gait.

To find the variation in segment lengths of the Plug-in Gait model, the distances between segment origins of the torso, clavicle, humerus, radius, and hand segments were found. The mean, standard deviation, minimum, and maximum distance between the segments origins are presented in Table 21. The variations in segment lengths were normally small, but in extreme ranges of motion the variation can become large.

Table 21: Variation in Plug-in Gait segment lengths for RoM tasks (mm)

	TRX to RCL	RCL to RHU	RHU to RRA	RRA to RHN	TRX to LCL	CLCL to LHU	LHU to LRA	LRA to LHN
Mean	184	284	261	177	188	284	267	154
S.D.	11	12	9	30	9	11	9	10
Min	142	198	171	127	136	205	186	98
Max	210	321	269	533	238	324	276	200

Investigation of the source of highest variation and joint angle error seems to occur primarily in instances where the Plug-in Gait behaves abnormally. The exact cause was unknown as the calculations of the Plug-in Gait are proprietary, but the error may be

partially caused by interpolation during instances of marker dropout. Figure 28 shows an example of a trial with abnormally high error caused by marker dropout. Although most trials did not contain significant marker dropout, all points where both models calculated segment kinematics were used.

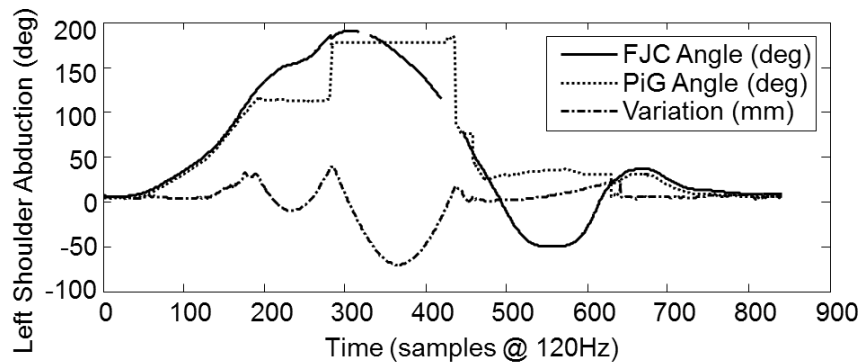


Figure 28: Plug-in Gait abnormality and associated variation in segment length

Chapter 5: Methods for Predicting Human Motion

In this chapter the formulations of the least norm (LN), weighted least norm (WLN), probability density gradient projection of the null space (GP), and artificial neural network (NN) methods for reconstructing human motion are presented. This study was developed to increase the accuracy and realism of upper body simulations and to make the results easily verifiable. Other studies have been done to predict upper-limb motion but they often restrict the origin of the simulation to the shoulder joint and therefore lack the necessary complexity to predict compensatory motions [43, 45]. The kinematics of the human upper body are highly redundant. There are an infinite number of configurations in joint angle space that can produce the same position and orientation of the hand in Cartesian space. Therefore, there are an infinite number of solutions to the inverse kinematics of the upper body. The range of solutions that are human-like is smaller than the total number of possible solutions. To maintain a human pose it is necessary to find joint angles that not only satisfy the kinematic constraints, but also are realistic human poses. This challenge has been the subject of study in a variety of fields, and several solutions have been presented [12, 15, 16, 43, 101-103]. However the task of predicting motion of prostheses users possesses unique challenges. The kinematics of an upper limb amputee is dependent on the RoM of their prosthesis, their ability to utilize that prosthesis, and the RoM of their body including that of the residual limb. When predicting the movement of the upper body for prosthesis simulation, the functional capabilities of individual and of the prosthetic device must be considered.

To properly predict the motion of an upper limb prosthesis user, a highly adaptable control algorithm must be selected. The model must consider the user, the prosthesis, and the task. To select the best algorithms several techniques for the inverse kinematic control of the upper body model were evaluated and the technique that produced the best results was selected for use in the simulation. Results of the individual control methodologies were analyzed for potential integration of methods, and the robustness of each control algorithm was also evaluated by varying the number subjects used to train the algorithms.

5.1 Training Data Filtering and Preprocessing

TrainBi.m, Appendix B.16, compiles the data from motion analysis into the form used in the training and testing algorithms. Gaps in the joint angle data are filled with *FilGap.m*, Appendix B.17. Any trials with more than a total of one second of gaps, are segmented into smaller sections that have no gaps. In order to include data from all of the subjects it was necessary to condense the number of points in the training set. If we consider the braced subjects to be additional subjects, there are 24 subject data sets. Each subject performs 5 ADLs, each ADL is repeated 3 times, the model has 25 DoFs, and most trials are approximately 3 seconds long, with 120 points for each joint per second. This led to approximately 3.2 million pieces of data that could be used for training. To decrease the amount of time required to train and test the various control algorithms the amount of data were reduced. To facilitate reduction of the number of training points the *condense.m*, Appendix B.18, algorithm was used to effectively reduce the sampling rate of the data collected, from 120 Hz to 20 Hz, by replacing every 6 data points in the time series with an average of the data points for that series.

5.2 Defining Error

For this study the accuracy of each method was defined as being inversely related to the error of the predicted joint angles. The error of each method was defined by the joint square error, Eq. 31, or the root mean squared (RMS) error, Eq. 32, of the predicted joint angles, θ_x , relative to the recorded joint angles from the motion analysis data, θ_{MA} , where N is the number of points in the reported error. This operation can be calculated on a model or joint basis, or on a model basis, the error squared is the mean of the joint angle error squared.

$$\text{Eq. 31} \quad \text{ErrorSquared} = (\theta_{MA} - \theta_x)^2$$

$$\text{Eq. 32} \quad \text{RMSerror} = \sqrt{\frac{\sum \text{ErrorSquared}}{N}}$$

The error is reported several ways:

1. Dynamic error: the error squared for every instance of a trial.
2. Trial error: the RMS error of a single trial. This is equal to the square root of the sum of the dynamic error divided by the number of points in a trial.
3. Subject error: the RMS error for a specific subject. This is equal to the square root of the mean of trial error squared for all trials performed by the subject.
4. Task error: the RMS error for a specific task. This is the square root of the mean error of trial error squared for all trials associated with a specific task.
5. Global error: the RMS error for all tasks and subjects. This is equal to the root mean of the trial error squared for all trials.

The error squared was calculated in radians in each of the algorithm testing functions, and the trial, subject, task, and global RMS error on both joint and model basis were calculated and converted into degrees in the *CompileError.m* function, Appendix B.15.

5.3 Robustness of Methods

In addition to the accuracy of the selected methods, their robustness was also an important consideration. The robustness is the ability of the model to accurately predict the pose of an individual who was not part of the training data. The robustness was a significant part of the analyses because the purpose of the RHBM is to predict human motion to decrease the need for direct observation. To test the robustness of each method, subjects were excluded from the training set associated with each method. Data included in the training is referred to as the included data set and data that is excluded is referred to as the excluded data. The error is then calculated for all data. Initially only subject C01 was in the included data set, then subjects C02-C10 are transferred to the included set and the accuracy re-evaluated until all subjects' data have been added to the included data set. The data distribution for the robustness test number is illustrated in Table 22.

Table 22: Data distribution for robustness testing

		Robustness Test Number									
		1	2	3	4	5	6	7	8	9	10
Excluded Data	C01	C01	C01	C01	C01	C01	C01	C01	C01	C01	C01
	C02	C02	C02	C02	C02	C02	C02	C02	C02	C02	C02
	C03	C03	C03	C03	C03	C03	C03	C03	C03	C03	C03
	C04	C04	C04	C04	C04	C04	C04	C04	C04	C04	C04
	C05	C05	C05	C05	C05	C05	C05	C05	C05	C05	C05
	C06	C06	C06	C06	C06	C06	C06	C06	C06	C06	C06
	C07	C07	C07	C07	C07	C07	C07	C07	C07	C07	C07
	C08	C08	C08	C08	C08	C08	C08	C08	C08	C08	C08
	C09	C09	C09	C09	C09	C09	C09	C09	C09	C09	C09
	C10	C10	C10	C10	C10	C10	C10	C10	C10	C10	C10

The rate of convergence, calculated by the extrapolation of data onto a logarithmic function, of the included and excluded set error approximates the robustness of the method. All methods that are stable will eventually converge at a point where the addition of data from the excluded set to the included set has an insignificant impact on

the error associated with each set. However the number of subjects required to achieve convergence may be very large. In a robust method, the included and excluded subjects' average error will converge quickly. The error and number of subjects required was the primary consideration when evaluating the differences between methods and selecting the optimal method for this study.

5.4 Least Norm Solution (LN)

For this study the least norm solution, $\hat{\theta}_{LN}$, was used as a baseline to compare the performance of the various control algorithms and to serve as a reference for making qualitative assessments of motion. The least norm method uses the pseudo inverse of the Jacobian to find the mapping between end effector Cartesian velocity and joint angle velocity; this can be used to find an inverse kinematics solution by finding the difference between the forward kinematic solution and the desired end effector position.

For the RHBM, x was a 12 by 1 vector containing the Cartesian position and orientation of the right and left end effectors respectively, and θ represents the 1 by 25 joint angle vectors. The torso was represented by the first three joints of both the right and left arm models. The Jacobian is the mapping between the joint angle velocity, $\dot{\theta}$, and the end effector velocity and rotation in Cartesian space, \dot{x} . Composition of the bilateral Jacobian, J , from the Jacobians of the right and left arms, J_R and J_L respectively, and the forward kinematic equation is given in Eq. 33. The least norm solution, $\hat{\theta}_{LN}$, to inverse kinematics is given in Eq. 35, as described by the pseudo inverse of the Jacobian in Eq. 34. The first three joints of the right and left arm represent the movement of the torso and are shared

by both arms. The joint angles for joints 4-14 of the left and right arm are independent in the forward kinematic equation, but are dependent in the inverse kinematic solution.

$$\text{Eq. 33} \quad \dot{x} = \begin{bmatrix} \dot{x}_R \\ \dot{x}_L \end{bmatrix} = J\dot{\theta} = \begin{bmatrix} J_{R1-3} & J_{R4-14} & \mathbf{0} \\ J_{L1-3} & \mathbf{0} & J_{L4-14} \end{bmatrix} \begin{bmatrix} \dot{\theta}_{R\&L1-3} \\ \dot{\theta}_R \\ \dot{\theta}_L \end{bmatrix}$$

$$\text{Eq. 34} \quad J^+ = J^T(JJ^T)^{-1}$$

$$\text{Eq. 35} \quad \dot{\theta}_{LN} = J^+ \dot{x}$$

In this formulation both arms can move simultaneously but the movements of the arms are coupled. If the left hand moves and the right hand's position and orientation remains static, the joint angles of the right arm will have to change as well to accommodate the movement of the torso. Given a series of end effector positions and orientations, the corresponding joint angles were calculated by solving for each step in an iterative time series. Due to the non-linearity of the equations, error was introduced based on the size of the step between end effector trajectory points. In this application, this error was small due to the 20Hz effective frame rate and slow movement during the ADLs. However, error was prevented from accumulating by using the forward kinematics of the current position at each iteration when calculating the end effector difference. The formula for the iterative least norm solution is given in Eq. 36. Where θ_i is the current joint angle vector at iteration i , x_{i+1} is the desired end effector position and orientation, $fkine(\theta_i)$ is the current end effector position and orientation as determined by the forward kinematics of the RHBM, and θ_{i+1} is the joint angle vector correlating to the desired end effector position.

$$\text{Eq. 36} \quad \theta_{i+1} = \theta_i + J^+(x_{i+1} - fkine(\theta_i))$$

This method is referred to as the least norm solution because it produces the solution to the inverse kinematics that minimizes the norm of the joint angular velocity, Eq. 37.

$$\text{Eq. 37} \quad |\dot{\theta}|_{LN} = \sqrt{\dot{\theta}^T \dot{\theta}}$$

The function *testBiLN.m*, Appendix B.19, was used to test the least norm solution and calculate the error squared relative to the recorded joint angles from motion analysis.

5.5 Weighted Least Norm (WLN)

Based on the work by Chan and Dubey [73], the relative motion of joints can be penalized by adding a weighting term to the joint angle velocity norm, Eq. 38.

$$\text{Eq. 38} \quad |\dot{\theta}|_{WLN} = \sqrt{\dot{\theta}^T W \dot{\theta}}$$

Where W is a symmetric positive definite weighting matrix of size n by n , where n is the number of joints of the robot. For analysis, the weighted Jacobian and weighted joint angle velocity were defined as the following.

$$\text{Eq. 39} \quad J_W = J W^{-\frac{1}{2}} \quad \text{and} \quad \dot{\theta}_W = W^{\frac{1}{2}} \dot{\theta}$$

By substituting Eq. 39 into Eq. 33 and Eq. 38, the forward kinematics, Eq. 40, and weighted least norm, Eq. 41, equations can be verified.

$$\text{Eq. 40} \quad \dot{x} = J_W \dot{\theta}_W = J W^{-\frac{1}{2}} W^{\frac{1}{2}} \dot{\theta} = J \dot{\theta}$$

$$\text{Eq. 41} \quad |\dot{\theta}|_{WLN} = \sqrt{\dot{\theta}_W^T \dot{\theta}_W} = \sqrt{\dot{\theta}^T W \dot{\theta}}$$

The inverse of Eq. 40 can then be written as Eq. 42.

$$\text{Eq. 42} \quad \dot{\theta}_W = J_W^+ \dot{x}$$

The WLN solution can then be obtained by removing the weights from the angular velocity vector, Eq. 43.

Eq. 43
$$\dot{\theta}_{WLN} = W^{-\frac{1}{2}}\dot{\theta}_W = W^{-\frac{1}{2}}J_W^+\dot{x}$$

Eq. 43 can then be expanded through the definition of the pseudo inverse to become Eq. 44, resulting in the weighted least norm as a function of the inverse weights.

Eq. 44
$$\dot{\theta}_{WLN} = W^{-1}J^T[JW^{-1}J^T]\dot{x}$$

This can be used in an iterative manner similar to the least norm solution, as in Eq. 45.

Eq. 45
$$\theta_{i+1} = \theta_i + W^{-1}J^T[JW^{-1}J^T] * (x_{i+1} - \mathbf{fkine}(\theta_i))$$

For this study the weights were extracted from the motion analysis data so that they can be used to calculate the joint velocities in a simulation where they are unknown. Since there is no known closed form solution to directly calculate the weights, the first attempt to approximate the joint weight was to find the relative motion of each joint to the least norm solution of that joint for each instance in time Eq. 46.

Eq. 46
$$W^{-1} = \dot{\theta}_{MA}/\dot{\theta}_{LN}$$

However this method often produces a less desirable motion, likely due to the non-linearity, and interdependence of the weighted least norm solution. A linear change in weight has a non-linear change in joint angle, and changing the weight of one joint affects the change in joint angle of all joints. To determine the best set of joint weights the optimization toolkit in Matlab was used. The *fmincon* function is called to minimize the error of the weighted least norm solution by varying the values of, W^{-1} , which finds the appropriate weights to make the weighted least norm solution match the recorded joint angle velocity for every step in the trial. It is important to note that this method directly solves for the elements of the inverse of the weighting matrix on the range of 0.001 to 1, this is done to prevent the necessity of taking the extra step to invert the weighting matrix, W , in the optimization algorithm since it requires that the error

function be called many times. The weighting matrix is defined as a positive definite matrix [73], and the values are relative, so the lower bound of the inverse matrix must be greater than 0, and modification of the upper bound has a small impact on the results. The initial guess for the elements of the inverse of the weighting matrix were set to 0.5, which was chosen because it is the midpoint of the selected bounds, and the weights are all relative so any number can be used as the initial weight.

To evaluate the data completely the optimization was performed at several levels. First the weights were extracted at every point in the data series, and are referred to as the dynamic weights. The dynamic weights were evaluated on a constrained and an unconstrained basis. The constrained optimization added coefficients B and A to limit the rate of change of the joint weights, and the distance from the initial guess for the joint weights respectively Eq. 47. This decreases the variation of the extracted joint weights.

$$\text{Eq. 47 } Cost = \sum \left((\dot{\theta}_{WLN}(W) - \dot{\theta}_{MA})^2 + A * (W_{initial} - W)^6 + B * (W_{previous} - W)^2 \right)$$

The constrained and unconstrained dynamic weight optimization and testing was performed with *TestBiWLN_Dyn.m*, Appendix B.20, which allows for different weights at each instance of every trial. Then the weights were optimized for each trial, using one set of weights for each trial, producing the static weights using *TestBiWLN_Sta.m*, Appendix B.21. Weights were then optimized using one set of weights for each subject to form the subject weights, *TestBiWLN_Sub.m*, Appendix B.22, and then one set of weights for each task to form the task weights, *TestBiWLN_Tas.m*, Appendix B.23. Finally a single set of weights was extracted for all of the included data to form the global weights, *TestBiWLN_Glo.m*, Appendix B.24. The task and global weights use weights based on

the dominant hand, reordering the weighting matrix appropriately so weights 4-14 correspond to the dominant arm, and 15-25 to the non-dominant arm.

5.6 Probability Density Gradient Projection (GP)

The gradient projection method makes use of the null-space of the Jacobian to optimize the redundancy of the system. The pseudo-inverse of the Jacobian is defined in Eq. 34.

The joint angle velocity of the gradient projection method $\dot{\theta}_{GPM}$, is described in Eq. 48, and is a function of the Jacobian J , the end effectors' velocity \dot{x} , and the gradient vector ∇H . The gradient vector ∇H is described by the gradient of a function of the joint angles that should be minimized.

$$\text{Eq. 48} \quad \dot{\theta}_{GPM} = J^+ * \dot{x} + (I - J^+J) * \nabla H_{optim}$$

In this study, the performance was defined by the ability to reproduce the pose of the RHBM to match the pose of the subjects performing the recorded tasks. Therefore, the inverse of the joint angle density function, obtained from the motion data, was used to find the gradient vector as shown in Eq. 49. Here the gradient vector is formed by taking the partial derivative of the inverse of the joint angle density function for each of the joint angles. This method used the inverse of the probability density as the minimization function for the gradient projection method, where the probability density is the non-parametric density distribution as calculated by the Matlab function 'ksdensity.m'. The scalar quantity k was used to affect the rate of convergence of the solution on the inverse density function.

$$\text{Eq. 49} \quad \nabla H_i = k * \frac{d}{d\theta_i} (\text{Density}^{-1}(\theta_i))$$

To increase the accuracy of the solution, the joint angle data were divided into groups based on end effector position. The end effector space was divided into evenly spaced

increments, along the x, y, and z axes of the reference frame. This creates a number of discrete sets of data, equal to the cube of the number of increments along each axis, that were used to create the probability density distributions. The selection of the increment used was based on the position of the hands (end effectors) at each instance of the trial. The associated probability density distribution for that increment was then used to find the gradient vector. This accuracy of the probability density gradient projection was tested for increments from 1 to 20. Creation of the density function and testing of the algorithm was performed using *TestBiGP.m*, Appendix B.25.

5.7 Artificial Neural Network (NN)

An NN operates by performing a series of simple transfer functions on the weighted summation of a series of data. Each application of the transfer function is referred to as a neuron, and the neurons are arranged into layers. The output values of each layer become the inputs into each of the neurons in the next layer. In this study the NN was used to create a direct solution of the inverse kinematics given the control data set. One of the primary advantages of NNs is that they can easily be scaled based on the desired inputs and outputs. Simultaneous control of the left and right models can be achieved by simply including it in the training data and expanding the number of neurons to suit the additional data. The NN was implemented in Matlab using the neural network toolbox Version 7 [104]. The network consisted of a feed forward network with 18 input neurons, one hidden layer consisting of n neurons, and an output layer with 25 neurons, as shown in Figure 29.

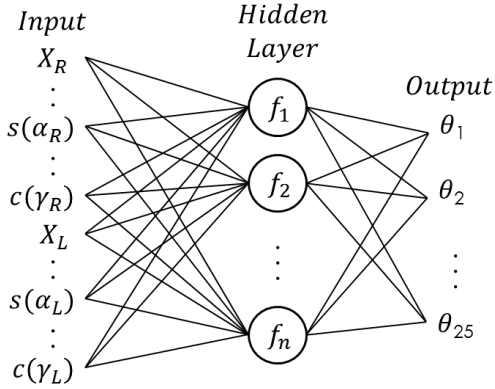


Figure 29: Neural network diagram

A sigmoid transfer function was used for the hidden layer and a linear transfer function was used for the output later. The inputs to the neural network consisted of the desired position and orientation of the end effectors, with the orientation broken down into the sine and cosine of each rotation. This was done to prevent singularities near π and $-\pi$. The output of the neural network was the joint angle vector of the upper body model, θ_{1-25} . The number of neurons in the hidden layer was varied from 10 to 100, in 10 neuron increments, for analysis of network performance as a function of size. Data from the ADLs recorded were used to train the neural network, using the Levenberg-Marquardt back propagation training function [104]. Training and testing of the neural network was performed in *TestBiNN.m*, Appendix B.26.

5.8 Combined Methods

This section evaluates combined methods that attempt to utilize the advantages of the different control schemes in an intelligent way to maximize the accuracy of the system. The initial investigation of methods identified the global WLN solution, the NN, and the GP, as the most potentially useful algorithms for this study. Therefore the combinations of the NN and WLN, and WLN and GP were selected for further study, since the WLN solution is the simplest algorithm to integrate with other methods.

5.8.1 Neural Network with Weighted Least Norm Correction (NN+WLN)

Since the NN algorithm is a numerically optimized prediction of the joint angle it does not guarantee a valid solution to the inverse kinematics, and the solution can be very jerky. To correct for this error the NN solution was smoothed using a weighted moving average filter. Then the WLN was used to correct end effector error, while minimizing the change in joint angles from the neural network solution. The process for correcting end effector error is similar to the WLN solution. The correction is performed by finding the difference between the forward kinematics of the NN solution and the desired end effector position, then multiplying the difference in position by the weighted pseudo inverse of the Jacobian, as shown in Eq. 50.

$$\text{Eq. 50} \quad \theta_{NN+WLN_i} = \theta_{NN_i} + W^{-1}J^T[JW^{-1}J^T] * (x_i - \text{fkine}(\theta_{NN_i}))$$

The robustness of this method was tested with 90 neurons in the hidden layer using *TestBiNN_WLN.m*, Appendix B.28. The large number of neurons was used because this solution was expected to increase the robustness, and decrease the accuracy of the solution.

5.8.2 Global Weighted Least Norm with Probability Density Correction (GP+WLN)

This method used a combination of the WLN solution with the GP method to maximize the probability density function, Eq. 51. This method gives us detailed control over the manipulation of the RHBM, as we can control the relative rate of each joint, as well as the optimum pose for static configurations.

$$\text{Eq. 51} \quad \dot{\theta}_{Final} = W^{-1}J^T[JW^{-1}J^T]\dot{x} + (I - J^+J)\nabla H$$

The discrimination of the workspace was set to 5 by 5 by 5 increments (*inc* = 5). Testing of the GP+WLN method was performed in *TestBiGP_WLN.m*, Appendix B.27.

Chapter 6: Motion Prediction Results and Analysis of Error

This section reviews the potential of different control methodologies for use in the control of the RHBM. The advantages and disadvantages of each method are discussed and analysis is described and presented. A brief summary of the primary methods investigated and their results are presented in Table 23 for reference. The detailed results of each method are given in the following sections. It is important to note that because each method is different the formulation of the results is presented in a different manner. The values given in Table 23 are based on the predicted performance of the associated method when the error for data included in and excluded from training are the same, as described in the robustness testing in Section 5.2.

Table 23: Brief summary of primary methods and results

Method	Sub-Method	Robustness	Predicted Convergence	RMS Error
LN	None	Perfect	N/A	11.1°
WLN	Global	Very High	3 subjects	8.0°
GP	Prob. Density (inc=19)	Very Low	Never	-
	Prob. Density (inc=10)	Very Low	162 subjects	6.6°
	Prob. Density (inc=5)	Moderate	28 subjects	7.5°
NN	Large (n=90)	Low	32 subjects	5.9°
	Medium (n=50)	Moderate	27 subjects	7.1°
	Small (n=30)	Moderate	28 subjects	8.2°

For many of the primary methods there are several sub-methods that are discussed later in this chapter, but are excluded from Table 23 for clarity. Significant differences were determined by analysis of variance and multiple comparison tests in Matlab using the *anovan.m* and *multcompare.m* function with a 95% confidence interval.

6.1 Analysis of Least Norm Solution Error

In this section selected quantitative aspects of the recorded ADLs were used to establish aspects of human motion control. Analyzing the movement of the distal joints relative to the least norm solution provides insight into the motivations behind human movement. To establish joints of interest the joints with the highest error were analyzed in detail. Data from the ten control subjects were used to find the error associated with the least norm solution. The least norm solution was used to find joints of potential interest for analysis and discussion. Table 24 and Table 25 show the RMS error on subject and task basis respectively. The error of the least norm solution is used as a baseline of comparison for the more complicated methods.

Table 24: Right arm RMS subject error for LN solution (degrees)

Subject	1	2	3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	Avg.	S.D.
C01	15	6	7	4	8	14	17	10	24	18	5	10	7	10	11	6
C02	17	6	4	7	11	10	9	11	16	19	7	15	15	12	11	5
C03	15	4	5	5	14	10	12	11	17	18	9	11	12	9	11	4
C04	19	5	7	8	16	16	14	14	21	16	14	16	15	9	13	5
C05	21	6	6	7	13	14	18	14	24	16	11	13	11	8	13	5
C06	14	4	5	8	7	11	11	6	14	21	7	13	8	12	10	5
C07	12	3	5	4	7	11	10	13	22	18	7	11	7	10	10	5
C08	18	7	4	4	16	10	19	12	23	17	8	15	13	6	12	6
C09	12	4	9	6	8	9	12	8	18	10	10	10	10	7	9	3
C10	17	4	7	7	6	9	18	10	22	27	15	15	11	8	13	7
Avg.	16	5	6	6	11	11	14	11	20	18	9	13	11	9	11	
S.D.	3	1	1	1	4	2	4	3	4	4	3	2	3	2		

The joints for the dominant arm with the highest RMS error are joints D9, followed by joints D10, 1, and D7, which represent upper arm rotation, elbow flexion, torso flexion, and upper arm flexion respectively. The brushing hair and opening a door ADLs had the highest error for the tasks, and subjects C04, C05 and C10 had the highest errors for subjects.

Table 25: Right arm RMS task error for LN solution (degrees)

Task	1	2	3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	Avg.	S.D.
Brush	25	6	7	6	14	13	14	16	26	25	10	15	15	13	15	7
Drink	10	3	6	4	7	8	13	7	18	7	6	5	5	6	8	4
Eat	8	4	4	4	7	7	12	6	11	9	5	9	9	6	7	2
Lift	17	5	6	6	13	11	16	10	22	18	9	9	10	7	11	5
Open	16	5	8	8	10	15	13	10	19	17	13	18	11	7	12	4
Avg.	15	5	6	5	10	11	14	10	19	15	9	11	10	8	11	
S.D.	7	1	1	2	3	3	2	4	6	7	3	5	4	3		

It is interesting to note that there is a greater variation in error between joints than between subjects or tasks, and a greater variation between tasks than subjects. In Subsections 6.1.1 through 6.1.5 joints with high error are investigated in detail.

6.1.1 Brushing Hair

Some movement of the torso was typically involved when picking up and putting down the brush, however, the majority of the movement for this task comes from the upper arm and forearm. In the least norm solution the proximal joints, torso flexion in particular, have an increased movement relative to the recorded joint angles, as shown in Figure 30.

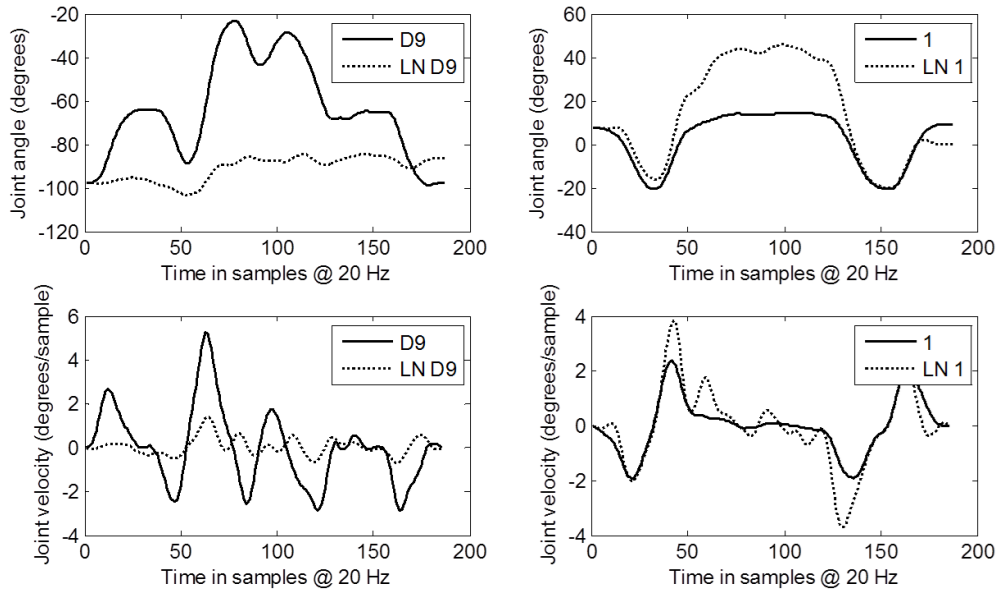


Figure 30: Upper arm rotation (left) and torso flexion (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz) for recorded data and least norm solution for brushing hair task, subject C04

The least norm solution results in greater movement of the torso, joint 1, and decreased movement of the shoulder, joint D9. The velocity profile of the least norm solution was similar to the recorded data; this shows the areas where the ability of the joint to perform the task movement is highly correlated with the recorded motion. There is also a considerable amount of noise in the recorded joint velocity, suggesting that additional filtering may be necessary if using joint velocity in a control algorithm.

6.1.2 Drinking From a Cup

In this task the cup must be raised to the mouth and be properly oriented. The cup must remain vertical while it was being raised to the mouth, and carefully controlled as the user drank (although in our recording the cup was empty so the control was potentially not as strict). Since the relative position of the mouth to the hand is independent of torso orientation there was, very little movement of the torso, as shown in Figure 31.

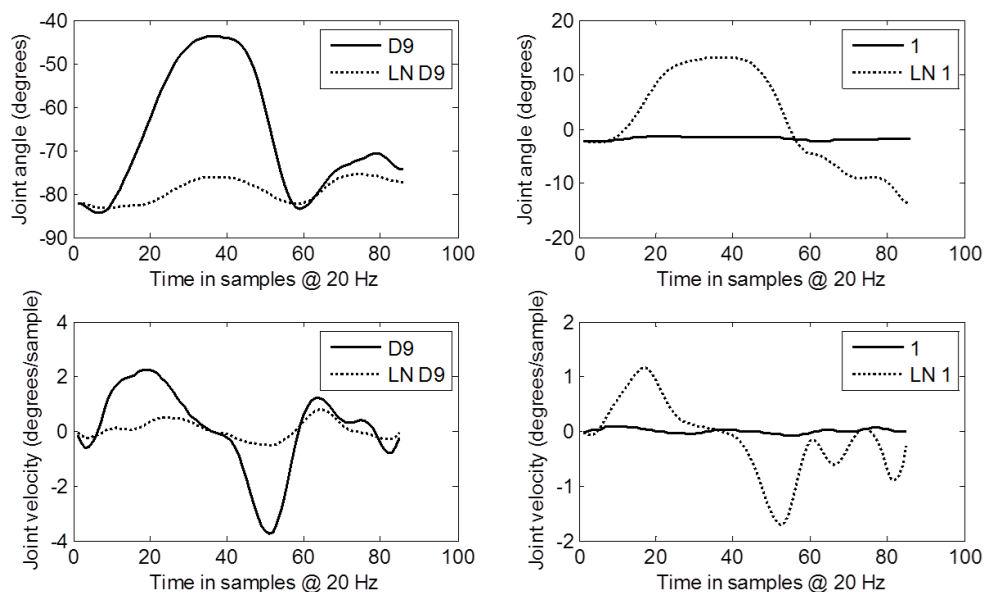


Figure 31: Upper arm rotation (left) and torso flexion (right) joint angles (top) and rotational velocity (bottom), recorded data and least norm solution, drinking task, subject C01

The difference between the least norm solution and the recorded data were even more evident in this task. The least norm solution had a large amount of movement in torso flexion, and very little movement in upper arm rotation.

6.1.3 Eating With a Knife and Fork

The eating task was performed from a seated position and was a bilateral task. It requires dexterous movement of the wrist for the positioning of the utensils. For this task joints D10 and D12, elbow flexion and forearm pronation, were investigated. The elbow flexion angle had a high angular velocity when the subject performed a cutting motion. The wrist has a few movements throughout the trial, an initial orientation, an orientation for cutting, and a peak where the food is brought to the mouth. The ability of the least norm solution to predict the proper motion can be seen in that the paths are similar, but there appears to be a difference in the magnitude of movement. This suggests that the weighted least norm solution may be sufficient to predict the motion of this task, at least for this subject.

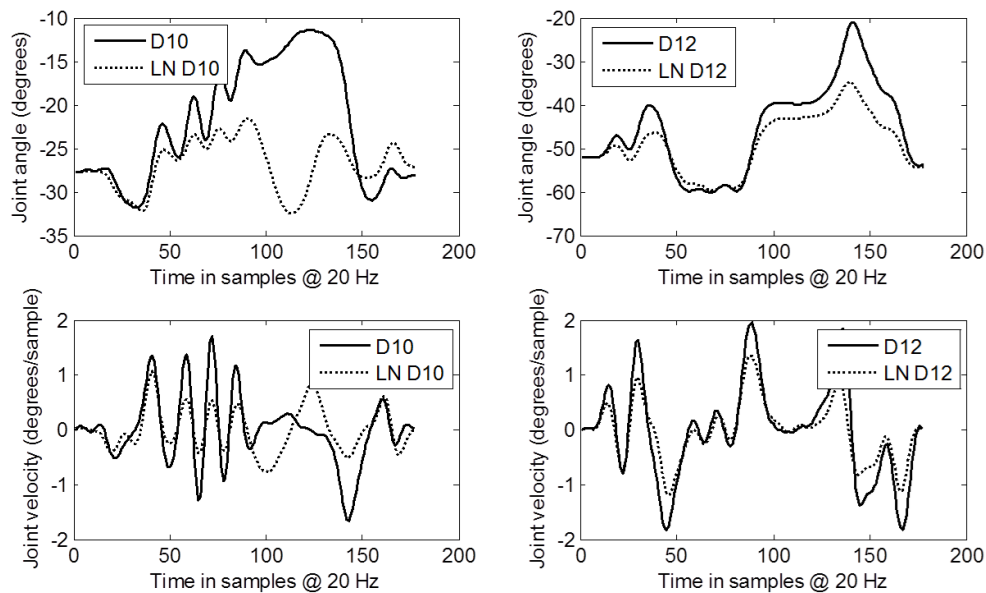


Figure 32: Elbow flexion (left) and forearm pronation (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, eating task, subject C05

6.1.4 Lifting a Laundry Basket

This task requires a large amount of movement in the torso, as well as the ability to lift a load. The task is nearly symmetric for the arms, so we see similar joint profiles in the right and left joints for the control subjects. The joints 1 and D7 representing flexion of the torso and upper arm respectively, were investigated for this task. In this case we see that the least norm solution is actually predicting a smaller range of motion in the torso and the upper arm than in the recorded data. This is likely due to the position of the joints at the start of the task, from a comfortable standing position the instantaneous velocity produce by torso flexion is primarily forward, where the desired path is for the hands to move downward towards the basket. The change of pose of the subject from one that is comfortable for normal standing, to one that better facilitates the performance of the tasks is likely the reason the least norm solution performs poorly for this task.

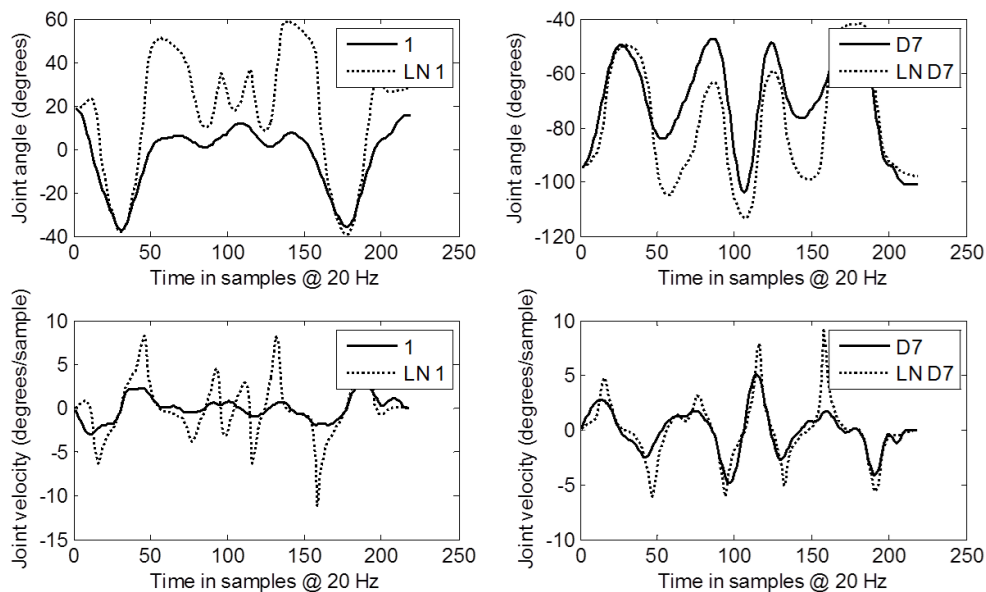


Figure 33: Torso flexion (left) and upper arm flexion (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, lifting task, subject C05

6.1.5 Opening a Door

This task requires dexterous manipulation at a location that is often on the edge of the workspace. The natural inclination is to stand sufficiently far away from the door to permit its opening without moving backward. This requires movement of the torso and upper arm to bring the hand to the knob, and the motion of the wrist and forearm to turn the knob and open the door. Joints 1 and D9, torso flexion and upper arm rotation, were investigated for this task. Similarly to brushing hair and drinking from a cup, there was increased movement of the torso for the least norm solution, and decreased movement of the upper arm.

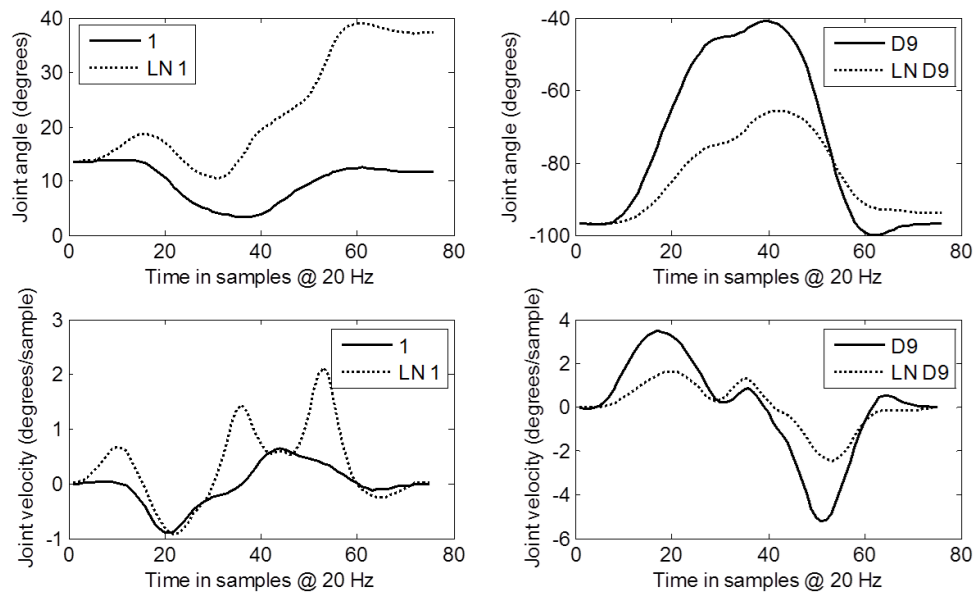


Figure 34: Torso flexion (left) and upper arm rotation (right) joint angles (rad) (top) and rotational velocity (rad/sample) (bottom) relative to time (sample 20Hz), recorded data and least norm solution, opening task, subject C02

From these analyses it is clear that the least norm solution is a poor predictor of human pose, but it does provide insight into the relation between the task and the joint movements required to complete them. While the raw position data used for this section

was filtered, it is clear that a noise remains in the joint angle data and that additional filtering may be required if the angular velocity is to be used in control algorithms.

6.2 Weighted Least Norm

The results of the motion reconstruction given the optimized weights on a subjects and task basis are given in Table 26 and Table 27 respectively. The error of the weight extraction methods were significantly different ($p < 0.05$) except the subject, task, and global weights. These results show the diminishing return of implementing more complicated functions for finding the joint weights.

Table 26: RMS error by subject for optimized weights (degrees)

Subject	Dynamic	Static	Subject	Task	Global	LN	Avg.	S.D.
C01	1.5	6.3	8.0	9.0	9.4	12.0	7.7	3.6
C02	1.0	5.2	6.3	6.4	7.0	10.8	6.1	3.2
C03	1.1	5.2	6.6	6.3	7.5	9.7	6.1	2.9
C04	1.2	6.8	8.7	8.2	9.7	12.7	7.9	3.8
C05	1.1	6.9	8.8	8.6	9.5	13.3	8.0	4.0
C06	1.2	4.1	5.7	5.4	6.1	8.9	5.2	2.5
C07	1.1	4.9	5.7	7.4	7.2	10.0	6.0	3.0
C08	1.5	6.6	8.6	8.5	9.5	11.5	7.7	3.5
C09	1.1	4.9	5.4	6.1	6.1	9.0	5.4	2.6
C10	1.5	5.5	9.0	7.3	8.4	11.5	7.2	3.4
Avg.	1.2	5.6	7.3	7.3	8.0	11.0	6.7	
S.D.	0.2	0.9	1.5	1.2	1.4	1.5		

There were also significant differences between subjects across the tested methods; subject C06 had the lowest average error which was significantly different ($p < 0.05$) from subjects C01, C04, C05, C08, and C10. Subject C05 had the highest error and was significantly different from subjects C02, C03, C06, C07, and C09. Between the tasks, the lowest error was found in the drinking task, which was significantly different from the brushing, lifting, and opening tasks. The lifting task had the highest error, and was significantly different ($p < 0.05$) from the drinking eating and opening tasks.

Table 27: RMS error by task for optimized weights (degrees)

Subject	Dynamic	Static	Subject	Task	Global	LN	Avg.	S.D.
Brush	1.2	7.0	8.4	8.9	9.4	14.4	8.2	4.3
Drink	1.1	3.0	4.4	4.9	5.0	7.1	4.3	2.0
Eat	1.1	3.9	5.9	5.0	6.2	8.1	5.0	2.4
Lift	1.6	8.0	9.8	10.3	11.1	12.8	8.9	3.9
Open	1.2	5.2	7.6	6.7	7.9	11.4	6.7	3.4
Avg.	1.2	5.7	7.5	7.4	8.2	11.1	6.9	
S.D.	0.2	2.1	2.1	2.4	2.4	3.1		

One of the contributing factors to the high error of the lifting tasks is likely the relatively large amount of movement of the torso for this task, which is not seen in the other tasks. The extracted values of the Dynamic weights in a time series exhibited very non-linear behavior, and showed little consistency between trials. In an attempt to reduce noise and increase repeatability of the dynamic control using WLN methods, a series of constraints were added to the optimization error function of the dynamic method. This attempt had similar issues to previous attempts in that the optimal values for the coefficients B and A were not consistent between trials and were highly sensitive. If the constraints were too high the results were inaccurate, if they were too low the results remained noisy. The dynamic WLN was therefore determined to be insufficient as a control algorithm for the model, since no function could be found to recreate the extracted values in a dynamic context. Additionally, the use of a neural network to approximate the solution of the joint weights as a function of the joint angles led to divergent solutions when more than one task was considered, presumably due to the inconsistency of the data.

Since the dynamic WLN method did not seem feasible, values of the static weights were investigated. In this method the same joint weights were used for an entire trial. The ratio of the joint movement relative to the least norm solution method was attempted again, only using the sum of the joint velocity or all points in the trial, but showed similar

results to the dynamic process. The static WLN solutions were still significantly more accurate than the task, subject, and global WLN solutions; however like the dynamic solution the results were not consistent between trials. This makes the implementation of the static weights difficult to implement, similar to the problems with the dynamic weights. Since no significant difference was found between the subject, task, and global weight errors, there was not sufficient reason to pursue the more complicated subject and task based weighting. The global weighted least norm solution showed significant improvement over the least norm method, and was selected for use in combination with the other methods.

6.2.1 WLN Robustness

The robustness of the WLN method was very high. The global weights from one subject work fairly well for all subjects, and the addition of more subjects to the training set had a relatively small effect on the error. This makes the global WLN method a promising method to use in conjunction with other control methods, as it provides a very consistent solution and variations can be assumed to be caused by the secondary method.

6.3 Probability Density Gradient Projection (GP)

Figure 36 shows the joint angle density distribution and the density function fit with the default settings of the Matlab '*ksdensity.m*' function. The inverse of the density function is used as the minimization function; hence the gradient vector is the derivative of inverse density function Eq. 49. The probability function serves partially as a joint limit function by restricting movement outside of observed joint angles. This helps to ensure that a stable solution is reached. The ranges of observed joint angles were always within theoretical anatomical joint limits for control subjects. Therefore, the probability density

function imposes a greater constraint on motion than a joint limit constraint would. An example of the joint angle density function is shown in Figure 35, and the associated inverse density and gradient function are shown in Figure 36.

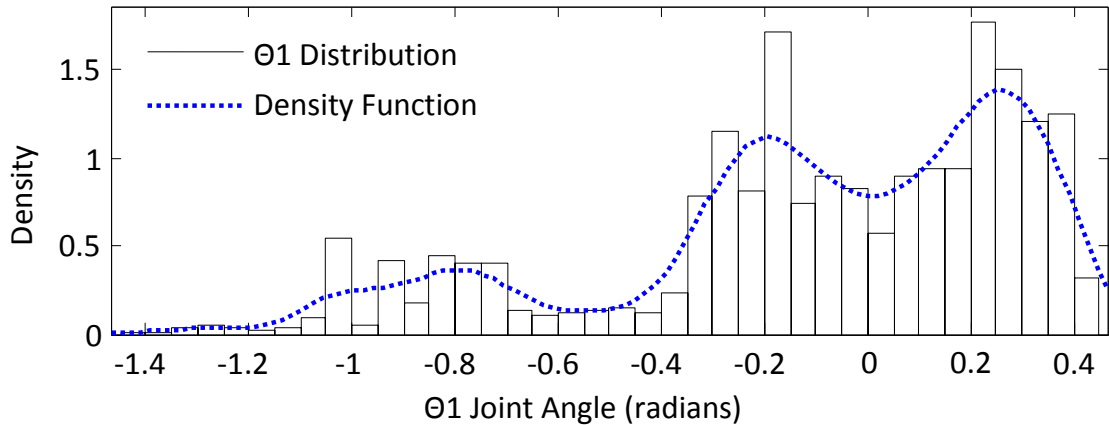


Figure 35: Density function for joint 1 (torso flexion)

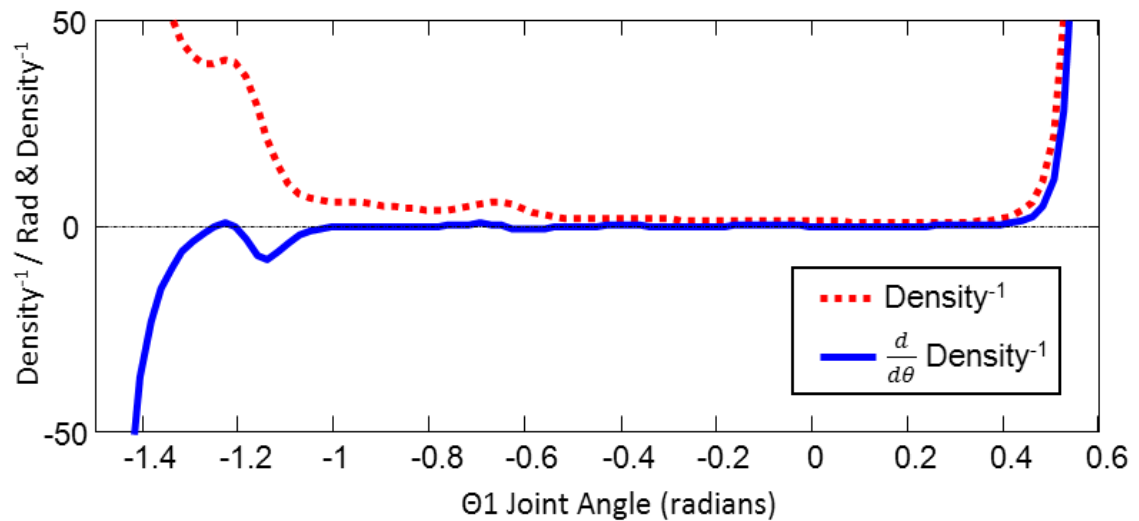


Figure 36: Inverse density and gradient function for joint 1 (torso flexion)

This method exhibits increasing accuracy as the division of the workspace increases, as shown in Figure 37. In the extreme case, this would end in each point of the workspace being assigned a specific joint angle distribution, if sufficient data were available.

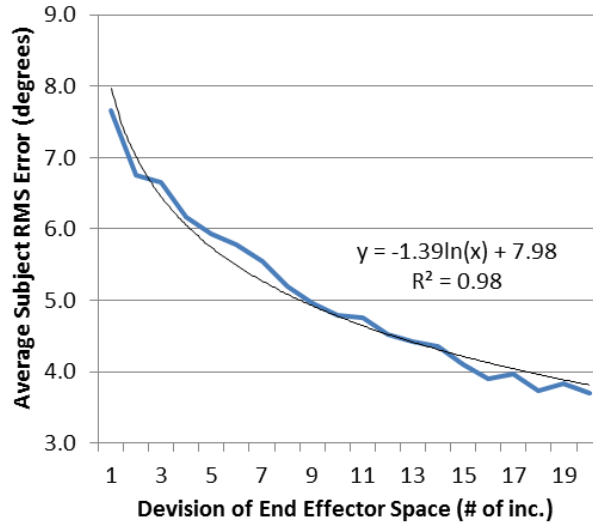


Figure 37: GP accuracy vs. division of end effector space

The impact of adding additional increments was greatest when the number of increments was low, and decreases as the number grows. With the limited data available for this study increasing the number of increments also increases the number of end effector sets where no data were available, in these positions, the GP method behaves the same as the least norm solution.

6.3.1 GP Robustness

The robustness of the GP method was very low for greater number of increments, the error for persons included in the trained data set was very low, while the error for persons in the excluded data set was high, and the addition of more data to included data set has little effect on the error of either set. As the number of increments decreases, the robustness of the GP increases. Using a logarithmic regression on each data set, $inc=19$ will likely never converge, $inc=10$ will likely converge with 162 subjects (at an estimated average error of 6.6°), and with $inc=5$ will likely converge with 23 subjects (at an estimated average error of 7.5°). The logarithmic fit was poor ($r^2 < 0.8$) for the included data sets, and good ($0.80 < r^2 < 0.99$) for the excluded GP data sets. The average RMS

joint angle error for each data set of the three increment levels included in the robustness tests are given in Figure 38.

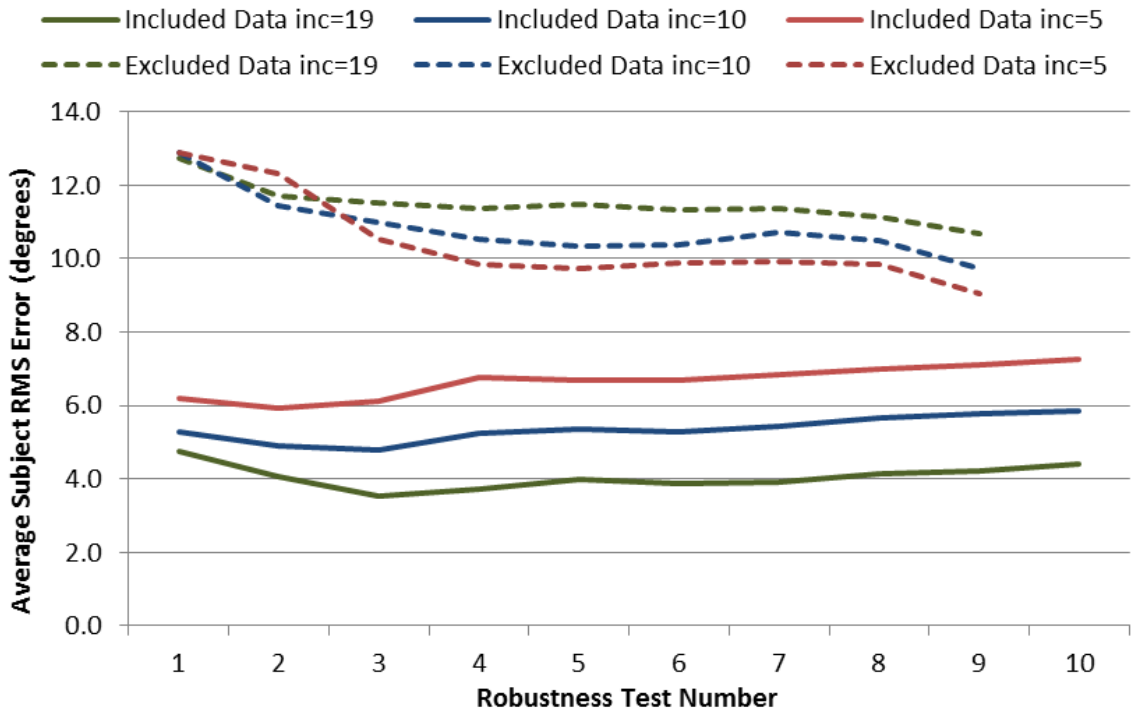


Figure 38: Robustness of the GP method

6.4 Neural Network

Increasing the number of neurons in the hidden layer decreases the RMS error, but increases the time and memory required to train the network. For this system, it was found that the decrease in error follows a logarithmic decay relative to increases in network size, as shown in Figure 39, for the range of networks tested. The neural network becomes more accurate in reconstructing joint angles than the least norm solution when the number of neurons was greater than 10.

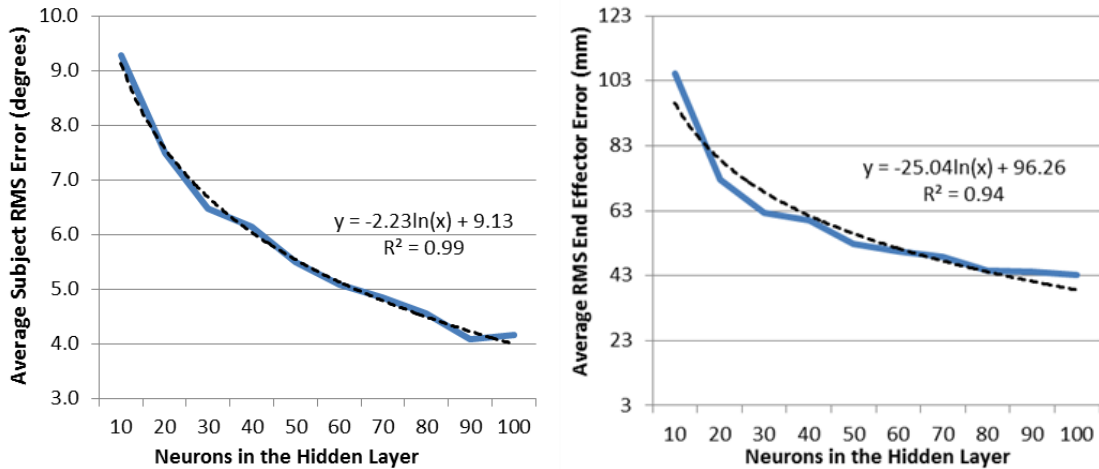


Figure 39: Effect of network size on bilateral NN performance

Since the NN was an approximation of the joint angles, the forward kinematics of the joint angles were not guaranteed to match the desired end effector position. The end effector error, Figure 39, showed a similar trend as the average joint angle error for changes in size of the hidden layer. The creation of task specific network was tested to determine if a significant increase in accuracy could be achieved. Specifying networks for each task did decrease the error of the trained subject data, but the error of the untrained subjects also increased. The error gains appear to be more likely due to the network having to fit to less data than to a relationship between the tasks and the joint angles.

6.4.1 NN Robustness

The robustness of the neural network was similar to the GP, except the addition of the initial subjects produced a drastic decrease in the error of the excluded set. Smaller network size shows higher robustness, however the change in size was more apparent in increasing error of the included data set than in decreasing error of the excluded set, as can be seen in Figure 40. Using a logarithmic regression on each data set, $n=90$ will likely converge with 32 subjects (at an estimated average error of 5.9°), $n=50$ will likely

converge with 27 subjects (at an estimated average error of 7.1°), and n=30 will likely converge with 28 subjects (at an estimated average error of 8.2°). The logarithmic fit was very good ($r^2 > 0.99$) for the NN included data sets, and poor ($r^2 < 0.80$) for the excluded data sets.

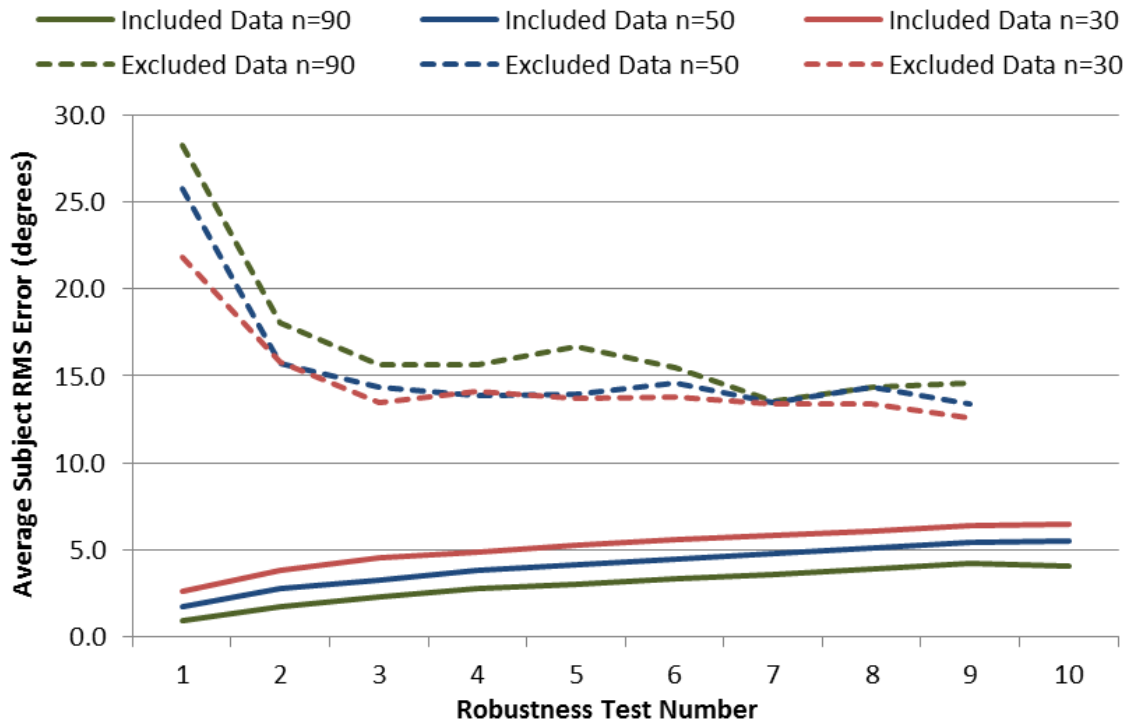


Figure 40: Robustness of the NN approximation

6.5 Neural Network with Weighted Least Norm Correction

No significant difference ($p < 0.05$) was found between the subject RMS joint angle error of the NN and NN+WLN methods, significant difference was found between the end effector error of the corrected and uncorrected data. The end effector error of the WLN+NN was set to be less than 0.01 mm. Despite the initial hypothesis that the WLN correction would lower the accuracy of the NN method; the NN+WLN actually had a slightly increased accuracy. The predicted convergence occurred at an error of 4.1° with

19 subjects, although this convergence was skewed by the very high initial error of the excluded data set.

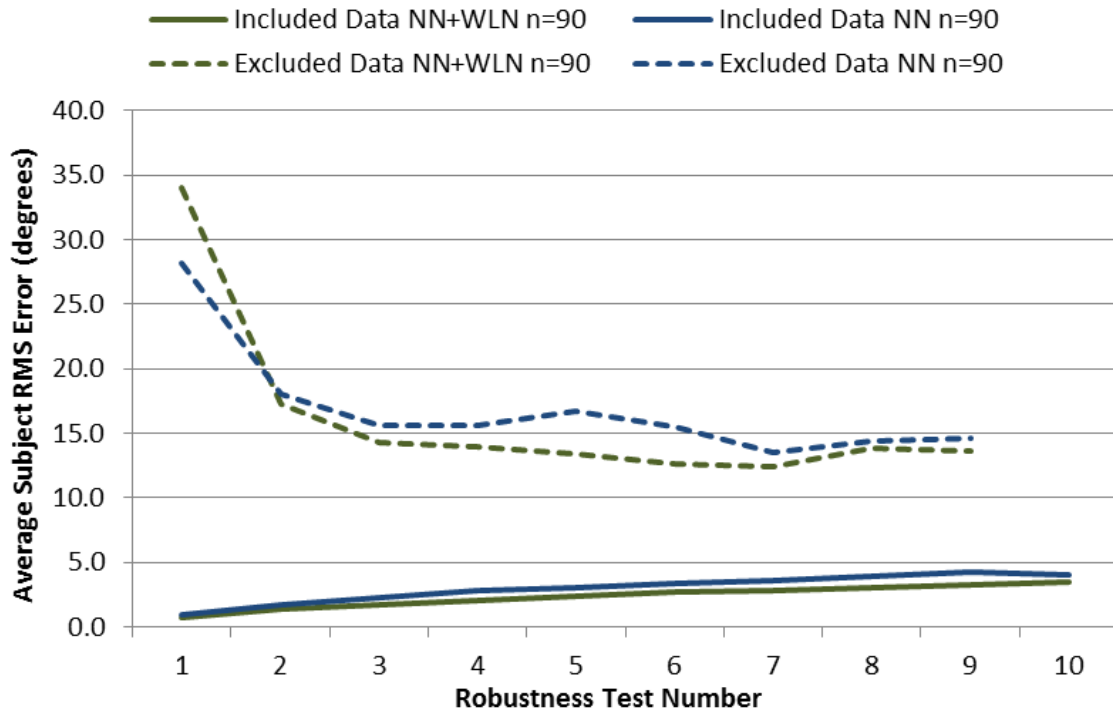


Figure 41: Robustness of NN+WLN method

6.6 Global Weighted Least Norm with Probability Density Correction

Unfortunately the effect of adding the weighted least norm solution to probability gradient vector solution did not drastically reduce the error for subjects in the excluded data set. The robustness of the final method is RMS error for all of the control subjects did increase, however the error of the untrained data only reaches the same level as the WLN solution when 9 of the 10 subjects are in the included data set. This means that when less than 9 subjects were in the included data set that the probability density correction is decreasing the accuracy of the excluded data set relative to the WLN solution. The robustness results for the GP+WLN method is shown in Figure 42. The significant drop in included data error led to a likely convergence of 5°, however since

the decrease in excluded data error was not as good the robustness of the solution actually decreased to a predicted convergence with 160 subjects.

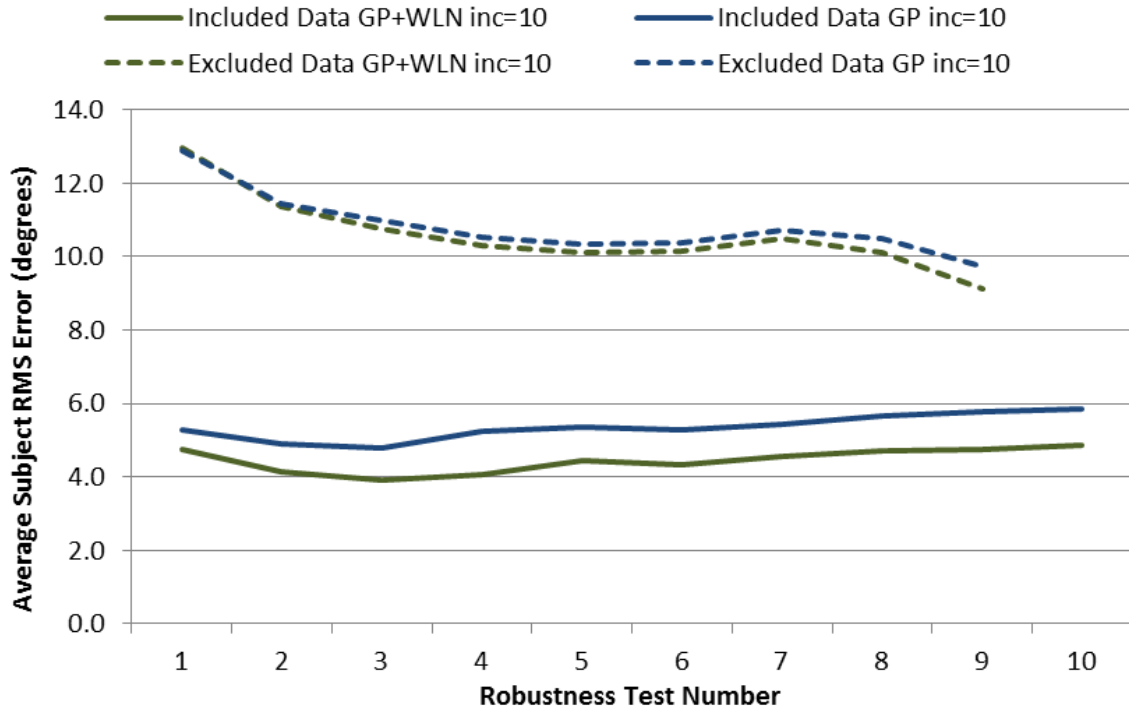


Figure 42: Robustness of the GP+WLN method

6.7 Braced Subject Testing

The next test of each method’s capability was their ability to predict the motions of persons wearing the arm brace. This was the gate way to accurately predicting the motion of amputee subjects and therefore serves as the second screen in selecting the appropriate control method. For this section the impact of the brace on the error of the global weighted least norm, the neural network method, and the probability distribution method was evaluated. The effect of the brace on the weighted least norm was evaluated by extracting the global WLN for the braced subjects and comparing the results to the global WLN of the un-braced subjects. The neural network method was tested by adding an input parameter indicating that the subject was or was not wearing the brace. The probability method was tested by adding the braced joint limits to the gradient function.

6.7.1 Braced Weighted Least Norm Testing

For this test the global weights for the braced subjects were extracted and compared to the global weights of the un-braced subjects. The weights were optimized with the braced and un-braced subject data for: all control and braced subjects, just control subjects, just braced subject, the first half control and braced subjects, and the second half of the control and braced subjects. Subject B07 had a significantly higher error and was excluded from the data sets.

Table 28: WLN RMS subject error for braced and un-braced subjects (degrees)

	All	Control	Braced	First	Second	Avg.
C01	9	9	9	9	8	9
C02	7	7	8	7	7	7
C03	7	7	7	7	8	7
C04	9	9	9	9	10	9
C05	9	9	10	9	9	9
C06	6	6	6	6	5	6
C07	7	7	7	7	6	7
C08	9	9	9	9	9	9
C09	6	6	6	6	6	6
C10	8	8	8	8	8	8
B01	12	12	12	12	12	12
B02	8	8	8	8	8	8
B03	8	7	8	8	7	8
B04	8	7	7	7	7	8
B05	7	8	8	8	8	7
B06	8	8	8	8	8	8
B08	7	7	7	7	7	7
B09	6	6	6	6	5	6
B10	6	6	6	6	6	6
Control Avg.	8	8	8	8	8	8
Braced Avg.	8	8	8	8	8	8
Total Avg.	8	8	8	8	8	8

*Error from trained data shown in **bold**.

A significant difference ($p < 0.05$) was found between the RMS joint angle error for the control and braced subjects when the global weights were optimized for the braced subjects only. No significant difference was found for all other cases. For the braced

subjects the inverse weights are reduced for the distal limb causing a restricted motion.

The control and braced inverse weights for the right (R) arm are given in Table 29.

Table 29: Global control and braced inverse weights for the dominant arm

Joint	1	2	3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14
Control	0.07	0.04	0.06	0.18	0.18	0.06	0.31	0.10	0.32	0.52	0.00	0.99	0.28	0.34
Braced	0.09	0.03	0.06	0.18	0.20	0.12	0.46	0.11	0.98	0.39	0.06	0.18	0.07	0.02

6.7.2 Braced Neural Network Testing

For this test the NN method was modified to include an additional input, which was a Boolean identifier of braced verses un-braced condition, braced = 1, un-braced = 0. The small sized NN, n=30 was used to train to the network using data using the same test sets as described for the WLN method in the previous subsection.

Table 30: NN RMS subject error for braced and un-braced subjects (degrees)

	All	Control	Braced	First	Second	Avg.
C01	14	9	18	14	19	15
C02	13	7	18	13	20	14
C03	15	15	19	15	24	18
C04	12	6	15	13	17	13
C05	10	6	13	10	15	11
C06	10	7	13	14	11	11
C07	15	7	20	17	16	15
C08	12	6	14	14	11	11
C09	14	6	14	16	11	12
C10	14	6	16	16	15	14
B01	9	18	8	7	23	13
B02	8	15	6	6	18	11
B03	8	15	6	7	16	10
B04	7	13	6	5	13	9
B05	7	13	6	6	16	9
B06	8	15	6	18	5	10
B08	8	12	7	10	6	8
B09	8	15	7	17	5	10
B10	7	18	6	18	5	11
Control Avg.	13	7	16	14	16	13
Braced Avg.	8	15	6	10	12	10
Total Avg.	10	11	12	12	14	12

*Error from trained data shown in **bold**

As seen previously the neural network performance drastically decreases for data not included in the training set. However we do see that the Boolean brace input is somewhat effective in controlling the output of the network, the error of the First and Second training sets is not significantly worse than that of the Control and Braced training sets. However the error of the control subjects is higher than the braced subject error. Since the robustness of the neural network is not good we will not use it for direct inverse kinematics, but it may be implemented at a later date to control additional parameters.

6.7.3 Braced Probability Density Gradient Projection Testing

The range of motion on the braced subject was used to add additional constraint to the gradient vector of the probability density gradient function by removing joint angle vectors that exceeded the braced subjects' RoM. This was done because the joint limits of the braced subjects lies within the constraints imposed by the probability density function of the control subjects. This acts as an additional constraint, preventing the braced limb from exceeding its braced limits. The joint limit performance criteria defined by Chan [73] was used in instances where no training data within the subjects range of motion was available in a given increment of end effector space. The joint limit function is given in Eq. 52, and its gradient is given in Eq. 53. The weighting factor of the joint limit, k_{JL} , was set to 0.05. In the condition that the joint angle becomes greater than the maximum joint limit the gradient value for that joint is set to the maximum gradient value, 0.1, and if it is below the minimum the negative of the maximum value is used.

$$\text{Eq. 52} \quad H_{JL}(\theta_i) = k_{JL} \frac{(\theta_{imax} - \theta_{imin})^2}{4(\theta_{imax} - \theta_i)(\theta_i - \theta_{imin})}$$

$$\text{Eq. 53} \quad \nabla H_{JL}(\theta_i) = k_{JL} \frac{(\theta_{imax} - \theta_{imin})^2 (2\theta_i - \theta_{imax} - \theta_{imin})}{4(\theta_{imax} - \theta_i)^2 (\theta_i - \theta_{imin})^2}$$

The data included for creating the probability density was varied similarly to those in the WLN and neural network implementations in the previous subsections. The RMS error for the probability density model is given in Table 31.

Table 31: GP RMS subject error for braced and un-braced subjects (degrees)

	All	Control	Braced	First	Second	Avg.
C01	7	6	13	7	12	9
C02	5	5	10	5	9	7
C03	5	5	10	5	11	7
C04	7	7	11	7	12	9
C05	6	6	10	6	11	8
C06	5	5	11	11	4	7
C07	5	5	13	10	5	7
C08	7	6	10	9	6	7
C09	4	4	9	9	4	6
C10	5	5	9	10	5	7
B01	7	15	7	7	19	11
B02	6	11	6	6	10	8
B03	5	10	4	5	10	7
B04	5	12	4	5	12	8
B05	5	10	5	5	12	7
B06	4	15	4	6	4	7
B08	5	8	5	9	5	7
B09	4	7	4	10	4	6
B10	5	13	5	15	5	9
Control Avg.	6	5	10	8	8	7
Braced Avg.	5	11	5	8	9	8
Total Avg.	5	8	8	8	8	8

*Error from trained data shown in **bold**

The probability density method has shown the best results and is significantly better ($p < 0.05$) than the WLN and neural network braced implementations. It also shows relatively good results even when the brace data were not included in the density function generation, which suggests that this is a reasonable control scheme for adaption of dissimilar subject data.

6.8 Analysis of Distribution of Error

To better understand the sources of error associated with each method the error as a function of joint angle, and the relative performance of each method for each task was evaluated.

6.8.1 Joint Angle Distribution of Error

Analysis of the joint angle error of the components of WLN solution, GP, and NN method shows that in fact the error of the solutions relative to the joint numbers are similar. The normalized distribution of error was calculated by dividing the global joint RMS error by the average of the joint RMS error for each method trained with 5 subjects in the included data set and five subjects in the excluded data set. The results are shown in Figure 43.

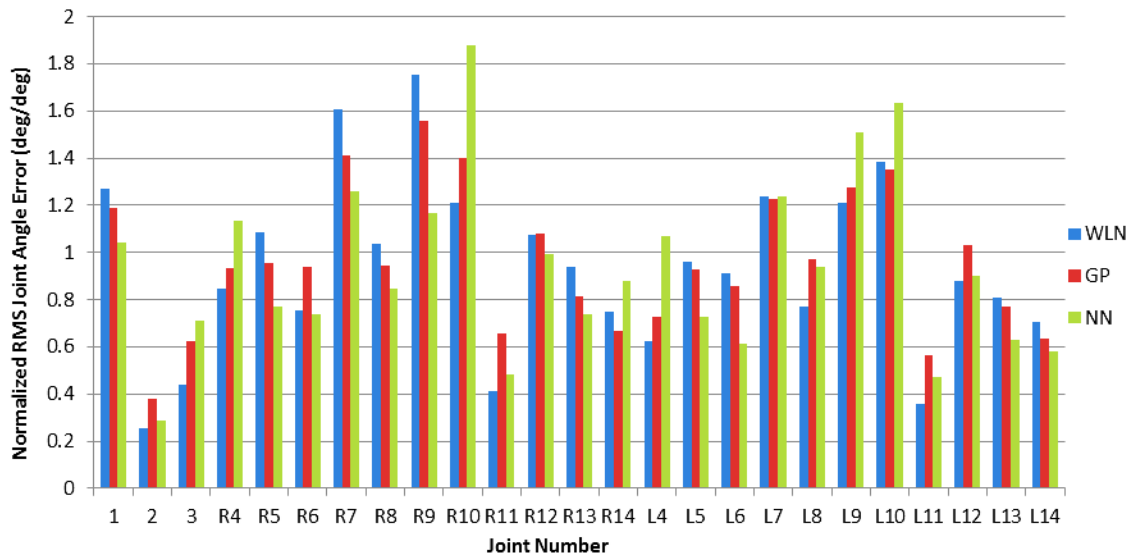


Figure 43: RMS error of each joint, C01-C05 included, C06-C10 excluded

6.8.2 Task Based Comparison of Methods

This section compares the performance of selected methods on a task basis. The values used for the NN, NN+WLN, GP, and GP+WLN are the results of the convergence of the robustness test for each task. Since the robustness of the LN and WLN methods is very

high, the LN has no training, and the effect of adding additional subjects to the WLN solution was insignificant, the a values for LN and WLN were simply the task RMS error. The error for each method is given in Table 32.

Table 32: Comparison of methods task RMS error (degrees)

Task	LN	WLN	NN	NN+WLN	GP	GP+WLN
Brush	14.4	9.5	6.3	3.4	9.5	5.8
Drink	7.1	4.9	5.3	4.0	6.3	3.3
Eat	8.1	6.1	6.6	3.8	8.0	4.2
Lift	12.8	11.0	7.3	5.2	7.7	5.0
Open	11.4	8.0	7.0	4.3	8.3	6.6
Avg.	10.7	7.9	6.5	4.1	8.0	5.0
S.D.	3.1	2.5	0.8	0.7	1.2	1.3

Based on the error at convergence for the tasks, the NN+WLN had the best performance in all tasks except the drinking from a cup, where the GP+WLN method was more accurate. The predicted numbers of subjects required for convergence are given in Table 33. The NN+WLN also had a relatively low convergence numbers for all tasks except drinking from a cup. The GP+WLN had very high convergence numbers.

Table 33: Predicted number of subjects for convergence

	NN	NN+WLN	GP	GP+WLN
Brush	36	17	71	116
Drink	36	51	12	56
Eat	32	15	40	871
Lift	16	17	26	686
Open	23	20	23	94
Avg.	29	24	34	365
S.D.	9	15	23	384

The accuracy and robustness of the NN+WLN method may be exaggerated by the high error of the untrained data set in the first step of the training.

Chapter 7: Discussion and Future Work

The RHBM is designed to fit into a larger simulation tool for the prediction of prostheses outcomes. The operation of the simulation calls the RHBM with the input structures *prosthesis* and *task*. The *task* structure contains end effector position, rotation, and force constraints. The *prosthesis* structure contains the coefficients of the prosthetic constraints, which affect the joint parameters of the joints of the model, as well as the interface constraints which characterize the socket / residual limb interface. In the simulation the user selects which module they want to use and inputs the desired subject parameters, the module then performs an iterative analysis, finding the performance of the subject in simulation given a variety prostheses and task constrains. The flow of data in the simulation is given in Figure 44.

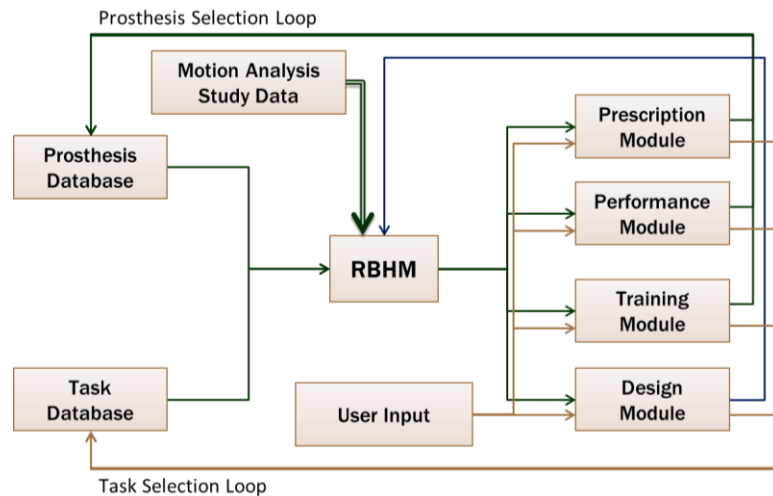


Figure 44: Diagram of upper body prosthesis simulation tool

The RHBM operates by opening files created by the ‘Motion Analysis Study Data’, which is comprised of the upper body model created from the methods described in

Chapter 3:, and the control algorithm from the methods described in 4.1. The upper body model is created by the *CreateUBM.m* script, and the neural network is created and trained by the *TrainNN.m* script.

The simulation itself exist to predict the motion of the upper body and the prosthesis, this allows for the prediction of various outcome measures, such as compensatory motion, prior to the patient ever donning a device. This enables the prosthetist to better select prosthetic components, a designer to make faster design iterations, a therapist to select effective strategies, or a researcher to establish areas of interest for further study.

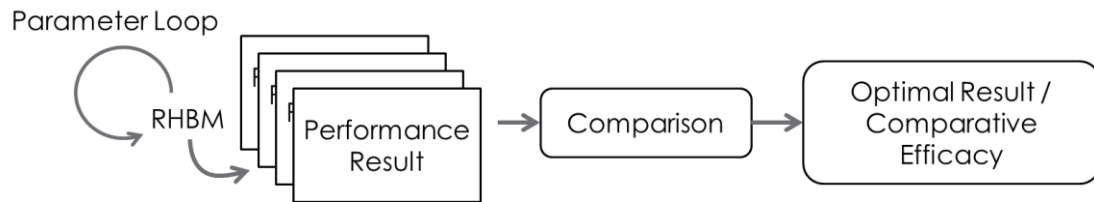


Figure 45: Diagram of simulation function

This simulation could be used in conjunction with a simulation similar to those developed by Lamounier et al. [105] and Hauschild et al. [60], which enable the user to interact in a virtual environment given a virtual prosthesis. Currently the RHBM is suitable for predicting the motion of healthy individuals with minimal error. However there has been insufficient data to confidently support the accuracy of the model in predicting the motions of persons using a variety of prosthetic devices.

7.1 Discussion

Human movement is a complicated function. The cerebellum coordinates movement and balance, but is controlled by our intentions and our capabilities within the environment. Generalized prediction of upper body motion remains a topic with plenty of room for improvements. This work has led to significant achievements in the accuracy of upper

body tracking and motion reconstruction using marker based optical tracking.

Additionally the control methods investigated provides an accurate prediction of upper body inverse kinematics for a general workspace. The basis for this research started as an extension from applied research in biomechanics, and it has followed a path leading to significant findings in basic research. As the results of this study are incorporated into applied research the benefits will become clear and new paths for future research will open.

Table 34: Review of tested methods

Method	Pros	Cons
Global Weighted Least Norm (WLN)	<ul style="list-style-type: none"> • Scalable • Easy to implement • Inherent model knowledge • Easily combined with other Jacobian based methods 	<ul style="list-style-type: none"> • Requires singularity avoidance / compensation • Only effects in-motion action (velocity mapping) • High error relative to other methods
Probability Density Gradient Projection (GP)	<ul style="list-style-type: none"> • Potential to add additional constraints • Easy to combine with other methods • Qualitative meaning • Stable with the inclusion of braced subject data. 	<ul style="list-style-type: none"> • Incrementation of workspace can require a lot of memory • Has a lot of parameters that have to be tuned
Gradient Projection + Weighted Least Norm (GP+WLN)	<ul style="list-style-type: none"> • Improved accuracy for subjects included in the training set • Increased number of potential control variables 	<ul style="list-style-type: none"> • Decreased robustness • Increased the complexity of solution
Artificial Neural Network (NN)	<ul style="list-style-type: none"> • Scalable • Easy to implement • Direct inverse kinematics. 	<ul style="list-style-type: none"> • No Inherent model knowledge • Poor robustness • Not stable with the inclusion of braced subject data
Neural Network + Weighted Least Norm (NN + WLN)	<ul style="list-style-type: none"> • Increased accuracy • Removed end effector error from solution 	<ul style="list-style-type: none"> • Low reliability of robustness projection • High error for subjects excluded from the training set

7.1.1 Contributions to the State of the Science

This work has made several contributions to areas of basic and applied research.

1. Significant differences were found between the range of motion (RoM) and task performance of persons wearing braces and control subjects. Compensatory motions were observed for braced subjects and amputees. This contributes to the general knowledge of the impact of these devices on everyday activities.
2. A database of subjects' poses for the upper body during activities of daily living using the collected subject data was created.
3. A functional joint center method for determining subject specific parameters was created and used to make a highly accurate model of the upper body.
4. The least norm (LN), weighted least norm (WLN), probability density gradient projection (GP), and artificial neural network (NN) methods for inverse kinematic control of the robotic human body model (RHBM) were evaluated, and a combination of the probability density gradient projection and weighted least norm (GP+WLN) algorithms was selected for use in predicting the motion of human subjects.

The application of this work could be implemented in a variety of fields that use a model of the upper body. The RHBM can be used in studies that require accurate kinematic data, such as biomechanics and sports related studies. The inverse kinematic algorithms could be used to increase the realism of computer animations by ensuring that the upper body inverse kinematics produced realistic poses, enabling the animator to specify the position and orientation of the hand only.

7.1.2 Significance of Errors

The reported error in this work is defined by the RMS of the difference between the predictive models and the recorded motion analysis data as described in Section 5.2. However the clinical significance of the model error is difficult to interpret, clearly the 11° error of the least norm solution is worse than the 6° error of the weighted least norm with null space probability density correction. However the distribution and the source of the error were also important in evaluating the performance of the algorithm. For instance the predicted joint velocity of the large (90 neurons in the hidden layer) neural network was very jerky, which lowers its clinical acceptability but decreases error as it was defined in the study. Additionally the jerky solution of the NN method is an attempt by the training algorithm to compensate for variations between subjects and tasks. This suggests that the ideal solution will still have some error as described by this study, and that the best solution may have a greater error than a less acceptable solution with lower error.

In regard to the magnitude of the error, and the significance of that magnitude, it is important to note that even the conventional differences between the Plug-in Gait and the functional joint center segment kinematics had an average joint angle difference of 6° , this variation due to conventional difference has also been noted in the literature [61]. Standard deviation of clinical measurements using goniometry has been cited as 3.8° using clear plastic goniometers, and 4.2° using steel goniometers [106]. A recent article set the limit of agreement of 10° for acceptance of visual estimation as a reliable method to assess range of motion of elbow flexion [107]. Since these studies have evaluated single axis rotations, which typically are easier to measure than multi-axis rotation, it is

reasonable to assume that the global RMS error of 6° for the RHBM will be clinically acceptable.

7.1.3 Limitations

This study has several limitations that could be addressed in future work to increase the accuracy of the results.

1. The limited number of amputee subjects has prevented a thorough analysis of amputee motion, and implementation in the inverse kinematics algorithms. The ability of the model to behave appropriately should be investigated after additional amputee subject data has been collection.
2. Joint center locations were primarily verified by the accuracy of the model in reconstructing subject motion, which they did very well. However, cross-validation of the joint center locations with radiographic imaging could be performed to further validate the methods.
3. The accuracy of the motion prediction algorithm was near clinically acceptable levels for measurement. However the inter-subject variance in joint angles given similar tasks was not analyzed, advanced statistically methods could be used to sort task variances from subject variances to determine the true subject variance.
4. Further development of control method to include more subject parameters could improve the accuracy of the motion prediction algorithm. Additional subject data would most likely be required to ensure the robustness of the algorithms given the additional parameter space.

7.2 Future Work

The further development of the RHBM and associated prosthesis simulation tool are ongoing efforts at the University of South Florida, Center for Assistive Rehabilitation & Robotics Technologies. This section gives a brief outline of topics of planned research related to this work.

7.2.1 Integration and Verification with Additional Amputee Subject Data

An ongoing study focused on the further development of the simulation tool for upper extremity prostheses, and general analysis of prosthesis use during ADLs will lead to a greater subject database to train and test the capabilities of the RHBM in predicting the movement of prosthesis users. With this additional data it is the goal of the research team to develop a tool that will assist amputees, prosthetists, physical and occupational therapists, and designers optimize the efficacy of prosthetic devices.

7.2.2 Dynamic Analysis

The addition of force constraints in the simulation can be executed by determining the mass and inertia properties of the upper body segments based on subject measurements and literature data, such as those found by Veeger et al. [108]. This can then be combined with segment accelerations from the RHBM and external forces as dictated by the task to affect kinetic outcomes, such as the deformation of soft tissues and slippage at the residual limb / socket interface.

7.2.3 Residual Limb Interface

In addition to kinematic prediction the simulation will incorporate a dynamic model of the residual limb interface. This model will use forces acting on the prosthetic system to

calculate the force at the residual limb and then find the resulting rotation at the residual limb interface. The model will be developed by applying similar method as Sensinger and Weir [109], but will use a non-finite element model of the interface that is trained on data collected from a diverse sample of amputee subjects.

7.2.4 3D Visualization

A 3D human body model has been adapted for use with the RHBM. The model's skin can be swapped to allow for different visual representations of the subject, allowing the visuals to represent subjects of different age, skin color, build, and other aesthetic factors. However, all models will function from the parameters as defined by the RHBM, and the function of the simulation will not be affected by the display. The visualization was coded in the Microsoft XNA framework and is compatible with windows PC with direct X. Future work will focus on increasing the quality of anatomical visuals, and integration of visual feedback in the user interface.

7.2.5 Graphical User Interface

To facilitate the clinical use of the simulation a graphical user interface (GUI) is being developed to perform the iterative analysis of the simulation tool automatically, without knowledge of the underlying functions and code. This program should also implement an automated system to interpret quantitate model data into clinically relevant results. This is a necessary step in the development from research to a clinically applicable, and beneficial, tool.

References

1. Biddiss, E. and T. Chau, *Upper limb prosthesis use and abandonment: A survey of the last 25 years*. Prosthetics and Orthotics International, 2007. 31(3): p. 236-257.
2. Lindner, H., B. Nätterlund, and L. Hermansson, *Upper limb prosthetic outcome measures: Review and content comparison based on International Classification of Functioning, Disability and Health*. Prosthetics and Orthotics International, 2010. 34(2): p. 109-128.
3. Wright, V., *Prosthetic outcome measures for use with upper limb amputees: A systematic review of the peer-reviewed literature, 1970 to 2009*. JPO: Journal of Prosthetics and Orthotics, 2009. 21(9): p. P3.
4. Hermansson, L., et al., *Assessment of capacity for myoelectric control: a new Rasch-built measure of prosthetic hand control*. Journal of Rehabilitation Medicine, 2005. 37(3): p. 166-171.
5. Burger, H., et al., *Validation of the orthotics and prosthetics user survey upper extremity functional status module in people with unilateral upper limb amputation*. Journal of Rehabilitation Medicine, 2008. 40(5): p. 393-399.
6. Desmond, D. and M. MacLachlan, *Factor structure of the Trinity Amputation and Prosthesis Experience Scales (TAPES) with individuals with acquired upper limb amputations*. American Journal of Physical Medicine & Rehabilitation, 2005. 84(7): p. 506.
7. Demircan, E., et al., *Human motion reconstruction by direct control of marker trajectories*. Advances in Robot Kinematics: Analysis and Design, 2008: p. 263-272.
8. Wu, G., et al., *ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion--Part II: shoulder, elbow, wrist and hand*. Journal of Biomechanics, 2005. 38(5): p. 981-992.
9. Wu, G., et al., *ISB recommendation on definitions of joint coordinate system of various joints for the reporting of human joint motion-part I: ankle, hip, and spine*. International Society of Biomechanics. J Biomech, 2002. 35(4): p. 543-548.
10. Schwartz, M. and A. Rozumalski, *A new method for estimating joint parameters from motion data*. Journal of Biomechanics, 2005. 38(1): p. 107-116.

11. Kontaxis, A., et al., *A framework for the definition of standardized protocols for measuring upper-extremity kinematics*. Clinical Biomechanics, 2009. 24(3): p. 246-253.
12. Demircan, E., et al., *Human Motion Reconstruction and Synthesis of Human Skills*, in *Advances in Robot Kinematics: Motion in Man and Machine*2010, Springer Netherlands. p. 283-292.
13. De Groot, J. and R. Brand, *A three-dimensional regression model of the shoulder rhythm*. Clinical Biomechanics, 2001. 16(9): p. 735-743.
14. Cutti, A. and H. Veeger, *Shoulder biomechanics: today's consensus and tomorrow's perspectives*. Medical and Biological Engineering and Computing, 2009. 47(5): p. 463-466.
15. Lura, D., et al. *Robotic model for simulating upper body movement*. in *International Conference on Robotics and Biomimetics (ROBIO)*. 2009. Bangkok: IEEE Computer Society.
16. Lura, D., et al. *Robot kinematics based model to predict compensatory motion of transradial prosthesis while performing bilateral tasks*. in *International Conference on Robotics and Automation (ICRA)*. 2009. IEEE Press.
17. Lura, D., et al. *Simulated Compensatory Motion of Transradial Prostheses*. in *ASME International Mechanical Engineering Congress & Exposition (IMECE)*. 2008. Boston, Massachusetts: ASME.
18. Ziegler-Graham, K., et al., *Estimating the prevalence of limb loss in the United States: 2005 to 2050*. Arch Phys Med Rehabil, 2008. 89(3): p. 422-9.
19. Stansbury, L.G., et al., *Amputations in US military personnel in the current conflicts in Afghanistan and Iraq*. Journal of orthopaedic trauma, 2008. 22(1): p. 43.
20. Stinner, D.J., et al., *Prevalence of Late Amputations During the Current Conflicts in Afghanistan and Iraq*. Military Medicine, 2010. 175(12): p. 1027-1029.
21. Silcox, D.H., 3rd, et al., *Myoelectric prostheses. A long-term follow-up and a study of the use of alternate prostheses*. J Bone Joint Surg Am, 1993. 75(12): p. 1781-9.
22. Sherman, R.A., *Utilization of prostheses among US veterans with traumatic amputation: a pilot survey*. J Rehabil Res Dev, 1999. 36(2): p. 100-8.
23. Raichle, K.A., et al., *Prosthesis use in persons with lower-and upper-limb amputation*. Journal of rehabilitation research and development, 2008. 45(7): p. 961.
24. Kyberd, P., et al., *Survey of upper-extremity prosthesis users in Sweden and the United Kingdom*. JPO: Journal of Prosthetics and Orthotics, 2007. 19(2): p. 55.

25. Pylatiuk, C., S. Schulz, and L. Döderlein, *Results of an Internet survey of myoelectric prosthetic hand users*. *Prosthetics and Orthotics International*, 2007. 31(4): p. 362-370.
26. Schaffalitzky, E., et al., *Identifying the values and preferences of prosthetic users: a case study series using the repertory grid technique*. *Prosthet Orthot Int*, 2009. 33(2): p. 157-66.
27. Bertels, T., T. Schmalz, and E. Ludwigs, *Objectifying the Functional Advantages of Prosthetic Wrist Flexion*. *JPO: Journal of Prosthetics and Orthotics*, 2009. 21(2): p. 74-78 10.1097/JPO.0b013e3181a10f46.
28. Highsmith, M.J., et al., *Kinematic evaluation of terminal devices for kayaking with upper extremity amputation*. *Journal of Prosthetics and Orthotics*, 2007. 1: p. 7.
29. Carey, S.L., et al., *Compensatory movements of transradial prosthesis users during common tasks*. *Clin Biomech (Bristol, Avon)*, 2008. 23(9): p. 1128-35.
30. Kelly, B.M. *Upper Limb Prosthetics*. *Medical Devices* 2009 01/14/2009 [cited 2009 01/10/2011]; Available from: <http://emedicine.medscape.com/article/317234-overview>.
31. Davids, J.R., F. Rowan, and R.B. Davis, *Indications for orthoses to improve gait in children with cerebral palsy*. *J Am Acad Orthop Surg*, 2007. 15(3): p. 178-88.
32. Highsmith, M.J., et al., *Safety, energy efficiency, and cost efficacy of the C-Leg for transfemoral amputees: A review of the literature*. *Prosthetics and Orthotics International*, 2010(00): p. 1-16.
33. Hafner, B.J. and D.G. Smith, *Differences in function and safety between Medicare Functional Classification Level-2 and-3 transfemoral amputees and influence of prosthetic knee joint control*. *J Rehabil Res Dev*, 2009. 46(3): p. 417-33.
34. Ackermann, M. and A.J. van den Bogert. *Predictive simulation of gait in rehabilitation*. 2010. IEEE.
35. Fatone, S. and A.H. Hansen, *A model to predict the effect of ankle joint misalignment on calf band movement in ankle-foot orthoses*. *Prosthet Orthot Int*, 2007. 31(1): p. 76-87.
36. Crabtree, C.A. and J.S. Higginson, *Modeling neuromuscular effects of ankle foot orthoses (AFOs) in computer simulations of gait*. *Gait Posture*, 2009. 29(1): p. 65-70.
37. Wilson, J.R., et al., *A new methodology to measure the running biomechanics of amputees*. *Prosthet Orthot Int*, 2009. 33(3): p. 218-29.

38. Srinivasan, S., E.R. Westervelt, and A.H. Hansen, *A low-dimensional sagittal-plane forward-dynamic model for asymmetric gait and its application to study the gait of transtibial prosthesis users*. J Biomech Eng, 2009. 131(3): p. 031003.
39. Monheit, G. and N.I. Badler, *A kinematic model of the human spine and torso*. IEEE Computer Graphics and Applications, 1991. 11(2): p. 10.
40. Romilly, D., et al., *A functional task analysis and motion simulation for the development of a powered upper-limb orthosis*. Rehabilitation Engineering, IEEE Transactions on, 1994. 2(3): p. 119-129.
41. Maurel, W., *3D modeling of the human upper limb including the biomechanics of joints, muscles and soft tissues*, 1999.
42. Holzbaur, K., W. Murray, and S. Delp, *A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control*. Annals of biomedical engineering, 2005. 33(6): p. 829-840.
43. Abdel-Malek, K., et al., *Human upper body motion prediction*, in *ASM Conference on Applied Simulation and Modeling2004*: Rhodes, Greece.
44. Troncossi, M., et al., *Development of a prosthesis shoulder mechanism for upper limb amputees: application of an original design methodology to optimize functionality and wearability*. Med Biol Eng Comput, 2009. 47(5): p. 523-31.
45. Troncossi, M., V. Parenti-Castelli, and A. Davalli, *Design of upper limb prostheses: A new subject-oriented approach*, in *Journal of Mechanics in Medicine & Biology2005*, World Scientific Publishing Company. p. 383-390.
46. Lee, S., E. Sifakis, and D. Terzopoulos, *Comprehensive biomechanical modeling and simulation of the upper body*. ACM Trans. Graph, 2009. 28(4): p. 1–17.
47. Delp, S., et al., *OpenSim: open-source software to create and analyze dynamic simulations of movement*. Biomedical Engineering, IEEE Transactions on, 2007. 54(11): p. 1940-1950.
48. Delp, S., et al., *An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures*. Biomedical Engineering, IEEE Transactions on, 2002. 37(8): p. 757-767.
49. Abdullah, H.A., et al., *Dynamic biomechanical model for assessing and monitoring robot-assisted upper-limb therapy*. Journal of rehabilitation research and development, 2007. 44(1): p. 43.
50. Anglin, C. and U.P. Wyss, *Review of Arm Motion Analyses*. Proceedings of the Institution of Mechanical Engineers -- Part H -- Journal of Engineering in Medicine, 2000. 214(5): p. 541-555.

51. Gopura, R.A.R.C. and K. Kiguchi. *Mechanical designs of active upper-limb exoskeleton robots: State-of-the-art and design difficulties*. in *2009 IEEE 11th International Conference on Rehabilitation Robotics*. 2009. Kyoto, Japan.
52. Tolani, D., A. Goswami, and N.I. Badler, *Real-time inverse kinematics techniques for anthropomorphic limbs*. *Graphical models*, 2000. 62(5): p. 353-388.
53. Torres, E.B. and D. Zipser, *Reaching to grasp with a multi-jointed arm. I. Computational model*. *Journal of neurophysiology*, 2002. 88(5): p. 2355.
54. Vicon, M.S., *Plug-in Gait Product Guide*, in *Product Support2007*, OMG Plc: Oxford, UK.
55. Cappozzo, A., et al., *Surface-marker cluster design criteria for 3-D bone movement reconstruction*. *Biomedical Engineering, IEEE Transactions on*, 1997. 44(12): p. 1165-1174.
56. Piazza, S., N. Okita, and P. Cavanagh, *Accuracy of the functional method of hip joint center location: effects of limited motion and varied implementation*. *Journal of Biomechanics*, 2001. 34(7): p. 967-973.
57. Leardini, A., et al., *Validation of a functional method for the estimation of hip joint centre location*. *Journal of Biomechanics*, 1999. 32(1): p. 99-103.
58. Schönauer, C., *SKELETAL STRUCTURE GENERATION FOR OPTICAL MOTION CAPTURE*, in *Institute for Software Technologies and Interactive Systems2007*, Vienna University of Technology.
59. Dempster, W.T., *The anthropometry of body action*. *Annals of the New York Academy of Sciences*, 1955. 63(Dynamic Anthropometry): p. 559-585.
60. Hauschild, M., R. Davoodi, and G. Loeb, *A virtual reality environment for designing and fitting neural prosthetic limbs*. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 2007. 15(1): p. 9-15.
61. Cappozzo, A., et al., *Human movement analysis using stereophotogrammetry:: Part 1: theoretical background*. *Gait & posture*, 2005. 21(2): p. 186-196.
62. Drillis, R., R. Contini, and M. Bluestein, *Body segment parameters; a survey of measurement techniques*. *Artificial limbs*, 1964. 25: p. 44.
63. Winter, D., *Biomechanics and motor control of human movement*2009: Wiley.
64. Arun, K.S., T.S. Huang, and S.D. Blostein, *Least-squares fitting of two 3-D point sets*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 1987(5): p. 698-700.
65. Chang, P.H., *A Closed-Form Solution for Inverse Kinematics of Robot Manipulators with Redundancy*. *Ieee Journal of Robotics and Automation*, 1987. 3(5): p. 393-403.

66. Khadem, S. and R. Dubey. *A Global redundant robot control scheme for obstacle avoidance*. in *IEEE Southeast Conference*. 1988. Knoxville, TN.
67. Nakamura, Y., *Advanced Robotics: Redundancy and Optimization*. 1st ed 1990, Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. .
68. McGhee, S., et al. *Probability-based weighting of performance criteria for a redundant manipulator*. in *IEEE International Conference on Robotics and Automation (ICRA)*. 1994. San Diego, CA.
69. Guez, A. and Z. Ahmad. *Solution to the inverse kinematics problem in robotics by neural networks*. in *International Conference on Neural Networks (ICNN)*. 1988. San Diego, CA, USA: IEEE.
70. Kiguchi, K. and Q. Quan. *Muscle-model-oriented EMG-based control of an upper-limb power-assist exoskeleton with a neuro-fuzzy modifier*. 2008. IEEE.
71. Yang, J., et al. *Multi-objective optimization for upper body posture prediction*. 2004. Citeseer.
72. Yoshikawa, T., *Foundations of robotics : analysis and control* 1990, Cambridge, Mass.: MIT Press. x, 285 p.
73. Chan, T.F. and R.V. Dubey, *A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators*. *Robotics and Automation, IEEE Transactions on*, 1995. 11(2): p. 286-292.
74. McGhee, S., et al. *Probability -based weighting of performance criteria for a redundant manipulator*. in *IEEE International Conference on Robotics and Automation (ICRA)*. 1994. San Diego, CA.
75. Beiner, L. and J. Mattila, *An improved pseudoinverse solution for redundant hydraulic manipulators*. *Robotica*, 1999. 17(02): p. 173-179.
76. Zergeroglu, E., et al., *Nonlinear tracking control of kinematically redundant robot manipulators*. *IEEE-ASME Transactions on Mechatronics*, 2004. 9(1): p. 129-132.
77. Kwon, W., B. Hee Lee, and M. Hwan Choi, *Resolving kinematic redundancy of a robot using a quadratically constrained optimization technique*. *Robotica*, 1999. 17(05): p. 503-511.
78. Ellekilde, L., et al., *Robust inverse Jacobian control with joint limit and singularity handling for visual servoing applications*. *The International Journal of Robotics Research*, 2006.
79. Alqasemi, R. and R. Dubey. *Combined Mobility and Manipulation Control of a Newly Developed 9-DoF Wheelchair-Mounted Robotic Arm System*. in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA 2007)*. 2007. Rome, Italy.

80. Farello, F., R. Alqasemi, and R. Dubey. *Optimized dual-trajectory tracking control of a 9-DoF WMRA system for ADL tasks*. in *International Conference on Robotics and Automation (ICRA)*. 2010. Anchorage, Alaska, USA: IEEE.
81. Josin, G., D. Charney, and D. White. *Robot control using neural networks*. 1988. IEEE.
82. Xia, Y. and J. Wang, *A dual neural network for kinematic control of redundant robot manipulators*. *Systems, Man, and Cybernetics, Part B: Cybernetics*, IEEE Transactions on, 2001. 31(1): p. 147-154.
83. Xia, Y.S., G. Feng, and J. Wang, *A primal-dual neural network for online resolving constrained kinematic redundancy in robot motion control*. *Systems, Man, and Cybernetics, Part B: Cybernetics*, IEEE Transactions on, 2005. 35(1): p. 54-64.
84. Kundu, K., K. Kiguchi, and E. Horikawa. *Estimation of daily forearm and wrist motion from shoulder and elbow kinematics by using artificial neural networks*. 2008. IEEE.
85. Inohira, E. and H. Yokoi, *Improvement of a neural network-based motion generator with bimanual coordination for upper limb prosthesis*. *Artificial Life and Robotics*, 2010. 15(4): p. 504-507.
86. Ramírez-García, A., L. Leija, and R. Muñoz, *Active upper limb prosthesis based on natural movement trajectories*. *Prosthetics and Orthotics International*, 2010. 34(1): p. 58-72.
87. Rasmussen, C. and C. Williams, *Gaussian Processes for Machine Learning* 2006: MIT Press.
88. Lee, J., et al., *Interactive control of avatars animated with human motion data*. *ACM Transactions on Graphics*, 2002. 21(3): p. 491-500.
89. Wei, X., J. Min, and J. Chai, *Physically-Valid Statistical Models for Human Motion Generation*.
90. Corke, P.I., *A robotics toolbox for MATLAB*. *Robotics & Automation Magazine*, IEEE, 1996. 3(1): p. 24-32.
91. Mell, A.G., B.L. Childress, and R.E. Hughes, *The effect of wearing a wrist splint on shoulder kinematics during object manipulation*. *Archives of physical medicine and rehabilitation*, 2005. 86(8): p. 1661-1664.
92. Gordon, C., et al., *Anthropometric Survey of US Army Personnel: Methods and Summary Statistics 1988*, 1989.

93. Lura, D., S. Carey, and R. Dubey. *VALIDATION OF FUNCTIONAL METHODS FOR CALCULATING SHOULDER JOINT CENTERS USING 3D MOTION ANALYSIS*. in *International Mechanical Engineering Congress & Exposition*. 2010. Vancouver, BC, Canada.
94. Lura, D., S. Carey, and R. Dubey, *Validation of Functional Methods for Calculating the Shoulder Joint Center Using 3D Motion*, in *ASME International Mechanical Engineering Congress and Exposition (IMECE)2010*, ASME: Vancouver, Canada.
95. Walker, M.R. and M.J. Rainbow. *MATLAB Toolbox For C3Dserver - Version 2*. 2006 [cited 2011 April 6]; Available from: <http://www.c3d.org/applications/matlab.html>.
96. London, J., *Kinematics of the elbow*. The Journal of Bone and Joint Surgery, 1981. 63(4): p. 529.
97. Craig, J.J., *Introduction to robotics: mechanics and control*. 2nd ed1989: Addison-Wesley Longman Publishing Co., Inc.
98. Šenk, M. and L. Chèze, *Rotation sequence as an important factor in shoulder kinematics*. Clinical Biomechanics, 2006. 21, Supplement 1(0): p. S3-S8.
99. Williams, S., et al., *An upper body model for the kinematical analysis of the joint chain of the human arm*. Journal of Biomechanics, 2006. 39(13): p. 2419-2429.
100. Lura, D., S. Carey, and R. Dubey. *Automatic Generation of a Subject Specific Upper Body Model From Motion Analysis Data*. in *Proceeding of the ASME 2011 International Mechanical Engineering Congress & Exposition (IMECE2011)*. 2011. Denver, Colorado, USA: ASME.
101. Bouwsema, H., C. der Sluis, and R. Bongers, *Movement characteristics of upper extremity prostheses during basic goal-directed tasks*. Clinical Biomechanics, 2010. 25(6): p. 523-529.
102. Grochow, K., et al., *Style-based inverse kinematics*. ACM Transactions on Graphics (TOG), 2004. 23(3): p. 522-531.
103. Jung, E., D. Kee, and M. Chung, *Upper body reach posture prediction for ergonomic evaluation models*. International Journal of Industrial Ergonomics, 1995. 16(2): p. 95-107.
104. Beale, M., M. Hagan, and H. Demuth, *Matlab neural network toolbox user's guide version 6*. The MathWorks Inc. 2010.
105. Lamounier, E., et al. *On the use of Virtual and Augmented Reality for upper limb prostheses training and simulation*. 2010. IEEE.
106. Fish, D.R. and L. Wingate, *Sources of goniometric error at the elbow*. Physical Therapy, 1985. 65(11): p. 1666-1670.

107. Blonna, D., et al., *Accuracy and inter-observer reliability of visual estimation compared to clinical goniometry of the elbow*. Knee Surgery, Sports Traumatology, Arthroscopy, 2011: p. 1-8.
108. Veeger, H., et al., *Parameters for modeling the upper extremity*. Journal of Biomechanics, 1997. 30(6): p. 647-652.
109. Sensinger, J.W. and R.F. Weir, *Modeling and preliminary testing socket-residual limb interface stiffness of above-elbow prostheses*. IEEE Trans Neural Syst Rehabil Eng, 2008. 16(2): p. 184-90.

Appendices

Appendix A: Data Collection Documents


A.1 Subject Measurement Form

USF RRT Motion Analysis Lab

SUBJECT MEASUREMENT FORM

Study Name _____

Subject Identifier _____ Collection Date _____ Researcher _____



Patient Information

Patient's Name _____

Age _____ Sex _____ Weight _____ Height _____

Amputation Cause: _____ Date _____

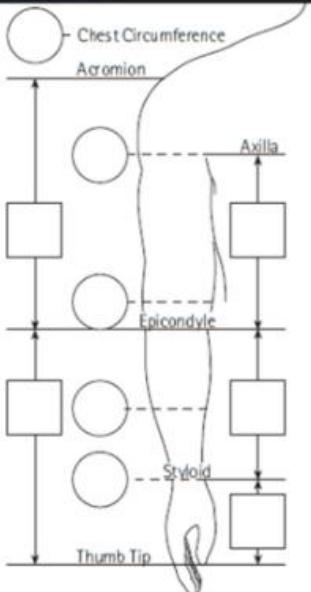
Amputation Level: _____ Right Left

Caucasian Brown Other Previous Prosthesis Worn? Yes No Shade# _____

(If this is a Replacement Prosthesis, please include measurements from old prosthesis on a separate form)

Is the Prosthesis to be shipped Ready for Fitting? Yes No Date _____

Measurements



Socket

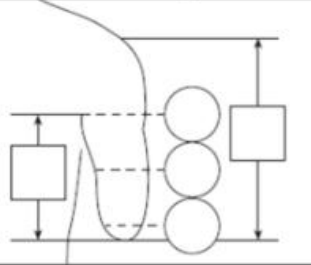
Open Socket End Bearing Split Socket

Part# _____ Part# _____ Part# _____

Model _____ Model _____ Model _____

Lightweight Standard Weight Heavy Duty

Special Inst: _____



Joint Type

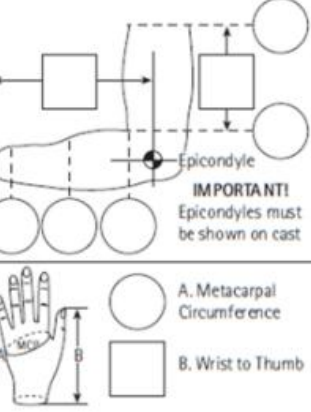
Wrist Elbow Hinge

Part# _____ Part# _____ Part# _____

Model _____ Model _____ Model _____

Check Here for Lift Assist

Special Inst: _____



IMPORTANT!
Epicondyles must be shown on cast

Terminal Device

Hook Hand Glove

Part# _____ Part# _____ Part# _____

Model _____ Model _____ Model _____

Special Inst: _____

Include

Harness Cuff Cables

Part# _____ Part# _____ Part# _____

Model _____ Model _____ Model _____

Other Accessories: _____

Special Inst: _____

Appendix A (Continued)

A.2 Data Collection Checklist

Data Collection Checklist				
Subject ID: _____		Name: _____		Date: _____
<input type="checkbox"/> Subject sign informed consent <input type="checkbox"/> Subject sign photography release (optional) <input type="checkbox"/> Measure and record subject height & weight <input type="checkbox"/> Measure and record prosthesis weight and dimensions <input type="checkbox"/> Attach marker set for pipeline based on subjects side and level of amputation (Data Collection\MarkerSets*.mkr) <input type="checkbox"/> Attach markers				
Torso	Other	Right Arm	Left Arm	Prosthetic
<input type="checkbox"/> T1	<input type="checkbox"/> RASI	<input type="checkbox"/> RSHOA	<input type="checkbox"/> LSHOA	<input type="checkbox"/> RSLA
<input type="checkbox"/> CLAV	<input type="checkbox"/> RPSI	<input type="checkbox"/> RSHOP	<input type="checkbox"/> LSHOP	<input type="checkbox"/> RSLP
<input type="checkbox"/> LBAK	<input type="checkbox"/> LASI	<input type="checkbox"/> RELB	<input type="checkbox"/> LELB	<input type="checkbox"/> SCKTA
<input type="checkbox"/> STERN	<input type="checkbox"/> LPSI	<input type="checkbox"/> RELBM	<input type="checkbox"/> LELBM	<input type="checkbox"/> SCKTP
<input type="checkbox"/> RGT	<input type="checkbox"/> RIC	<input type="checkbox"/> RWRA	<input type="checkbox"/> LWRA	<input type="checkbox"/> SCKTL
<input type="checkbox"/> LGT	<input type="checkbox"/> LIC	<input type="checkbox"/> RWRB	<input type="checkbox"/> LWRB	<input type="checkbox"/> RUPA
<input type="checkbox"/> RFRA	<input type="checkbox"/> LFRA	<input type="checkbox"/> RFIN	<input type="checkbox"/> LFIN	<input type="checkbox"/> LUPA
<input type="checkbox"/> Calibrate Cameras and Force Plates				
<input type="checkbox"/> Range of motion tasks x 3				
<input type="checkbox"/> Static Trial	<input type="checkbox"/> Shoulder Rotation			
<input type="checkbox"/> Elbow Flexion/Extension	<input type="checkbox"/> Torso Flexion/Extension			
<input type="checkbox"/> Forearm Pronation/Supination	<input type="checkbox"/> Torso Lateral Flexion			
<input type="checkbox"/> Shoulder Flexion/Extension	<input type="checkbox"/> Torso Rotation			
<input type="checkbox"/> Shoulder Abduction/Adduction				
<input type="checkbox"/> Recalibrate cameras and force plates				
<input type="checkbox"/> Activities of daily living x 3				
<input type="checkbox"/> Drinking From a Cup	<input type="checkbox"/> Eating With a Knife and Fork			
<input type="checkbox"/> Brushing Hair	<input type="checkbox"/> Lifting a Laundry Basket			
<input type="checkbox"/> Opening a Door				
<input type="checkbox"/> Recalibrate cameras and force plates				
<input type="checkbox"/> Doff prosthesis and place RSLA1-2 and RSLP1-2 on residual limb				
<input type="checkbox"/> Range of motion tasks without prosthesis x 3				
<input type="checkbox"/> Static Trial	<input type="checkbox"/> Shoulder Abduction/Adduction			
<input type="checkbox"/> Shoulder Flexion/Extension	<input type="checkbox"/> Shoulder Rotation			

Appendix B: Matlab Code

B.1 CreateUBM.m

```
% Model creation algorithm for the "Robotic Human Upper Body Model" (RHBM).  
% Version 1 release 02/06/2011 Derek J. Lura, University of South Florida.  
% Requires the Robotics Toolbox (P. Corke) and c3d server (M. R. Walker).
```

```
% Add subfunctions to the current path  
path ([cd, '\SubFunctions'], path)
```

```
% Clear variables from the current workspace  
clear all
```

```
% Close all open figure windows (plots)  
close all
```

```
% For all the specified subjects 's':  
% Load all of the range of motion trials to the ROM structure.  
% Then use them to calculate the joint centers.  
for s=[1:10] % Start subject loop
```

```
clear ROM ADL
```

```
% Determine if plot functions should be run  
plots = 0;
```

```
% Set the Subject listing.  
subjects = ...
```

```
['C01'; 'C02'; 'C03'; 'C04'; 'C05'; 'C06'; 'C07'; 'C08'; 'C09'; 'C10'; ...  
 'B01'; 'B02'; 'B03'; 'B04'; 'B05'; 'B06'; 'B07'; 'B08'; 'B09'; 'B10'; ...  
 'R01'; 'R02'; 'R03'; 'R04'; 'R05'; 'R06'; 'R07'; 'R08'; 'R09'; 'R10'; ...  
 'H01'; 'H02'; 'H03'; 'H04'; 'H05'; 'H06'; 'H07'; 'H08'; 'H09'; 'H10'; ...  
 'PT1'];
```

```
markers = {'T1', 'CLAV', 'RASI', 'RPSI', 'LASI', 'LPSI', ...  
          'RSHOA', 'RSHOP', 'RELB', 'RELB', 'RWRA', 'RWRB', 'RFIN', ...  
          'LSHOA', 'LSHOP', 'LELB', 'LELB', 'LWRA', 'LWRB', 'LFIN', ...  
          'T10', 'STRN', 'LBAK', 'RIC', 'LIC', 'RUPA', 'RFRA', 'LPUA', 'LFRA'};
```

```
Nmarker = size(markers,2);
```

```
% Initialize structure for joint center locations  
Centers.Torso = [];
```

```
% Change directory to the ROM folder of subjects(s)  
cd (['Subjects\', subjects(s,:), '\ROM'])  
% Load all of the *.c3d (motion trails) file information
```

Appendix B (Continued)

```
foldernfo = dir('*.*c3d');
% Set the variable subject to the current subjects(s)
subject = removewhite(subjects(s,:));
% Create the feild for subject, in structure ROM, set the feild filenames
% to the names of the files in the folder.
ROM.(subject).filenames = char(foldernfo.name);
% Create a variable for the number of .c3d files in the folder
ROM.(subject).nfiles = size(ROM.(subject).filenames,1);

% Create e feild for the compiled pelvis tracking markers
ROM.(subject).pelvisCompiled = [];

% For all files in ROM of subject
for i=1:ROM.(subject).nfiles;
    % Load c3d server.
    newServer = c3dserver;
    % Open the c3d files
    openc3d(newServer,0,ROM.(subject).filenames(i,:));
    % Set the variable name to the current file
    name = removewhite(ROM.(subject).filenames(i,:));
    % Get all of the targets (makers) from the c3d server
    newtarget = get3dtargets(newServer,1);
    % Assign the targets to the trial feild
    ROM.(subject).(name) = newtarget;

    % Set Nsamples equal to the number of samples in the trial.
    Nsamples = size(ROM.(subject).(name).RASI,1);

    % Filter the raw marker data
    for j=1:Nmarker
        % If C7 marker convention is used rename to T1 convention
        if isfield(ROM.(subject).(name), 'C7')
            ROM.(subject).(name).T1 = WMAfilter(11,getfield(ROM.(subject).(name), 'C7',
{1:Nsamples,1:3}));
            end

            if isfield(ROM.(subject).(name), cell2mat(markers(j)))
                ROM.(subject).(name).(char(markers(j))) = ...
                WMAfilter(11,getfield(ROM.(subject).(name), char(markers(j)),
{1:Nsamples,1:3}));
            elseif j<=20
                disp(['No ', cell2mat(markers(j)), ' in ', subject, ', ', name])
                ROM.(subject).(name).(cell2mat(markers(j))) = zeros(Nsamples, 3)*NaN;
            end
        end
    end
end
```

Appendix B (Continued)

```
end
end

% Create the pelvis segment
ROM.(subject).Static.MPSI =
(ROM.(subject).Static.RPSI+ROM.(subject).Static.LPSI)/2;
ROM.(subject).Static.Pelvis = createSegment(ROM.(subject).Static.MPSI,
(ROM.(subject).Static.RASI-ROM.(subject).Static.LASI), (ROM.(subject).Static.RASI-
ROM.(subject).Static.MPSI), 'zyx');
ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.RASI);
ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.LASI);
ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.RPSI);
ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.LPSI);
if isfield(ROM.(subject).Static, 'RIC')
    ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.RIC);
end
if isfield(ROM.(subject).Static, 'LIC')
    ROM.(subject).Static.Pelvis = addPoint2(ROM.(subject).Static.Pelvis,
ROM.(subject).Static.LIC);
end

% Calculate the mean relative position of pelvis markers (for cluster
% reconstruction).
ROM.(subject).X(:, :) = nanmean(ROM.(subject).Static.Pelvis.Point);

% Find offset for verticle pelvis orientation
avgP = rpy2tr(nanmean(tr2rpy(ROM.(subject).Static.Pelvis.HT)));
Zvec = zeros(size(ROM.(subject).Static.RASI));
Zvec(:,3) = 1;
ROM.(subject).Static.VertPelvis = createSegment(ROM.(subject).Static.MPSI, Zvec,
(ROM.(subject).Static.RASI-ROM.(subject).Static.LASI), 'yxz');
avgV = rpy2tr(nanmean(tr2rpy(ROM.(subject).Static.VertPelvis.HT)));
ROM.(subject).OffsetT = avgP^-1*avgV;
for i=1:ROM.(subject).nfiles;
    name = removewhite(ROM.(subject).filenames(i,:));

    Markers = {'RPSI', 'RASI', 'LPSI', 'LASI', 'RFIN', 'LFIN'};
    for m = 1:size(Markers,2)
        if ~isfield(ROM.(subject).(name), Markers(m))
```

Appendix B (Continued)

```
ROM.(subject).(name).(char(Markers(m))) =
ones(size(ROM.(subject).(name).RASI))*NaN;
end
end

% Reconstruct any missing points on the pelvis (requires at least 3
% pelvis markers).
if isfield(ROM.(subject).(name), 'RIC')
    [ROM.(subject).(name).RASI, ROM.(subject).(name).LASI,
ROM.(subject).(name).RPSI, ROM.(subject).(name).LPSI] = ...
    clusterReconstruct(ROM.(subject).X, ROM.(subject).(name).RASI,
ROM.(subject).(name).LASI, ROM.(subject).(name).RPSI, ROM.(subject).(name).LPSI,
...
    ROM.(subject).(name).RIC, ROM.(subject).(name).LIC);
else
    [ROM.(subject).(name).RASI, ROM.(subject).(name).LASI,
ROM.(subject).(name).RPSI, ROM.(subject).(name).LPSI] = ...
    reconstruct(ROM.(subject).(name).RASI, ROM.(subject).(name).LASI,
ROM.(subject).(name).RPSI, ROM.(subject).(name).LPSI);
end

% Create virtual points based on marker positions.
% Calcualte anterior and posterior pelvis center markers
ROM.(subject).(name).MPSI =
(ROM.(subject).(name).RPSI+ROM.(subject).(name).LPSI)/2;

% For Upper Body collections we will originate at the hip.
% From ISB Recomend Standards:
% The origin is located at MPSI
% Z axis, connecting line from R&LASI, positive R.
% X axis, orthogonal to the Z-axis lying in the plane defined by RASI,
% LASI, and MPSI (positive anterior)
% Y aixs, orthagonal to X and Z.

% Using ISB Recomendations for Torso as an example
% Origin coincident with IJ (Incisura Jugularis)
ROM.(subject).(name).Pelvis = createSegment(ROM.(subject).(name).MPSI,
(ROM.(subject).(name).RASI-ROM.(subject).(name).LASI),
(ROM.(subject).(name).RASI-ROM.(subject).(name).MPSI), 'zyx',
ROM.(subject).OffsetT);
% Add the makers to the pelvis feild, within the pelvis frame
ROM.(subject).(name).Pelvis = addPoint2(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).RASI);
```

Appendix B (Continued)

```
ROM.(subject).(name).Pelvis = addPoint2(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).LASI);
ROM.(subject).(name).Pelvis = addPoint2(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).RPSI);
ROM.(subject).(name).Pelvis = addPoint2(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).LPSI);
ROM.(subject).(name).Pelvis = addPoint2(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).MPSI);

% Transform the torso marker into the pelvis frame and add to the
% Pelvis field
ROM.(subject).(name).Pelvis = addDistalPoint(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).T1);
ROM.(subject).(name).Pelvis = addDistalPoint(ROM.(subject).(name).Pelvis,
ROM.(subject).(name).CLAV);

% Compile the static marker position, and all of the motion trials
if strcmpi(name,'static')
    ROM.(subject).sPelvis = mean(ROM.(subject).(name).Pelvis.DistalPoint);
else
    ROM.(subject).pelvisCompiled = cat(1, ROM.(subject).pelvisCompiled,
ROM.(subject).(name).Pelvis.DistalPoint);
end

end % File loop end

% Set the directory back to /Subjects/Subject
cd ..

j = 1;
% Remove all gaps in the compiled motion data (required for joint center
% calculations).
while j<=size(ROM.(subject).pelvisCompiled,1)
    if (~(abs(ROM.(subject).pelvisCompiled(j,1,1)) >=
0)||~(abs(ROM.(subject).pelvisCompiled(j,1,2)) >= 0))
        ROM.(subject).pelvisCompiled(j,,:) = [];
    else
        j=j+1;
    end
end

% Calculate the torso joint center.
close all
```


Appendix B (Continued)

```
ROM.(subject).TorsoCenter = MLOptim(ROM.(subject).pelvisCompiled,[0,0,0]);
set(gcf,'name',['Torso ',subject]);
saveas(gcf,['C:\Documents and
Settings\dlura\Desktop\RHBM\Figures\' ,Torso',subject,'.fig']);
Centers.Torso = [Centers.Torso; ROM.(subject).TorsoCenter];
```

% Plot the torso marker positions relative to the pelvis frame

```
if plots
    figure('name',['Total Spine',subject])
    hold off
    plot3(ROM.(subject).pelvisCompiled(:,1,1), ROM.(subject).pelvisCompiled(:,2,1),
ROM.(subject).pelvisCompiled(:,3,1), 'Color', [0,0.5,0])
    hold on
    plot3(ROM.(subject).pelvisCompiled(:,1,2), ROM.(subject).pelvisCompiled(:,2,2),
ROM.(subject).pelvisCompiled(:,3,2))
    axis equal
end
```

% Calculate the static positions relative to the torso joint center.

```
for j=1:size(ROM.(subject).sPelvis,3)
    ROM.(subject).sPelvis(:,j) = ROM.(subject).sPelvis(:,j) -
ROM.(subject).TorsoCenter;
end
```

```
ROM.(subjects(s,:)).RTorsoCenter = ROM.(subjects(s,:)).TorsoCenter;
```

```
if 0 %isfield(ROM.(subject).Static, 'T10')
    for j = 1:size(ROM.(subject).Static.RASI,1);
        ROM.(subject).Static.TorsoJC(j,:) =
(ROM.(subject).Static.Pelvis.HT(1:3,1:4,j)*[ROM.(subject).TorsoCenter';1]);
    end
```

% Calculate virtual torso marker (origin) and create torso segment

```
ROM.(subject).Static.MTOR =
(ROM.(subject).Static.T1+ROM.(subject).Static.CLAV)/2;
ROM.(subject).Static.Torso = createSegment(ROM.(subject).Static.TorsoJC,
(ROM.(subject).Static.MTOR-ROM.(subject).Static.TorsoJC), (ROM.(subject).Static.T1-
ROM.(subject).Static.CLAV), 'yzz');
    % Add markers to the torso segment
    ROM.(subject).Static.Torso = addPoint2(ROM.(subject).Static.Torso,
ROM.(subject).Static.T1);
    ROM.(subject).Static.Torso = addPoint2(ROM.(subject).Static.Torso,
ROM.(subject).Static.T10);
```


Appendix B (Continued)

```
ROM.(subject).Static.Torso = addPoint2(ROM.(subject).Static.Torso,  
ROM.(subject).Static.CLAV);
```

```
ROM.(subject).Static.Torso = addPoint2(ROM.(subject).Static.Torso,  
ROM.(subject).Static.TorsoJC);
```

```
ROM.(subject).TX(:, :) = nanmean(ROM.(subject).Static.Torso.Point);
```

```
for i=1:ROM.(subject).nfiles;  
    name = removewhite(ROM.(subject).filenames(i,:));  
    for j = 1:size(ROM.(subject).(name).RASI,1)  
        ROM.(subject).(name).TorsoJC(j,:) =  
(ROM.(subject).(name).Pelvis.HT(1:3,1:4,j)*[ROM.(subject).TorsoCenter';1]);  
    end  
    % Reconstruct any missing points on the torso (requires at least 3  
    % pelvis markers).  
    [ROM.(subject).(name).T1, ROM.(subject).(name).T10,  
ROM.(subject).(name).CLAV] = ...  
        clusterReconstruct(ROM.(subject).TX, ROM.(subject).(name).T1,  
ROM.(subject).(name).T10, ROM.(subject).(name).CLAV, ...  
            ROM.(subject).(name).TorsoJC);  
    end  
end
```

```
[ROM, Centers.RShoulder] =  
autoSegment(ROM, 'Pelvis', 'RTorso', 'RShoulder', 'zyx', {'CLAV', 'T1'}, {'RSHOP', 'RSHOA'  
});  
close all  
[ROM, Centers.LShoulder] =  
autoSegment(ROM, 'Pelvis', 'Torso', 'LShoulder', 'zyx', {'CLAV', 'T1'}, {'LSHOP', 'LSHOA'})  
;  
close all
```

```
[ROM, Centers.RUpperArm] =  
autoSegment(ROM, 'RTorso', 'RShoulder', 'RUpperArm', 'zyx', {'RSHOP', 'RSHOA'}, {'REL  
B', 'RELBM'});  
close all  
[ROM, Centers.LUpperArm] =  
autoSegment(ROM, 'Torso', 'LShoulder', 'LUpperArm', 'zyx', {'LSHOP', 'LSHOA'}, {'LELB',  
LELBM'});  
close all  
[ROM, Centers.RForearm] =  
autoSegment(ROM, 'RShoulder', 'RUpperArm', 'RForearm', 'zxy', {'RELB', 'RELBM'}, {'RW  
RB', 'RWRA'});  
close all
```

Appendix B (Continued)

```
[ROM, Centers.LForearm] =  
autoSegment(ROM, 'LShoulder', 'LUpperArm', 'LForearm', 'zyx', {'LLEB', 'LELB'}, {'LW  
RB', 'LWRA'});  
close all
```

```
[ROM, Centers.RHand] =  
autoSegment(ROM, 'RUpperArm', 'RForearm', 'RHand', 'zyx', {'RWRB', 'RWRA'}, {'RFIN', '  
RFIN'});  
close all
```

```
[ROM, Centers.LHand] =  
autoSegment(ROM, 'LUpperArm', 'LForearm', 'LHand', 'zyx', {'LWRB', 'LWRA'}, {'LFIN', 'L  
FIN'});  
close all
```

```
[ROM] = autoSegment(ROM, 'RForearm', 'RHand', [], 'zyx', {'RWRB', 'RWRA', 'RFIN'});  
close all
```

```
[ROM] = autoSegment(ROM, 'LForearm', 'LHand', [], 'zyx', {'LWRB', 'LWRA', 'LFIN'});  
close all
```

```
ROM = autoFindTheta(ROM);
```

```
% Load all of the activiy if daily living trials to the ADL structure.  
% For all / or n subjects:  
% Change directory to the ADL folder of subjects(s)  
cd ('ADL\  
% Load all of the *.c3d (motion trails) files  
foldernfo = dir('*.*c3d');  
% Create the feild for subject, in structure ADL, set the feild filenames  
% to the names of the files in the folder.  
ADL.(subject).filenames = char(foldernfo.name);  
% Create a variable for the number of .c3d files in the folder  
ADL.(subject).nfiles = size(ADL.(subject).filenames,1);
```

```
ADL.(subject).TorsoCenter = ROM.(subject).TorsoCenter;  
ADL.(subject).RTorsoCenter = ROM.(subject).RTorsoCenter;
```

```
ADL.(subject).RShoulderCenter = ROM.(subject).RShoulderCenter;  
ADL.(subject).RUpperArmCenter = ROM.(subject).RUpperArmCenter;  
ADL.(subject).RForearmCenter = ROM.(subject).RForearmCenter;  
ADL.(subject).RHandCenter = ROM.(subject).RHandCenter;  
ADL.(subject).LShoulderCenter = ROM.(subject).LShoulderCenter;  
ADL.(subject).LUpperArmCenter = ROM.(subject).LUpperArmCenter;  
ADL.(subject).LForearmCenter = ROM.(subject).LForearmCenter;  
ADL.(subject).LHandCenter = ROM.(subject).LHandCenter;
```

Appendix B (Continued)

```
% For all files in ADL of subject
for i=1:ADL.(subject).nfiles;
    % Load c3d server.
    newServer = c3dserver;
    % Open the c3d files
    openc3d(newServer,0,ADL.(subject).filenames(i,:));
    % Set the variable name to the current file
    name = removewhite(ADL.(subject).filenames(i,:));
    % Get all of the targets (makers) from the c3d server
    newtarget = get3dtargets(newServer,1);
    % Assign the targets to the trial feild
    ADL.(subject).(name) = newtarget;

    % Set Nsamples equal to the number of samples in the trial.
    if isfield(ADL.(subject).(name), 'RPSI')
        Nsamples = size(ADL.(subject).(name).RPSI,1);
    elseif isfield(ADL.(subject).(name), 'LPSI')
        Nsamples = size(ADL.(subject).(name).LPSI,1);
    elseif isfield(ADL.(subject).(name), 'RASI')
        Nsamples = size(ADL.(subject).(name).RASI,1);
    else
        disp([name, 'No pelvis markers'])
        continue
    end

    % Filter the raw marker data
    for j=1:Nmarker
        % If C7 marker convention is used rename to T1 convention
        if isfield(ADL.(subject).(name), 'C7')
            ADL.(subject).(name).T1 = WMAfilter(11,getfield(ADL.(subject).(name), 'C7',
{1:Nsamples,1:3}));
        end

        if isfield(ADL.(subject).(name), cell2mat(markers(j)))
            ADL.(subject).(name).(char(markers(j))) = ...
                WMAfilter(11,getfield(ADL.(subject).(name), char(markers(j)),
{1:Nsamples,1:3}));
        elseif j<=20
            disp(['No ', cell2mat(markers(j)), 'in ', subject, ', ', name])
            ADL.(subject).(name).(cell2mat(markers(j))) = zeros(Nsamples, 3)*NaN;
        end
    end
end
end
```

Appendix B (Continued)

```
ADL.(subject).Static.MPSI =  
(ADL.(subject).Static.RPSI+ADL.(subject).Static.LPSI)/2;  
ADL.(subject).Static.Pelvis = createSegment(ADL.(subject).Static.MPSI,  
(ADL.(subject).Static.RASI-ADL.(subject).Static.LASI), (ADL.(subject).Static.RASI-  
ADL.(subject).Static.MPSI), 'zyx');
```

```
ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.RASI);  
ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.LASI);  
ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.RPSI);  
ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.LPSI);  
if isfield(ADL.(subject).Static, 'RIC')  
    ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.RIC);  
end  
if isfield(ADL.(subject).Static, 'LIC')  
    ADL.(subject).Static.Pelvis = addPoint2(ADL.(subject).Static.Pelvis,  
ADL.(subject).Static.LIC);  
end
```

```
ADL.(subject).X(:, :) = nanmean(ADL.(subject).Static.Pelvis.Point);  
avgP = rpy2tr(nanmean(tr2rpy(ADL.(subject).Static.Pelvis.HT)));  
Zvec = zeros(size(ADL.(subject).Static.RASI));  
Zvec(:,3) = 1;  
ADL.(subject).Static.VertPelvis = createSegment(ADL.(subject).Static.MPSI, Zvec,  
(ADL.(subject).Static.RASI-ADL.(subject).Static.LASI), 'yxz');  
avgV = rpy2tr(nanmean(tr2rpy(ADL.(subject).Static.VertPelvis.HT)));  
ADL.(subject).OffsetT = avgP^-1*avgP;
```

```
for i=1:ADL.(subject).nfiles;  
    name = removewhite(ADL.(subject).filenames(i,:));
```

```
    Markers = {'RPSI', 'RASI', 'LPSI', 'LASI', 'RFIN', 'LFIN'};  
    for m = 1:size(Markers,2)  
        if ~isfield(ADL.(subject).(name), Markers(m))  
            ADL.(subject).(name).(char(Markers(m))) =  
ones(size(ADL.(subject).(name).RASI))*NaN;  
        end  
    end
```

% Reconstruct any missing points on the pelvis (requires at least 3

Appendix B (Continued)

```
% pelvis markers).
if isfield(ADL.(subject).(name), 'RIC')
    [ADL.(subject).(name).RASI, ADL.(subject).(name).LASI,
    ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI] = ...
    clusterReconstruct(ADL.(subject).X, ADL.(subject).(name).RASI,
    ADL.(subject).(name).LASI, ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI,
    ...
    ADL.(subject).(name).RIC, ADL.(subject).(name).LIC);
else
    [ADL.(subject).(name).RASI, ADL.(subject).(name).LASI,
    ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI] = ...
    reconstruct(ADL.(subject).(name).RASI, ADL.(subject).(name).LASI,
    ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI);
end

% Reconstruct any missing points on the pelvis (requires at least 3
% pelvis markers).
[ADL.(subject).(name).RASI, ADL.(subject).(name).LASI,
ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI] = ...
reconstruct(ADL.(subject).(name).RASI, ADL.(subject).(name).LASI,
ADL.(subject).(name).RPSI, ADL.(subject).(name).LPSI);

% Create virtual points based on marker positions.
% Calcualte anterior and posterior pelvis center markers
ADL.(subject).(name).MASI =
(ADL.(subject).(name).RASI+ADL.(subject).(name).LASI)/2;
ADL.(subject).(name).MPSI =
(ADL.(subject).(name).RPSI+ADL.(subject).(name).LPSI)/2;

% Using ISB Recomendations for Torso as an example
% Origin coincident with IJ (Incisura Jugularis)
ADL.(subject).(name).Pelvis = createSegment(ADL.(subject).(name).MPSI,
(ADL.(subject).(name).RASI-ADL.(subject).(name).LASI),
(ADL.(subject).(name).MASI-ADL.(subject).(name).MPSI), 'zyx',
ADL.(subject).OffsetT);
% Add the makers to the pelvis feild, within the pelvis frame
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).RASI);
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).LASI);
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).MASI);
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).RPSI);
```

Appendix B (Continued)

```
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).LPSI);
ADL.(subject).(name).Pelvis = addPoint2(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).MPSI);
% Transform the torso marker into the pelvis frame and add to the
% Pelvis structure
ADL.(subject).(name).Pelvis = addDistalPoint(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).T1);
ADL.(subject).(name).Pelvis = addDistalPoint(ADL.(subject).(name).Pelvis,
ADL.(subject).(name).CLAV);

% Compile the static marker position, and all of the motion trials
if strcmpi(name,'static')
    ADL.(subject).sPelvis = mean(ADL.(subject).(name).Pelvis.DistalPoint);
end

end % File loop end

% Calculate the static positions relative to the torso joint center.
for j=1:size(ADL.(subject).sPelvis,3)
    ADL.(subject).sPelvis(:,j) = ADL.(subject).sPelvis(:,j) -
ADL.(subject).TorsoCenter;
end

if 0 % isfield(ADL.(subject).Static, 'T10')
    for j = 1:size(ADL.(subject).Static.RASI,1);
        ADL.(subject).Static.TorsoJC(j,:) =
(ADL.(subject).Static.Pelvis.HT(1:3,1:4,j)*[ADL.(subject).TorsoCenter';1]);
    end

    % Calculate virtual torso marker (origin) and create torso segment
    ADL.(subject).Static.MTOR =
(ADL.(subject).Static.T1+ADL.(subject).Static.CLAV)/2;
    ADL.(subject).Static.Torso = createSegment(ADL.(subject).Static.TorsoJC,
(ADL.(subject).Static.MTOR-ADL.(subject).Static.TorsoJC), (ADL.(subject).Static.T1-
ADL.(subject).Static.CLAV), 'yzx');
    % Add markers to the torso segment
    ADL.(subject).Static.Torso = addPoint2(ADL.(subject).Static.Torso,
ADL.(subject).Static.T1);
    ADL.(subject).Static.Torso = addPoint2(ADL.(subject).Static.Torso,
ADL.(subject).Static.T10);
    ADL.(subject).Static.Torso = addPoint2(ADL.(subject).Static.Torso,
ADL.(subject).Static.CLAV);
```

Appendix B (Continued)

```
ADL.(subject).Static.Torso = addPoint2(ADL.(subject).Static.Torso,  
ADL.(subject).Static.TorsoJC);
```

```
ADL.(subject).TX(:, :) = nanmean(ADL.(subject).Static.Torso.Point);
```

```
for i=1:ADL.(subject).nfiles;  
    name = removewhite(ADL.(subject).filenames(i,:));  
    if isfield(ADL.(subject).(name), 'T10') && isfield(ADL.(subject).(name), 'LBAK')  
        for j = 1:size(ADL.(subject).(name).RASI,1)  
            ADL.(subject).(name).TorsoJC(j,:) =  
(ADL.(subject).(name).Pelvis.HT(1:3,1:4,j)*[ADL.(subject).TorsoCenter';1]);  
        end  
        % Reconstruct any missing points on the torso (requires at least 3  
        % pelvis markers).  
        [ADL.(subject).(name).T1, ADL.(subject).(name).T10,  
ADL.(subject).(name).CLAV] = ...  
        clusterReconstruct(ADL.(subject).TX, ADL.(subject).(name).T1,  
ADL.(subject).(name).T10, ADL.(subject).(name).CLAV, ...  
ADL.(subject).(name).TorsoJC);  
    end  
end  
end
```

```
% Create Segments for the ADL tasks
```

```
ADL =  
autoADLSegments(ADL, 'Pelvis', 'RTorso', 'RShoulder', 'yzx', {'CLAV', 'T1'}, {'RSHOP', 'RS  
HOA'});  
ADL =  
autoADLSegments(ADL, 'Pelvis', 'Torso', 'LShoulder', 'yzx', {'CLAV', 'T1'}, {'LSHOP', 'LSH  
OA'});
```

```
ADL =  
autoADLSegments(ADL, 'RTorso', 'RShoulder', 'RUpperArm', 'zyx', {'RSHOP', 'RSHOA'}, {'  
RELB', 'RELBM'});
```

```
ADL =  
autoADLSegments(ADL, 'Torso', 'LShoulder', 'LUpperArm', 'zyx', {'LSHOP', 'LSHOA'}, {'L  
ELB', 'LELBM'});
```

```
ADL =  
autoADLSegments(ADL, 'RShoulder', 'RUpperArm', 'RForearm', 'zxy', {'RELB', 'RELBM'},  
{'RWRB', 'RWRA'});
```

```
ADL = autoADLSegments(ADL, 'LShoulder', 'LUpperArm', 'LForearm', 'zxy', {'LELBM',  
'LELB'}, {'LWRB', 'LWRA'});
```

Appendix B (Continued)

```
ADL =
autoADLSegments(ADL,'RUpperArm','RForearm','RHand','zyx',{ 'RWRB','RWRA'},{ 'R
FIN','RFIN'});
ADL =
autoADLSegments(ADL,'LUpperArm','LForearm','LHand','zyx',{ 'LWRB','LWRA'},{ 'LF
IN','LFIN'});

ADL = autoADLSegments(ADL,'RForearm','RHand',[],'zyx',{ 'RWRB','RWRA','RFIN'});
ADL = autoADLSegments(ADL,'LForearm','LHand',[],'zyx',{ 'LWRB','LWRA','LFIN'});

% Find the joint angles for all ADL tasks
ADL = autoFindTheta(ADL);

Test = [];
Train = [];

[RDHmatrix, LDHmatrix, RUpperBody, LUpperBody] = ...
    createRobot(ADL.(subject).TorsoCenter, ...
        ADL.(subject).RShoulderCenter, ...
        ADL.(subject).RUpperArmCenter, ...
        ADL.(subject).RForearmCenter, ...
        ADL.(subject).RHandCenter, ...
        ADL.(subject).LShoulderCenter, ...
        ADL.(subject).LUpperArmCenter, ...
        ADL.(subject).LForearmCenter, ...
        ADL.(subject).LHandCenter);

% Create theta in robot terms for all ADLs and ROM tasks.
subject = char(subjects(s,:));

for i=1:ADL.(subject).nfiles
    name = removewhite(ADL.(subject).filenames(i,:));
    if ~(strcmpi(name, 'static'))
        RTheta = ADL.(subject).(name).Theta(:,1:14);
        LTheta = [ADL.(subject).(name).Theta(:,1:3),ADL.(subject).(name).Theta(:,16:26)];
        for j = 1:size(RTheta,1)
            Train.(subject).(name).RTheta(j,:) = RTheta(j,:)+RDHmatrix(:,3)';
            Train.(subject).(name).LTheta(j,:) = LTheta(j,:)+LDHmatrix(:,3)';
        end
    end
end % Files

for i=1:ROM.(subject).nfiles
```


Appendix B (Continued)

```
name = removewhite(ROM.(subject).filenames(i,:));
if ~(strcmpi(name, 'static'))
    RTheta = ROM.(subject).(name).Theta(:,1:14);
    LTheta =
[ROM.(subject).(name).Theta(:,1:3),ROM.(subject).(name).Theta(:,16:26)];
    for j = 1:size(RTheta,1)
        Test.(subject).(name).RTheta(j,:) = RTheta(j,:)+RDHmatrix(:,3)';
        Test.(subject).(name).LTheta(j,:) = LTheta(j,:)+LDHmatrix(:,3)';
    end
end
end % Files

% Claculate clinical joint angles and range of motion.
[ROM, ADL] = ROMtest(ROM, ADL, subject);

% Set the directory back to /Subjects
cd .. % /RHBM/Subjects/subject
cd .. % /RHBM/Subjects

% Add robot object to Train structure
Train.(subject).RUpperBody = RUpperBody;
Train.(subject).LUpperBody = LUpperBody;

% Save data for training and record.
save([subject,'UpperBodyModel'], '-struct', 'Train');
save([subject,'Data']);

% Set the directory back to /RHBM
cd .. % /RHBM

end % End subject loop
```

B.2 SubFunctions\removewhite.m

```
% White Space Remover
function string2 = removewhite(string1)
spacemat = isspace(string1);
i = 1;
while i<=size(string1,2)
    if (spacemat(i)==1)
        string1(i) = [];
        spacemat(i) = [];
    else
        i=i+1;
    end
end
end
```

Appendix B (Continued)

```
string2 = string1;

if (size(string1,2)>=4)
    if strcmp(string1(1,(size(string1,2)-3):(size(string1,2))),'.c3d')
        string2 = string1(1,1:(size(string1,2)-4));
    end
end

end
```

B.3 SubFunctions\WMAfilter.m

```
% Weighted moving average filter
function [xfil] = WMAfilter(n, x)
% Moving average filter
% x = Array of points to be filtered.
% n = Width of the filter.
% xfil = Filtered array of input array x.
% Define weighting array

WA = [];
for i = 1:n
    if i<=floor(n/2)
        WA = [WA,i];
    else
        WA = [WA,n-i+1];
    end
end
WA = WA/sum(WA);

xfil = zeros(size(x));
% defining a zero matrix, of the same size as array x.

xnew = x;
for i=1:floor(n/2)
    xnew = cat(1, x(i+1, :, :), xnew);
    xnew = cat(1, xnew, x(size(x,1)-i, :, :));
end

for i=1:size(x,1)
    % iterations, from 1 to number of rows of the array x.
    for j = 1:n
        xfil(i, :, :) = xfil(i, :, :) + WA(j)*xnew((i+j-1), :, :);
    end
end

end
```

Appendix B (Continued)

```
if all(all(isnan(xfil)))
    disp('To Many NaNs to filter')
    xfil = x;
end
%repeat until size (x,1) has been reached
```

B.4 SubFunctions\createSegment.m

```
classdef createSegment
% Creates a segment frame for a set of marker positions using an origin
% point, two defining lines and an order.
% The Segment Frame is centered at the Origin.
% The first axis lies along the first defining line.
% The second axis is the cross product of the first and second defining
% lines.
% The thrid axis is the cross of the two first axes.
    properties
        Origin;
        Xaxis;
        Yaxis;
        Zaxis;
        HT;
        Point = [];
        DistalPoint = [];
    end
    methods
        function seg = createSegment(origin, Line1, Line2, Order, OffsetT)
            if(nargin <= 2)
                'Segment must contain at least an origin and 2 defining lines'
            end
            seg.Origin = origin;

            e2preunit = cross(Line1, Line2);
            e3preunit = cross(Line1, e2preunit);

            e1 = vec2unit(Line1);
            e2 = vec2unit(e2preunit);
            e3 = vec2unit(e3preunit);
            if ((nargin == 3)||strcmpi(Order, 'xyz'))
                seg.Xaxis = e1;
                seg.Yaxis = e2;
                seg.Zaxis = e3;
            elseif strcmpi(Order, 'xzy')
                seg.Xaxis = e1;
                seg.Yaxis = -e3;
                seg.Zaxis = e2;
            end
        end
    end
end
```

Appendix B (Continued)

```
elseif strcmpi(Order, 'yxz')
    seg.Xaxis = e2;
    seg.Yaxis = e1;
    seg.Zaxis = -e3;
elseif strcmpi(Order, 'yzx')
    seg.Xaxis = e3;
    seg.Yaxis = e1;
    seg.Zaxis = e2;
elseif strcmpi(Order, 'zxy')
    seg.Xaxis = e2;
    seg.Yaxis = e3;
    seg.Zaxis = e1;
elseif strcmpi(Order, 'zyx')
    seg.Xaxis = -e3;
    seg.Yaxis = e2;
    seg.Zaxis = e1;
end
for i=1:size(seg.Xaxis,1)
    seg.HT(:,i) = cat(2, seg.Xaxis(i,:)', seg.Yaxis(i,:)', seg.Zaxis(i,:)', origin(i,:));
end
seg.HT(4,4,:) = 1;

if(nargin >= 5)
    for i=1:size(seg.HT,3)
        seg.HT(:,i) = seg.HT(:,i)*OffsetT^-1;
    end
end

end % Function Create Segment

end % Methods
end % Class Def
```

B.5 SubFunctions\addPoint2.m

```
% Adds a point to the current segment
% RHBM 2/7/2011 Derek J. Lura
function seg = addPoint2(seg, point)
newpoint = 1;
point = point(:,1:3);
segPoint(:,i) = point - seg.Origin;
for i=1:size(point,1)
    segPoint(i,:) = [dot(segPoint(i,:),seg.Xaxis(i,:)), dot(segPoint(i,:),seg.Yaxis(i,:)),
dot(segPoint(i,:),seg.Zaxis(i,:))];
end
```

Appendix B (Continued)

```
PN = 1;
if size(seg.Point,1)>0;
    PN = size(seg.Point,3) + 1;
end

if newpoint
    seg.Point(:, :, PN) = segPoint;
end

end
```

B.6 SubFunctions\addDistalPoint.m

```
% Add a distal point to the current segment
% distal points are used for functional joint center estimation of the distal segment.
% RHBM 2/7/2011 Derek J. Lura
function seg = addDistalPoint(seg, point)
newpoint = 1;
point = point(:, 1:3);
segPoint(:, :) = point - seg.Origin;
for i=1:size(point,1)
    segPoint(i,:) = [dot(segPoint(i,:),seg.Xaxis(i,:)), dot(segPoint(i,:),seg.Yaxis(i:)),
dot(segPoint(i,:),seg.Zaxis(i,:))];
end

PN = 1;
if size(seg.DistalPoint,1)>0;
    PN = size(seg.DistalPoint,3) + 1;
end

if newpoint
    seg.DistalPoint(:, :, PN) = segPoint;
end

end
```

B.7 SubFunctions\reconstruct.m

```
% Single maker droupout reconstruction algorithm
% RHBM 2/7/2011
function [Pta, Ptb, Ptc, Ptd] = reconstruct(PtA, PtB, PtC, PtD)

function HT = createSeg(origin, Line1, Line2)
    e2preunit = cross(Line1, Line2);
    e3preunit = cross(Line1, e2preunit);
    Xaxis = vec2unit(Line1);
    Yaxis = vec2unit(e2preunit);
```

Appendix B (Continued)

```
Zaxis = vec2unit(e3preunit);
for i=1:size(Xaxis,1)
    HT(:,i) = cat(2, Xaxis(i,:), Yaxis(i,:), Zaxis(i,:), origin(i,:));
end
HT(4,4,:) = 1;
end

function avgPoint = findAvg(HT,Pt)
tempOrigin(:,i) = HT(1:3,4,:);
segPoint = Pt - tempOrigin';
for i=1:size(Pt,1)
    segPoint(i,:) = [dot(segPoint(i,:),HT(1:3,1,i)), dot(segPoint(i,:),HT(1:3,2,i)),
dot(segPoint(i,:),HT(1:3,3,i))];
end
segPoint(any(isnan(segPoint),2),:) = [];
avgPoint = mean(segPoint);
end

HT_ABC = createSeg(PtA, PtB-PtA, PtC-PtA);
HT_BCD = createSeg(PtB, PtC-PtB, PtD-PtB);
HT_CDA = createSeg(PtC, PtD-PtC, PtA-PtC);
HT_DAB = createSeg(PtD, PtA-PtD, PtB-PtD);

Avg_PtA = findAvg(HT_BCD,PtA);
Avg_PtB = findAvg(HT_CDA,PtB);
Avg_PtC = findAvg(HT_DAB,PtC);
Avg_PtD = findAvg(HT_ABC,PtD);

%function Pt1 = bestPoint(Pt1, Pt2, Pt3)
for j=1:size(PtA,1)
    if (isnan(PtA(j,1)))
        PtA(j,:) = (HT_BCD(1:3,1:3,j)*Avg_PtA'+HT_BCD(1:3,4,j));
    elseif (isnan(PtB(j,1)))
        PtB(j,:) = (HT_CDA(1:3,1:3,j)*Avg_PtB'+HT_CDA(1:3,4,j));
    elseif (isnan(PtC(j,1)))
        PtC(j,:) = (HT_DAB(1:3,1:3,j)*Avg_PtC'+HT_DAB(1:3,4,j));
    elseif (isnan(PtD(j,1)))
        PtD(j,:) = (HT_ABC(1:3,1:3,j)*Avg_PtD'+HT_ABC(1:3,4,j));
    end
end

Pta = PtA;
Ptb = PtB;
Ptc = PtC;
```

Appendix B (Continued)

```
Ptd = PtD;  
end
```

B.8 SubFunctions\clusterReconstruct.m

```
% Marker cluster based recontruction algorithm
```

```
% RHBM 2/7/2011
```

```
function [Pta, Ptb, Ptc, Ptd] = clusterReconstruct(X, varargin)
```

```
%function Pt1 = bestPoint(Pt1, Pt2, Pt3
```

```
for j=1:size(varargin{1},1)
```

```
    y = [];
```

```
    xt = [];
```

```
    Y = [];
```

```
    Xt = [];
```

```
    for i=1:nargin-1
```

```
        if ~(isnan(varargin{i}(j,1)))
```

```
            y = [y, varargin{i}(j,:)'];
```

```
            xt = [xt, X(:,i)];
```

```
        end
```

```
    end
```

```
    yb = mean(y,2);
```

```
    xb = mean(xt,2);
```

```
    for i=1:size(y,2)
```

```
        Y(:,i) = y(:,i)-yb;
```

```
        Xt(:,i) = xt(:,i)-xb;
```

```
    end
```

```
    Z = Y*Xt';
```

```
    [U,S,V] = svd(Z);
```

```
    R = U*diag([1,1,det(U*V')])*V';
```

```
    p = mean((y - R*xt),2);
```

```
    for i=1:4
```

```
        if (isnan(varargin{i}(j,1)))
```

```
            varargin{i}(j,:) = R*X(:,i) + p;
```

```
        end
```

```
    end
```

```
end
```

```
Pta = varargin{1};
```

```
Ptb = varargin{2};
```

```
Ptc = varargin{3};
```

```
Ptd = varargin{4};
```

```
End
```

Appendix B (Continued)

B.9 SubFunctions\MLOptim.m

% Gradient Based Functional Joint Center Method

% RHBM 2/7/2011

function center = MLOptim(P, G)

P=floor(P);

aStr = ['X1'; 'X2'; 'X3'];

for n = 1:size(P,3)

 j=1;

 X(:,j) = P(:,j);

 while j<size(X,1)

 if all(X(j+1,:,j)==X(j,:,j))

 X(j+1,:,j) = [];

 else

 j = j+1;

 end

 end

 X = X+.5;

 if n == 1

 New.X1 = X;

 elseif n == 2

 New.X2 = X;

 elseif n == 3;

 New.X3 = X;

 end

 X = [];

end

% sizeP = size(P);

% for i=1:size(P,3)

% weight(:,i) = weightPoints(P(:,j),i);

% end

% for all points in grid 2 x 2 x 2 (with resolution res).

function Cost = costfun(iv,jv,kv)

 Cost = 0;

 SizeX = 0;

 for n = 1:size(P,3)

 X = New.(aStr(n,:));

 SizeX = SizeX+size(X,1);

 % Calculate the average distance to the point iv, jv, kv

 % Sum of (Distance to point) / Number of Points

 Ravg = sum(sqrt(sum([X(:,1)-iv, X(:,2)-jv, X(:,3)-kv]'.^2)))/size(X,1);

 % Calculate the square of the difference between the

Appendix B (Continued)

```
% average and current distance Ravg - RDist.
% Sum of (point weight*(Distance to point - Ravg)^2)
% size(sqrt(sum([P(:,1,n)-iv,P(:,2,n)-jv,P(:,3,n)-kv]'.^2)) - Ravg)
% size(weight(:,n))
% Cost = Cost + sum( weight(:,n)'*(sqrt(sum([P(:,1,n)-iv,P(:,2,n)-jv,P(:,3,n)-
kv]'.^2)) - Ravg).^2);
Cost = Cost + sum((sqrt(sum([X(:,1)-iv,X(:,2)-jv,X(:,3)-kv]'.^2)) - Ravg).^2);
end

Cost = Cost/SizeX;
% Cost is equal to the average varance of radius
end

% Set F = my function
f = @costfun;

% Define initial guess
x0 = G';

% Define Bounds
lb = [-500; -500; -500];
ub = [500; 500; 500];

% Start with the default options
options = optimset;
% Modify options setting
options = optimset(options,'Display','off');
options = optimset(options,'MaxIter', 300);
options = optimset(options,'LargeScale','off');
options = optimset(options,'Algorithm','active-set');
options = optimset(options,'PlotFcns',{ @optimplotfval });
% [centert] = fmincon(@(x)f(x(1),x(2),x(3)),x0,[],[],[],[],lb,ub,[],options);
[centert] = fminunc(@(x)f(x(1),x(2),x(3)),x0,options);

center = centert';
end
```

B.10 SubFunctions\autoSegment.m

```
% Calculate the segments for the RoM tasks
% RHBM 2/7/2011 Derek J. Lura
function [targets, centers] = autoSegment(targets, ProSegment, Segment, DisSegment,
Order, SegPoint, DisPoint)
% targets
% filenames
% SegmentCenter
```

Appendix B (Continued)

```
% Static
% Segment
% Njoints
% Set the negative notation for left handed segments
if Segment(1) == 'L'
    L = -1;
else
    L = 1;
end

plots = 0;
centers = [];

for s=1:size(fieldnames(targets),1);
    subjects = fieldnames(targets);
    subject = char(subjects(s));
    targets.(subject).([Segment,'Compiled']) = [];
for i=1:targets.(subject).nfiles
    name = removewhite(targets.(subject).filenames(i,:));
    Nsamples = size(targets.(subject).(name).LPSI,1);

    % Create any necessary virtual points based on pure marker positions.
    if (strcmp(Segment,'RTorso')||strcmp(Segment,'Torso'))
        for j = 1:Nsamples
            targets.(subject).(name).([Segment,'JC'])(j,:) =
(targets.(subject).(name).(ProSegment).HT(1:3,1:4,j)*[targets.(subject).([Segment,'Center
']);1]);
        end
    end

    if (strcmp(Segment,'RHand')||strcmp(Segment,'LHand'))
        AVG = targets.(subject).(name).(char(SegPoint(3)));
        targets.(subject).(name).(Segment) =
createSegment(targets.(subject).(name).([Segment,'JC']), (AVG-
targets.(subject).(name).([Segment,'JC'])), targets.(subject).(name).(ProSegment).Xaxis,
Order);
    else
        AVG =
(targets.(subject).(name).(char(SegPoint(1)))+targets.(subject).(name).(char(SegPoint(2)
)))/2;
        targets.(subject).(name).(Segment) =
createSegment(targets.(subject).(name).([Segment,'JC']), (AVG-
targets.(subject).(name).([Segment,'JC'])),
```

Appendix B (Continued)

```
L*(targets.(subject).(name).(char(SegPoint(2))))-
targets.(subject).(name).(char(SegPoint(1))))), Order);
end

    targets.(subject).(name).(Segment) = addPoint2(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(SegPoint(1)))));
    targets.(subject).(name).(Segment) = addPoint2(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(SegPoint(2)))));

    if nargin == 7
        targets.(subject).(name).(Segment) =
addDistalPoint(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(DisPoint(1)))));
        targets.(subject).(name).(Segment) =
addDistalPoint(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(DisPoint(2)))));
    end

    if strcmpi(name,'static')
        targets.(subject).(['s',Segment]) =
nanmean(targets.(subject).(name).(Segment).Point);
        CenterEst = (nanmean(targets.(subject).(name).(Segment).Point(:,1)) +
nanmean(targets.(subject).(name).(Segment).Point(:,2)))/2
        SegWidth = nanmean(targets.(subject).(name).(Segment).Point(:,1)) -
nanmean(targets.(subject).(name).(Segment).Point(:,2));
    elseif nargin == 7
        targets.(subject).([Segment,'Compiled']) = cat(1,
targets.(subject).([Segment,'Compiled']), targets.(subject).(name).(Segment).DistalPoint);
    end

end %Files

if nargin == 7

targets.(subject).([Segment,'Compiled'])(any(any(isnan(targets.(subject).([Segment,'Com
piled'])),3),2),:,:) = [];

    targets.(subject).([DisSegment,'Center']) =
MLOptim(targets.(subject).([Segment,'Compiled'])(:,:,:),CenterEst);
    set(gcf,'name',[DisSegment,' ',subject]);
    saveas(gcf,['C:\Documents and
Settings\dlura\Desktop\RHBM\Figures\',DisSegment,subject,'.fig']);
```

Appendix B (Continued)

```
if sum((targets.(subject).([DisSegment,'Center']) -
CenterEst).^2)>3*sum(SegWidth.^2)
    %input(['Bad FJC for ',DisSegment,' press enter to continue with Static Joint Center
Esitimation']);
    disp(['Bad FJC for ',DisSegment,' press to continuing with Static Joint Center
Esitimation']);
    targets.(subject).([DisSegment,'Center']) = CenterEst;
end
centers = [centers; targets.(subject).([DisSegment,'Center'])];

if plots
    figure('name',[DisSegment, subject])
    hold off
    plot3(targets.(subject).([Segment,'Compiled'])(:,1,1),
targets.(subject).([Segment,'Compiled'])(:,2,1),
targets.(subject).([Segment,'Compiled'])(:,3,1), 'Color', [0,0.5,0])
    hold on
    plot3(targets.(subject).([Segment,'Compiled'])(:,1,2),
targets.(subject).([Segment,'Compiled'])(:,2,2),
targets.(subject).([Segment,'Compiled'])(:,3,2))
    axis equal
    plot3(targets.(subject).([DisSegment,'Center'])(1),
targets.(subject).([DisSegment,'Center'])(2), targets.(subject).([DisSegment,'Center'])(3),
'b+', 'LineWidth',2, 'MarkerSize',10)
    end
end

end %Subjets

if ~(strcmp(Segment,'RHand')||strcmp(Segment,'LHand'))
    %Redefine Segments with distal joint centers.
    for s=1:size(fieldnames(targets),1);
        subjects = fieldnames(targets);
        subject = char(subjects(s));
        if ~(strcmp(Segment,'RTorso')||strcmp(Segment,'Torso'))
            targets.(subject).([Segment,'Compiled']) = [];
        end
    end
    for i=1:targets.(subject).nfiles
        name = removewhite(targets.(subject).filenames(i,:));
        Nsamples = size(targets.(subject).(name).LPSI,1);

        for j = 1:Nsamples
```

Appendix B (Continued)

```
targets.(subject).(name).([DisSegment,'JC'])(j,:) =
(targets.(subject).(name).(Segment).HT(1:3,1:4,j)*[targets.(char(subjects(s))).([DisSegme
nt,'Center']');1]);
end

if ~(strcmp(Segment,'RTorso')||strcmp(Segment,'Torso'))
targets.(subject).(name).(Segment) =
createSegment(targets.(subject).(name).([Segment,'JC']),
(targets.(subject).(name).([DisSegment,'JC'])-targets.(subject).(name).([Segment,'JC'])),
L*(targets.(subject).(name).(char(SegPoint(2)))-
targets.(subject).(name).(char(SegPoint(1))))), Order);

targets.(subject).(name).(Segment) = addPoint2(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(SegPoint(1))));
targets.(subject).(name).(Segment) = addPoint2(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(SegPoint(2))));

targets.(subject).(name).(Segment) =
addDistalPoint(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(DisPoint(1))));
targets.(subject).(name).(Segment) =
addDistalPoint(targets.(subject).(name).(Segment),
targets.(subject).(name).(char(DisPoint(2))));
targets.(subject).(name).(Segment) =
addDistalPoint(targets.(subject).(name).(Segment),
targets.(subject).(name).([DisSegment,'JC']));

if strcmpi(name,'static')
targets.(subject).(['s',Segment]) = mean(targets.(subject).(name).(Segment).Point);
elseif nargin == 7
targets.(subject).([Segment,'Compiled']) = cat(1,
targets.(subject).([Segment,'Compiled']), targets.(subject).(name).(Segment).DistalPoint);
end
end

end %Files

end %Subjets

end %If not Torso or Hand

end %Function
```

B.11 SubFunctions\autoFindTheta.m

%Automatically find the joint angles "Theta" for the given data structure.

Appendix B (Continued)

% Finds joint angles for all points in all trials for all subjects.
% Designed to be used with the RHBM set of functions, will not take a
% generic structure.

%

% Derek J. Lura, University of South Florida, 2011

function targets = autoFindTheta(targets)

```
for s=1:size(fieldnames(targets),1);
    subjects = fieldnames(targets);
    subject = char(subjects(s));
    for i=1:targets.(subject).nfiles
        name = removewhite(targets.(subject).filenames(i,:));
        Nsamples = size(targets.(subject).(name).LSHOA,1);

        Theta = zeros(Nsamples,27);
        for j=1:Nsamples
            R_Torso = targets.(subject).(name).Pelvis.HT(1:3,1:3,j)^-
1*targets.(subject).(name).Torso.HT(1:3,1:3,j);

            R_RShoulder = targets.(subject).(name).RTorso.HT(1:3,1:3,j)^-
1*targets.(subject).(name).RShoulder.HT(1:3,1:3,j);
            R_RUpperArm = targets.(subject).(name).RShoulder.HT(1:3,1:3,j)^-
1*targets.(subject).(name).RUpperArm.HT(1:3,1:3,j);
            R_RForearm = targets.(subject).(name).RUpperArm.HT(1:3,1:3,j)^-
1*targets.(subject).(name).RForearm.HT(1:3,1:3,j);
            R_RHand = targets.(subject).(name).RForearm.HT(1:3,1:3,j)^-
1*targets.(subject).(name).RHand.HT(1:3,1:3,j);

            R_LShoulder = targets.(subject).(name).Torso.HT(1:3,1:3,j)^-
1*targets.(subject).(name).LShoulder.HT(1:3,1:3,j);
            R_LUpperArm = targets.(subject).(name).LShoulder.HT(1:3,1:3,j)^-
1*targets.(subject).(name).LUpperArm.HT(1:3,1:3,j);
            R_LForearm = targets.(subject).(name).LUpperArm.HT(1:3,1:3,j)^-
1*targets.(subject).(name).LForearm.HT(1:3,1:3,j);
            R_LHand = targets.(subject).(name).LForearm.HT(1:3,1:3,j)^-
1*targets.(subject).(name).LHand.HT(1:3,1:3,j);

            Theta(j,1:3) = findTheta('zxy', R_Torso);
            Theta(j,4:6) = findTheta('yxz', R_RShoulder);
            Theta(j,7:9) = findTheta('yxz', R_RUpperArm);
            Theta(j,10:12) = findTheta('yxz', R_RForearm);
            Theta(j,13:15) = findTheta('xyz', R_RHand);

            Theta(j,16:18) = findTheta('yxz', R_LShoulder);
```

Appendix B (Continued)

```
Theta(j,19:21) = findTheta('yxz', R_LUpperArm);
Theta(j,22:24) = findTheta('yxz', R_LForearm);
Theta(j,25:27) = findTheta('xyz', R_LHand);

if (Theta(j,16)<=0)
    Theta(j,16) = Theta(j,16)+2*pi;
end

end %samples
targets.(subject).(name).Theta = Theta;

end %Trials
end %Subjects

end

B.12 SubFunctions\findTheta.m
% Calculates the euler angles given a rotation order and a rotation matrix.
% Derek Lura, University of South Florida 2011
function theta = findTheta(order, R)

Ro = R;
thetaM = zeros(size(Ro,3),3);

for i = 1:size(Ro,3)
    R = [];
    R(:,i) = Ro(:,i);

    if strcmp(order,'zxy')
        x = asin(R(3,2));
        y = acos(R(3,3)/cos(x));
        y2 = asin(-R(3,1)/cos(x));
        z = acos(R(2,2)/cos(x));
        z2 = asin(-R(1,2)/cos(x));

        if y2<=0
            y= -y;
        end
        if z2<=0
            z= -z;
        end

        Rzxy = [ cos(z)*cos(y)-sin(z)*sin(x)*sin(y),           -sin(z)*cos(x),
                cos(z)*sin(y)+sin(z)*sin(x)*cos(y);
```

Appendix B (Continued)

```
sin(z)*cos(y)+cos(z)*sin(x)*sin(y), cos(z)*cos(x), sin(z)*sin(y)-  
cos(z)*sin(x)*cos(y);  
-cos(x)*sin(y), sin(x), cos(x)*cos(y)];
```

```
test = R-Rzxy;  
if sum(sum(test.^2))>=0.001  
    disp('Error in angle calculation zxy')  
end  
theta = real([z, x, y]);  
  
elseif strcmp(order, 'yxz')  
    x = asin(-R(2,3)); %returns x from -pi/2 to pi/2  
    y = acos(R(3,3)/cos(x)); %returns y from 0 to pi  
    y2 = asin(R(1,3)/cos(x)); %returns y from -pi/2 to pi/2  
    z = acos(R(2,2)/cos(x)); %returns z from 0 to pi  
    z2 = asin(R(2,1)/cos(x)); %returns z from -pi/2 to pi/2  
  
    if y2<=0  
        y= -y;  
    end  
    if z2<=0  
        z= -z;  
    end  
  
    y = atan(R(1,3)/(R(3,3)+0.001*(R(3,3)^-1)));  
    z = atan(R(2,1)/(R(2,2)+0.001*(R(2,2)^-1)));  
  
    y = atan2(R(1,3),R(3,3));  
    z = atan2(R(2,1),R(2,2));  
    % if R(3,3)<0  
    % if x>0;  
    % x = x+2*(pi/2 - x);  
    % else  
    % x = x+2*(-pi/2 - x);  
    % end  
    % end
```

```
Ryxz = [ sin(z)*sin(x)*sin(y)+cos(z)*cos(y), cos(z)*sin(x)*sin(y)-sin(z)*cos(y),  
cos(x)*sin(y);  
sin(z)*cos(x), cos(z)*cos(x), -sin(x);  
sin(z)*sin(x)*cos(y)-cos(z)*sin(y), cos(z)*sin(x)*cos(y)+sin(z)*sin(y),  
cos(x)*cos(y)];
```

```
test = R-Ryxz;
```


Appendix B (Continued)

```
if sum(sum(test.^2))>=0.001
    disp(['Error in angle calculation xyz', num2str(sum(sum(test.^2))])
end
theta = real([y,x,z]);

elseif strcmp(order,'xyz')
    y = asin(R(1,3)); %returns x from -pi/2 to pi/2
    x = acos(R(3,3)/cos(y)); %returns x from 0 to pi
    x2 = asin(-R(2,3)/cos(y)); %returns x from -pi/2 to pi/2
    z = acos(R(1,1)/cos(y)); %returns z from 0 to pi
    z2 = asin(-R(1,2)/cos(y)); %returns z from -pi/2 to pi/2

    if x2<=0
        x= -x;
    end
    if z2<=0
        z= -z;
    end

    Rxyz = [
        cos(y)*cos(z),          -cos(y)*sin(z),
        sin(y)*sin(x)*sin(y)*cos(z)+cos(x)*sin(z), -sin(x)*sin(y)*sin(z)+cos(x)*cos(z),
        -sin(x)*cos(y);
        -cos(x)*sin(y)*cos(z)+sin(x)*sin(z), cos(x)*sin(y)*sin(z)+sin(x)*cos(z),
        cos(x)*cos(y)];
    test = R-Rxyz;
    if sum(sum(test.^2))>=0.001
        disp('Error in angle calculation xyz')
    end
    theta = real([x,y,z]);

elseif strcmp(order,'zyx')
    y = asin(-R(3,1)); %returns x from -pi/2 to pi/2
    x = acos(R(3,3)/cos(y)); %returns x from 0 to pi
    x2 = asin(R(3,2)/cos(y)); %returns x from -pi/2 to pi/2
    z = acos(R(1,1)/cos(y)); %returns z from 0 to pi
    z2 = asin(R(2,1)/cos(y)); %returns z from -pi/2 to pi/2

    if x2<=0
        x= -x;
    end
    if z2<=0
        z= -z;
    end
end
```

Appendix B (Continued)

```
Rzyx = [ cos(z)*cos(y), -sin(z)*cos(x)+cos(z)*sin(y)*sin(x),  
sin(z)*sin(x)+cos(z)*sin(y)*cos(x);  
sin(z)*cos(y), cos(z)*cos(x)+sin(z)*sin(y)*sin(x), -  
cos(z)*sin(x)+sin(z)*sin(y)*cos(x);  
-sin(y), cos(y)*sin(x), cos(y)*cos(x)];  
test = R-Rzyx;  
if sum(sum(test.^2))>=0.001  
disp(['Error in angle calculation zyx', num2str(sum(sum(test.^2))])]  
end  
theta = real([z,y,x]);  
  
elseif strcmp(order,'xzy')  
z = asin(-R(1,2)); %returns x from -pi/2 to pi/2  
x = acos(R(2,2)/cos(z)); %returns x from 0 to pi  
x2 = asin(R(3,2)/cos(z)); %returns x from -pi/2 to pi/2  
y = acos(R(1,1)/cos(z)); %returns z from 0 to pi  
y2 = asin(R(1,3)/cos(y)); %returns z from -pi/2 to pi/2  
  
if x2<=0  
x= -x;  
end  
if y2<=0  
z= -z;  
end  
  
Rxzy = [ cos(z)*cos(y), -sin(z),  
cos(z)*sin(y); sin(z)*cos(x)*cos(y)+sin(x)*sin(y), cos(z)*cos(x),  
sin(z)*sin(y)*cos(x)-cos(y)*sin(x); sin(z)*sin(x)*cos(y)-cos(x)*sin(y), cos(z)*sin(x),  
sin(z)*sin(y)*sin(x)+cos(y)*cos(x)];  
test = R-Rxzy;  
if sum(sum(test.^2))>=0.001  
disp('Error in angle calculation Rxzy')  
end  
theta = real([x,z,y]);  
  
elseif strcmp(order,'yzx')  
z = asin(R(2,1)); %returns x from -pi/2 to pi/2  
x = acos(R(2,2)/cos(z)); %returns x from 0 to pi  
x2 = asin(-R(2,3)/cos(z)); %returns x from -pi/2 to pi/2  
y = acos(R(1,1)/cos(z)); %returns z from 0 to pi  
y2 = asin(-R(3,1)/cos(z)); %returns z from -pi/2 to pi/2
```

Appendix B (Continued)

```
if x2<=0
    x= -x;
end
if y2<=0
    z= -z;
end

Ryzx = [ cos(z)*cos(y), -sin(z)*cos(x)*cos(y)+sin(x)*sin(y),
sin(z)*sin(x)*cos(y)+cos(x)*sin(y);
        sin(z),          cos(z)*cos(x),          -cos(z)*sin(x);
        -cos(z)*sin(y), sin(z)*sin(y)*cos(x)+cos(y)*sin(x), -
sin(z)*sin(y)*sin(x)+cos(y)*cos(x)];
test = R-Ryzx;
if sum(sum(test.^2))>=0.001
    disp('Error in angle calculation zyx')
end
theta = real([y,z,x]);

elseif strcmp(order,'zxz')
    z = asin(R(2,1)); %returns x from -pi/2 to pi/2
    x = acos(R(2,2)/cos(z)); %returns x from 0 to pi
    x2 = asin(-R(2,3)/cos(z)); %returns x from -pi/2 to pi/2
    y = acos(R(1,1)/cos(z)); %returns z from 0 to pi
    y2 = asin(-R(3,1)/cos(z)); %returns z from -pi/2 to pi/2

    if x2<=0
        x= -x;
    end
    if y2<=0
        z= -z;
    end

    Ryzx = [ cos(z)*cos(y), -sin(z)*cos(x)*cos(y)+sin(x)*sin(y),
sin(z)*sin(x)*cos(y)+cos(x)*sin(y);
            sin(z),          cos(z)*cos(x),          -cos(z)*sin(x);
            -cos(z)*sin(y), sin(z)*sin(y)*cos(x)+cos(y)*sin(x), -
sin(z)*sin(y)*sin(x)+cos(y)*cos(x)];
    test = R-Ryzx;
    if sum(sum(test.^2))>=0.001
        disp('Error in angle calculation zyx')
    end
    theta = real([y,z,x]);
end
```

Appendix B (Continued)

```
thetaM(i,:) = theta;
```

```
end
```

```
theta = thetaM;
```

```
end
```

B.13 SubFunctions\createRobot.m

```
% Joint Center to Denavit & Hartenberg Calculator
```

```
% RHBM 2/4/2011
```

```
function [RDHmatrix, LDHmatrix, RUpperBody, LUpperBody] = createRobot(TRJC,  
RSJC, RUAJC, RFJC, RHJC, LSJC, LUAJC, LFJC, LHJC)
```

```
% TRJC = Torso Joint Center
```

```
% RSJC = Right Shoulder Joint Center
```

```
% RUAJC = Right Upper Arm Joint Center
```

```
% RFJC = Right Forearm Joint Center
```

```
% RHJC = Right Hand Joint CenterD
```

```
% LSJC = Left Shoulder Joint Center
```

```
% LUAJC = Left Upper Arm Joint Center
```

```
% LFJC = Left Forearm Joint Center
```

```
% LHJC = Left Hand Joint Center
```

```
% DHmatrix(n,:) = [alpha A theta D], matrix of Denavit and Hartenberg Parameters, of  
DoF n.
```

```
RDHmatrix(1,:) = [0, 0, 0, 0]; % Torso Extension (Torso Z)
```

```
RDHmatrix(2,:) = [pi/2, 0, -pi/2, 0]; % Torso Lateral Flexion (Torso X)
```

```
RDHmatrix(3,:) = [-pi/2, 0, -pi/2-atan2(RSJC(3),RSJC(1)), 0]; % Torso Rotation  
(Torso Y)
```

```
RDHmatrix(4,:) = [0, sqrt(RSJC(1)^2+RSJC(3)^2), -atan2(RSJC(1),RSJC(3)), RSJC(2)];  
% RShoulder Abduction (RShoulder Y)
```

```
RDHmatrix(5,:) = [-pi/2, 0, -pi/2, 0]; % RShoulder Elivation (RShoulder X)
```

```
RDHmatrix(6,:) = [-pi/2, 0, -pi/2, RUAJC(3)]; % RShoulder Rotation (RShoulder Z)
```

```
RDHmatrix(7,:) = [-pi/2, RUAJC(1), -pi/2, RUAJC(2)]; % RUpperArm Flexion  
(Rupperarm Y)
```

```
RDHmatrix(8,:) = [-pi/2, 0, -pi/2, 0]; % RUpperArm Abduction (Rupperarm X)
```

```
RDHmatrix(9,:) = [-pi/2, 0, -pi/2, RFJC(3)]; % RUpperArm Rotation (RForearm Z)
```

```
RDHmatrix(10,:) = [-pi/2, RFJC(1), -pi/2, RFJC(2)]; % RForearm Flexion (RForearm  
Y)
```

```
RDHmatrix(11,:) = [-pi/2, 0, -pi/2, 0]; % RForearm Abduction (RForearm X)
```

```
RDHmatrix(12,:) = [-pi/2, 0, 0, RHJC(3)]; % RForearm Rotation (Rupperarm Z)
```

Appendix B (Continued)

```
RDHmatrix(13,:) = [pi/2, RHJC(1), pi/2, RHJC(2)]; % RHand Abduction (RHand X)
RDHmatrix(14,:) = [pi/2, 0, 0, 0]; % RHand Flexion (RHand Y)
```

```
% Left Side
```

```
LDHmatrix(1,:) = [0, 0, 0, 0]; % Torso Flexion (Torso Z)
LDHmatrix(2,:) = [pi/2, 0, -pi/2, 0]; % Torso Lateral Flexion (Torso X)
LDHmatrix(3,:) = [-pi/2, 0, -pi/2-atan2(LSJC(3),LSJC(1)), 0]; % Torso Rotation (Torso Y)
```

```
LDHmatrix(4,:) = [0, sqrt(LSJC(1)^2+LSJC(3)^2), pi-atan2(LSJC(1),LSJC(3)),
LSJC(2)]; % LShoulder Abduction (LShoulder Y)
```

```
if LDHmatrix(4,3)>pi
```

```
    LDHmatrix(4,3) = LDHmatrix(4,3)-2*pi;
```

```
end
```

```
LDHmatrix(5,:) = [pi/2, 0, pi/2, LUAJC(1)]; % LShoulder Elivation (LShoulder X)
LDHmatrix(6,:) = [-pi/2, 0, -pi/2, LUAJC(3)];
```

```
LDHmatrix(7,:) = [-pi/2, LUAJC(1), -pi/2, LUAJC(2)]; % LUpperArm Flexion (Lupperarm Y)
```

```
LDHmatrix(8,:) = [-pi/2, 0, -pi/2, 0]; % LUpperArm Abduction (Lupperarm X)
```

```
LDHmatrix(9,:) = [-pi/2, 0, -pi/2, LFJC(3)]; % LUpperArm Rotation (Lupperarm Z)
```

```
LDHmatrix(10,:) = [-pi/2, LFJC(1), -pi/2, LFJC(2)]; % LForearm Flexion (LForearm Y)
```

```
LDHmatrix(11,:) = [-pi/2, 0, -pi/2, 0]; % LForearm Abduction (LForearm X)
```

```
LDHmatrix(12,:) = [-pi/2, 0, 0, LHJC(3)]; % % LForearm Rotation (Lupperarm Z)
```

```
LDHmatrix(13,:) = [pi/2, LHJC(1), pi/2, LHJC(2)]; % LHand Flexion (LHand Y)
```

```
LDHmatrix(14,:) = [pi/2, 0, 0, 0]; % LHand Abduction (RHand X)
```

```
R1 = link(RDHmatrix(1:,:), 'mod');
R2 = link(RDHmatrix(2:,:), 'mod');
R3 = link(RDHmatrix(3:,:), 'mod');
R4 = link(RDHmatrix(4:,:), 'mod');
R5 = link(RDHmatrix(5:,:), 'mod');
R6 = link(RDHmatrix(6:,:), 'mod');
R7 = link(RDHmatrix(7:,:), 'mod');
R8 = link(RDHmatrix(8:,:), 'mod');
R9 = link(RDHmatrix(9:,:), 'mod');
R10 = link(RDHmatrix(10:,:), 'mod');
R11 = link(RDHmatrix(11:,:), 'mod');
R12 = link(RDHmatrix(12:,:), 'mod');
R13 = link(RDHmatrix(13:,:), 'mod');
R14 = link(RDHmatrix(14:,:), 'mod');
```

Appendix B (Continued)

```
L1 = link(LDHmatrix(1,:), 'mod');
L2 = link(LDHmatrix(2,:), 'mod');
L3 = link(LDHmatrix(3,:), 'mod');
L4 = link(LDHmatrix(4,:), 'mod');
L5 = link(LDHmatrix(5,:), 'mod');
L6 = link(LDHmatrix(6,:), 'mod');
L7 = link(LDHmatrix(7,:), 'mod');
L8 = link(LDHmatrix(8,:), 'mod');
L9 = link(LDHmatrix(9,:), 'mod');
L10 = link(LDHmatrix(10,:), 'mod');
L11 = link(LDHmatrix(11,:), 'mod');
L12 = link(LDHmatrix(12,:), 'mod');
L13 = link(LDHmatrix(13,:), 'mod');
L14 = link(LDHmatrix(13,:), 'mod');
```

```
% RTorso = robot({R1, R2, R3});
% RShoulder = robot({R1, R2, R3, R4, R5});
% RUpperArm = robot({R1, R2, R3, R4, R5, R6, R7, R8});
% RForearm = robot({R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11});
```

```
RUpperBody = robot({R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14});
RUpperBody.name = 'Right';
```

```
% LTorso = robot({L1, L2, L3});
% LShoulder = robot({L1, L2, L3, L4, L5});
% LUpperArm = robot({L1, L2, L3, L4, L5, L6, L7, L8});
% LForearm = robot({L1, L2, L3, L4, L5, L6, L7, L8, L9, L10, L11});
```

```
LUpperBody = robot({L1, L2, L3, L4, L5, L6, L7, L8, L9, L10, L11, L12, L13, L14});
LUpperBody.name = 'Left';
```

```
%plot(UpperBody,DHmatrix(:,3));
end
```

B.14 SubFunctions\ROMtest.m

```
% Calculate clinical joint angles and generate ROM plots
% RHBM 2/7/2011
function [ROM, ADL] = ROMtest(ROM, ADL, subject)
close all
figure('name','recitified')
ThetaCompiled = [];
for i=1:ROM.(subject).nfiles
    trial = removewhite(ROM.(subject).filenames(i,:));
    %trial = 'ElbFlex1';
    Theta = real(ROM.(subject).(trial).Theta);
```

Appendix B (Continued)

```
for j=1:size(Theta,1)
    Theta(j,7:9) = ClinicalSho('yxz', Theta(j,7:9));
    Theta(j,19:21) = ClinicalSho('yxz', Theta(j,19:21));
end

Theta(:,8) = pi-abs(Theta(:,8));
Theta(:,20) = pi-abs(Theta(:,20));

Theta(:,16) = -Theta(:,16)+pi;
Theta(:,18) = -Theta(:,18);
Theta(:,19) = -Theta(:,19);
Theta(:,21) = -Theta(:,21);
Theta(:,22) = -Theta(:,22);
Theta(:,24) = -Theta(:,24);
Theta(:,25) = -Theta(:,25);
Theta(:,27) = -Theta(:,27);

subplot(4,2,1)
hold on
plot(Theta(:,16:18))
title('L Shoulder')
legend('16','17','18')

subplot(4,2,2)
hold on
plot(Theta(:,4:6))
title('R Shoulder')
legend('4','5','6')

subplot(4,2,3)
hold on
plot(Theta(:,19:21))
title('L Upper Arm')
legend('19','20','21')

subplot(4,2,4)
hold on
plot(Theta(:,7:9))
title('R Upper Arm')
legend('7','8','9')

subplot(4,2,5)
hold on
plot(Theta(:,22:24))
```

Appendix B (Continued)

```
title('L Forearm')
legend('22','23','24')

subplot(4,2,6)
hold on
plot(Theta(:,10:12))
title('R Forearm')
legend('10','11','12')

subplot(4,2,7)
hold on
plot(Theta(:,25:27))
title('L Hand')
legend('25','26','27')

subplot(4,2,8)
hold on
plot(Theta(:,13:15))
title('R Hand')
legend('13','14','15')

ROM.(subject).(trial).ThetaClin = Theta;
saveas(gcf,['C:\Documents and
Settings\dlura\Desktop\RHBM\Figures\','ROM',subject,'.fig']);
ThetaCompiled = [ThetaCompiled; Theta];

end
ROM.(subject).RoM = [min(ThetaCompiled)', max(ThetaCompiled)',
max(ThetaCompiled)'-min(ThetaCompiled)'];

for i=1:ADL.(subject).nfiles
    trial = removewhite(ADL.(subject).filenames(i,:));
    Theta = real(ADL.(subject).(trial).Theta);

    for j=1:size(Theta,1)
        Theta(j,7:9) = ClinicalSho('yxz', Theta(j,7:9));
        Theta(j,19:21) = ClinicalSho('yxz', Theta(j,19:21));
    end

    Theta(:,8) = pi-abs(Theta(:,8));
    Theta(:,20) = pi-abs(Theta(:,20));

    Theta(:,16) = -Theta(:,16)+pi;
    Theta(:,18) = -Theta(:,18);
```


Appendix B (Continued)

```
Theta(:,19) = -Theta(:,19);  
Theta(:,21) = -Theta(:,21);  
Theta(:,22) = -Theta(:,22);  
Theta(:,24) = -Theta(:,24);  
Theta(:,25) = -Theta(:,25);  
Theta(:,27) = -Theta(:,27);
```

```
ADL.(subject).(trial).ThetaClin = Theta;  
ADL.(subject).(trial).RoM = [min(Theta)', max(Theta)', max(Theta)'-min(Theta)'];
```

end

%% Torso ROM Plot

```
close all  
figure('name', [subject, ' Torso Flexion (Extension +)'])  
hold on  
plot(ROM.(subject).TorFlex1.Theta(:,1:3))  
plot(ROM.(subject).TorFlex2.Theta(:,1:3))  
plot(ROM.(subject).TorFlex3.Theta(:,1:3))  
figure('name', [subject, ' Torso Lateral Flexion (Right +)'])  
hold on  
plot(ROM.(subject).TorLatF1.Theta(:,1:3))  
plot(ROM.(subject).TorLatF2.Theta(:,1:3))  
plot(ROM.(subject).TorLatF3.Theta(:,1:3))  
figure('name', [subject, ' Torso Rotation (Left +)'])  
hold on  
plot(ROM.(subject).TorRota1.Theta(:,1:3))  
plot(ROM.(subject).TorRota2.Theta(:,1:3))  
plot(ROM.(subject).TorRota3.Theta(:,1:3))
```

%% Shoulder ROM Plot

```
close all  
figure('name', [subject, ' Shoulder Abduction'])  
hold on  
plot(ROM.(subject).ShoAdbu1.Theta(:,4:6))  
plot(ROM.(subject).ShoAdbu2.Theta(:,4:6))  
plot(ROM.(subject).ShoAdbu3.Theta(:,4:6))  
plot(ROM.(subject).ShoAdbu1.Theta(:,16:18),':')  
plot(ROM.(subject).ShoAdbu2.Theta(:,16:18),':')  
plot(ROM.(subject).ShoAdbu3.Theta(:,16:18),':')  
figure('name', [subject, ' Shoulder Flexion'])  
hold on  
plot(ROM.(subject).ShoFlex1.Theta(:,4:6))  
plot(ROM.(subject).ShoFlex2.Theta(:,4:6))  
plot(ROM.(subject).ShoFlex3.Theta(:,4:6))
```

Appendix B (Continued)

```
plot(ROM.(subject).ShoFlex1.Theta(:,16:18),':')
plot(ROM.(subject).ShoFlex2.Theta(:,16:18),':')
plot(ROM.(subject).ShoFlex3.Theta(:,16:18),':')
figure('name', [subject, ' Shoulder Axial Rotation'])
hold on
plot(ROM.(subject).ShoRota1.Theta(:,4:6))
plot(ROM.(subject).ShoRota2.Theta(:,4:6))
plot(ROM.(subject).ShoRota3.Theta(:,4:6))
plot(ROM.(subject).ShoRota1.Theta(:,16:18),':')
plot(ROM.(subject).ShoRota2.Theta(:,16:18),':')
plot(ROM.(subject).ShoRota3.Theta(:,16:18),':')
```

%% Upper Arm ROM Plot

```
close all
subject = 'C04';
figure('name', [subject, ' Shoulder Abduction'])
hold on
plot(ROM.(subject).ShoAdbu1.Theta(:,7:9))
plot(ROM.(subject).ShoAdbu2.Theta(:,7:9))
plot(ROM.(subject).ShoAdbu3.Theta(:,7:9))
plot(ROM.(subject).ShoAdbu1.Theta(:,19:21),':')
plot(ROM.(subject).ShoAdbu2.Theta(:,19:21),':')
plot(ROM.(subject).ShoAdbu3.Theta(:,19:21),':')
figure('name', [subject, ' Shoulder Flexion'])
hold on
plot(ROM.(subject).ShoFlex1.Theta(:,7:9))
plot(ROM.(subject).ShoFlex2.Theta(:,7:9))
plot(ROM.(subject).ShoFlex3.Theta(:,7:9))
plot(ROM.(subject).ShoFlex1.Theta(:,19:21),':')
plot(ROM.(subject).ShoFlex2.Theta(:,19:21),':')
plot(ROM.(subject).ShoFlex3.Theta(:,19:21),':')
figure('name', [subject, ' Shoulder Axial Rotation'])
hold on
plot(ROM.(subject).ShoRota1.Theta(:,7:9))
plot(ROM.(subject).ShoRota2.Theta(:,7:9))
plot(ROM.(subject).ShoRota3.Theta(:,7:9))
plot(ROM.(subject).ShoRota1.Theta(:,19:21),':')
plot(ROM.(subject).ShoRota2.Theta(:,19:21),':')
plot(ROM.(subject).ShoRota3.Theta(:,19:21),':')
```

```
%%
% Subjects checking
close all
trial = 'Elbflex1';
```

Appendix B (Continued)

```
% figure
% plot(ROM.C05.(trial).Theta(:,1:3))
% title('Torso')
% legend('1','2','3')

figure('name','standard')
subplot(4,2,1)
plot(ROM.(subject).(trial).Theta(:,16:18))
title('L Shoulder')
legend('16','17','18')

subplot(4,2,2)
plot(ROM.(subject).(trial).Theta(:,4:6))
title('R Shoulder')
legend('4','5','6')

subplot(4,2,3)
plot(ROM.(subject).(trial).Theta(:,19:21))
title('L Upper Arm')
legend('19','20','21')

subplot(4,2,4)
plot(ROM.(subject).(trial).Theta(:,7:9))
title('R Upper Arm')
legend('7','8','9')

subplot(4,2,5)
plot(ROM.(subject).(trial).Theta(:,22:24))
title('L Forearm')
legend('22','23','24')

subplot(4,2,6)
plot(ROM.(subject).(trial).Theta(:,10:12))
title('R Forearm')
legend('10','11','12')

subplot(4,2,7)
plot(ROM.(subject).(trial).Theta(:,25:27))
title('L Hand')
legend('25','26','27')

subplot(4,2,8)
plot(ROM.(subject).(trial).Theta(:,13:15))
title('R Hand')
```

Appendix B (Continued)

```
% legend('13','14','15')
end
```

B.15 SubFunctions\CompileError.m

```
% Function for finding average global error, subject error, taskerror, and trial
% error
function Error = CompileError(Train, Signal)
```

```
Error.Global = [];
Error.Joint.Global = [];
```

```
Error.Brush = [];
Error.Drink = [];
Error.Eat = [];
Error.Lift = [];
Error.Open = [];
```

```
Error.Joint.Brush = [];
Error.Joint.Drink = [];
Error.Joint.Eat = [];
Error.Joint.Lift = [];
Error.Joint.Open = [];
```

```
Error.Trial = [];
Error.Joint.Trial = [];
```

```
for s=1:size(fieldnames(Train),1)
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));
```

```
Error.(subject) = [];
Error.Joint.(subject) = [];
```

```
for t=1:(size(names,1)-2)
    name = char(names(t,:));
    sections = fieldnames(Train.(subject).(name));
```

```
jointTaskError = [];
```

```
for i=3:size(sections,1)
    section = char(sections(i,:));
```

```
if isfield(Train.(subject).(name).(section),Signal)
```

Appendix B (Continued)

```
if ~any(any(isnan(Train.(subject).(name).(section).(Signal))))

    jointTaskError = [jointTaskError; Train.(subject).(name).(section).(Signal)];

end

end % is signal

end % Section

if size(jointTaskError,1)<1;
    continue
end

jointTaskError = mean(jointTaskError);
tskError = mean(jointTaskError);

Error.Global = [Error.Global; tskError];
Error.Joint.Global = [Error.Joint.Global; jointTaskError];

Error.(subject) = [Error.(subject); tskError];
Error.Joint.(subject) = [Error.Joint.(subject); jointTaskError];

if strcmpi(name(1),'B')
    Error.Brush = [Error.Brush ; tskError];
    Error.Joint.Brush = [Error.Joint.Brush ; jointTaskError];
elseif strcmpi(name(1),'D')&&(~strcmpi(name(1:2),'Do'))
    Error.Drink = [Error.Drink ; tskError];
    Error.Joint.Drink = [Error.Joint.Drink ; jointTaskError];
elseif strcmpi(name(1),'E')
    Error.Eat = [Error.Eat ; tskError];
    Error.Joint.Eat = [Error.Joint.Eat ; jointTaskError];
elseif strcmpi(name(1),'L')
    Error.Lift = [Error.Lift ; tskError];
    Error.Joint.Lift = [Error.Joint.Lift ; jointTaskError];
elseif strcmpi(name(1),'O')||strcmpi(name(1:2),'Do')
    Error.Open = [Error.Open ; tskError];
    Error.Joint.Open = [Error.Joint.Open ; jointTaskError];
end

Error.Trial = [Error.Trial, {[subject, name]; tskError.^0.5}];
Error.Joint.Trial = [Error.Joint.Trial, {[{subject, name}];
num2cell(jointTaskError'.^0.5)}];
```

Appendix B (Continued)

end % Trials

```
Error.(subject) = mean(Error.(subject)).^(0.5);  
Error.Joint.(subject) = mean(Error.Joint.(subject)).^(0.5);
```

end % Subjects

```
Error.Global = mean(Error.Global).^(0.5);  
Error.Joint.Global = mean(Error.Joint.Global).^(0.5);
```

```
Error.Brush = mean(Error.Brush).^(0.5);  
Error.Drink = mean(Error.Drink).^(0.5);  
Error.Eat = mean(Error.Eat).^(0.5);  
Error.Lift = mean(Error.Lift).^(0.5);  
Error.Open = mean(Error.Open).^(0.5);
```

```
Error.Joint.Brush = mean(Error.Joint.Brush).^(0.5);  
Error.Joint.Drink = mean(Error.Joint.Drink).^(0.5);  
Error.Joint.Eat = mean(Error.Joint.Eat).^(0.5);  
Error.Joint.Lift = mean(Error.Joint.Lift).^(0.5);  
Error.Joint.Open = mean(Error.Joint.Open).^(0.5);
```

end % function

B.16 TrainBi.m

```
% Create and training for inverse kinematics of the  
% "Robotics-based Human Upper Body Model" (RHBM).  
% Derek J. Lura 2/6/2011
```

```
% Add the SubFunctions folder to the path  
path([cd, '\SubFunctions'], path)
```

```
% Clear variables from the current workspace  
clear all
```

```
% Close all open figure windows (plots)  
close all
```

```
% Determine if plot functions should be run  
plots = 0;
```

```
% Initialize position vectors  
RP = [];  
LP = [];  
P = [];
```

Appendix B (Continued)

```
Bad = [];  
  
% Initialize joint angle vectors  
RT = [];  
LT = [];  
T = [];  
  
BName = ['B1'; 'B2'; 'B3'; 'B4'; 'B5'; 'B6'; 'B7'; 'B8'; 'B9'];  
  
% Set the to directory to /Subjects  
cd 'Subjects\  
  
% Load data for specified subjects  
for s=[1:20]  
  
    % Determine if plot functions should be run  
    plots = 0;  
  
    % Set the Subject listing.  
    subjects = ['C01'; 'C02'; 'C03'; 'C04'; 'C05'; 'C06'; 'C07'; 'C08'; 'C09'; 'C10'; ...  
               'B01'; 'B02'; 'B03'; 'B04'; 'B05'; 'B06'; 'B07'; 'B08'; 'B09'; 'B10'; ...  
               'R01'; 'R02'; 'R03'; 'R04'; 'R05'; 'R06'; 'R07'; 'R08'; 'R09'; 'R10'; ...  
               'T01'; 'T02'; 'T03'; 'T04'; 'T05'; 'T06'; 'T07'; 'T08'; 'T09'; 'T10'];  
  
    subject = char(subjects(s,:));  
    Temp = load([subject, 'UpperBodyModel']);  
  
    Train.(subject) = Temp.(subject);  
  
    RUpperBody = Train.(subject).RUpperBody;  
    LUpperBody = Train.(subject).LUpperBody;  
  
    names = fieldnames(Train.(subject));  
  
    for t=1:(size(names,1))  
        name = char(names(t,:));  
  
        if strcmpi(name, 'RUpperBody')||strcmpi(name, 'LUpperBody')  
            continue  
        end  
  
        % Get the Theta matrices  
        RTheta = Train.(subject).(name).RTheta;  
        LTheta = Train.(subject).(name).LTheta;
```

Appendix B (Continued)

```
Theta = [RTheta, LTheta(:,4:14)];
dT3 = LTheta(:,3) - RTheta(:,3);
dT3(isnan(dT3)) = [];
dT3 = mean(dT3);

% Check for unusual data
if any( (max(Theta) - min(Theta))>4 )
    disp([subject, name, ' is bad data'])
    Bad = [Bad, {subject;name}];
end

% Fill Gaps in Joint angle Data
while any(isnan(Theta(1,:)))
    Theta(1,:) = [];
end
while any(isnan(Theta(end,:)))
    Theta(size(Theta,1),:) = [];
end

% Fill Gaps in Joint angle Data
Theta = FilGap(Theta);

% Decrease frequency
Theta = condense(Theta, 6);

j = 1;
databreak = 0;
breakPoint = 1;
while j<=size(Theta,1)
    if any(isnan(Theta(j,:)))
        Theta(j,:) = [];
        if (~databreak)&&(j~=1)
            breakPoint = [breakPoint, j];
        end
        databreak = 1;
    else
        databreak = 0;
        j=j+1;
    end
end

if max(breakPoint)~=j
    breakPoint = [breakPoint, j];
end
```


Appendix B (Continued)

```
if (j<=20)||((size(breakPoint,2)>=10)
    disp([name, 'small break']);
    continue
end

for i=1:size(breakPoint,2)-1
    section = char(BName(i,:));
    sSize = (breakPoint(i+1)-breakPoint(i));
    if (sSize<=15)
        [name, ' Bilateral ', section, 'is to short'];
        continue
    end

    Train.(subject).(name).(section).Theta =
    WMAfilter(11,Theta(breakPoint(i):(breakPoint(i+1)-1),:));
    Thetai = Train.(subject).(name).(section).Theta;

    RThetai = Thetai(:,1:14);
    LThetai = Thetai(:,[1:3,15:25]);
    LThetai(:,3) = LThetai(:,3) + dT3;
    Train.(subject).(name).(section).dT3 = dT3;

    % Calculate End Effector Position
    RfPos = fkine(RUpperBody, RThetai);
    LfPos = fkine(LUpperBody, LThetai);

    Train.(subject).(name).(section).T = Thetai(:,:);

    % Reset end effector position for loop calculaions
    Rposition = [];
    Rposition(:, :) = RfPos(1:3,4,:);
    Rrotation = tr2rpy(RfPos(:, :, :));
    Train.(subject).(name).(section).RP = [Rposition; sin(Rrotation); cos(Rrotation)];

    % Reset end effector position for loop calculaions
    Lposition = [];
    Lposition(:, :) = LfPos(1:3,4,:);
    Lrotation = tr2rpy(LfPos(:, :, :));
    Train.(subject).(name).(section).LP = [Lposition; sin(Lrotation); cos(Lrotation)];

    % Compile right arm end effector position/orientation data.
    % This section can be used to include / exclude tasks from
    % training.
    if strcmpi(name(1), 'B')
```

Appendix B (Continued)

```
        P = [P, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP] ];
        T = [T, Train.(subject).(name).(section).T];
        elseif strcmpi(name(1),'D')&&(~strcmpi(name(1:2),'Do'))
        P = [P, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP] ];
        T = [T, Train.(subject).(name).(section).T];
        elseif strcmpi(name(1),'E')
        P = [P, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP] ];
        T = [T, Train.(subject).(name).(section).T];
        elseif strcmpi(name(1),'L')
        P = [P, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP] ];
        T = [T, Train.(subject).(name).(section).T];
        elseif strcmpi(name(1),'O')||strcmpi(name(1:2),'Do')
        P = [P, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP] ];
        T = [T, Train.(subject).(name).(section).T];
    end
```

end

end % Trials

end % Subjects

disp('so far so good')

B.17 SubFunctions\FilGap.m

```
function [MarkerFilled] = FilGap(Marker)
```

```
% Fills gaps in data Marker
```

```
Temp = [Marker(:,:(1:size(Marker,1)))];
```

```
Temp(any(isnan(Temp),2),:) = [];
```

```
if (size(Temp,1)+120 <= size(Marker,1))
```

```
    disp('To Many Gaps to Fill')
```

```
    MarkerFilled = Marker;
```

```
    return
```

```
end
```

```
for i = 1:size(Marker,1)
```

```
    for j = 1:size(Marker,2)
```

```
        if isnan(Marker(i,j))
```

Appendix B (Continued)

```
        Marker(i,j) = spline(Temp(:,size(Marker,2)+1), Temp(:,j), i);
    end
end
end
```

```
MarkerFilled = Marker;
```

B.18 SubFunctions\condense.m

```
% Remove Data From a DataSet
function xc = condense(x,n)
% Input data set, x
% Condensation factor, n (must be a whole number)
% NewSize =< OldSize/n
dsize = size(x,2);

for i=1:floor(size(x,1)/n)
    tempx = zeros(1,dsize);
    for j=1:n
        tempx = tempx + x((j+n*(i-1)),:);
    end % for j
    xc(i,:) = x(n*i,:); % tempx/n;
end % for i

end % function
```

B.19 SubFunctions\TestBiLN.m

```
%% Bilateral Least Norm testing script
for s=1:size(fieldnames(Train),1)
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    for t=1:(size(names,1)-2)
        name = char(names(t,:));
        sections = fieldnames(Train.(subject).(name));

        for i=3:size(sections,1)
            section = char(sections(i,:));

            if isfield(Train.(subject).(name).(section), 'Theta')

                Thetai = Train.(subject).(name).(section).Theta;
```

Appendix B (Continued)

```
sSize = size(Thetai,1);
if (sSize<=15)
    [name, section, 'is to short'];
    continue
end

RThetai = Thetai(:,1:14);
LThetai = Thetai(:,[1:3,15:25]);
LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;

RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

qR = RThetai(1,:);
qL = LThetai(1,:);
q = Thetai(1,:);

LNTheta = Thetai(1,:);
for j=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

    dq = J*(J'*J)^-1 * [eR; eL];
    q = q + dq;
    qR = qR + dq(1:14);
    qL = qL + dq([1:3,15:25]);
    LNTheta = [LNTheta; q'];
end

Train.(subject).(name).(section).BiLNTheta = LNTheta;
Train.(subject).(name).(section).ErrorLN = (LNTheta - Thetai).^2;
end

end % Section

end % Trials

end % Subjects
```

Appendix B (Continued)

B.20 SubFunctions\TestBiWLN_Dyn.m

% Bilateral Dynamic Weighted Least Norm testing script

% RHBM 2/6/2011 Derek J. Lura

function Train = TestBiWLN_Dyn(Train)

% Weights Optimization

function weights = optimBiWeights(Jaco, dtheta, dx, GWeight)

if nargin <= 3

 GWeight = ones(25,1)*.5;

end

% Unconstrained Weight Approximation

error = @(x) sum((dtheta - (diag(x)*Jaco*(Jaco*diag(x)*Jaco)^-1)*dx).^2);

% Constrained Weight Approximation

% Penalty for distance from initial guess

% A = 0.01

% Penalty for rate of change of the weights

% B = 0.02

% error = @(x) sum((dtheta - (diag(x)*Jaco*(Jaco*diag(x)*Jaco)^-1)*dx).^2 +
A*(ones(25,1)*.5-x).^6 + B*(GWeight-x).^2);

lb = ones(25,1)*0.001;

ub = ones(25,1);

options=optimset('Algorithm','active-set','Display','off');

[weights] = fmincon(error, GWeight,[],[],[],[],lb,ub,[],options);

end

for s=1:size(fieldnames(Train),1)

 subjects = fieldnames(Train);

 subject = char(subjects(s,:));

 names = fieldnames(Train.(subject));

 RUpperBody = Train.(subject).RUpperBody;

 LUpperBody = Train.(subject).LUpperBody;

 for t=1:(size(names,1)-2)

 name = char(names(t,:));

 sections = fieldnames(Train.(subject).(name));

Appendix B (Continued)

```
for i=3:size(sections,1)
    section = char(sections(i,:));

    if isfield(Train.(subject).(name).(section), 'Theta')

        Thetai = Train.(subject).(name).(section).Theta;
        sSize = size(Thetai,1);
        if (sSize<=15)
            disp([name, section, 'is to short']);
            continue
        end

        RThetai = Thetai(:,1:14);
        LThetai = Thetai(:,[1:3,15:25]);
        LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;

        RfPos = fkine(RUpperBody, RThetai);
        LfPos = fkine(LUpperBody, LThetai);

        qR = RThetai(1,:);
        qL = LThetai(1,:);
        q = Thetai(1,:);

        WLNTheta = Thetai(1,:);

        Warray = ones(25,1)*.5;
        for j=1:(size(RThetai,1)-1)
            JR = jacob0(RUpperBody, qR);
            JL = jacob0(LUpperBody, qL);
            J = JR;
            J(7:12,1:3) = JL(:,1:3);
            J(7:12,15:25) = JL(:,4:14);

            eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
            eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

            W = optimBiWeights(J, Thetai(j+1,:)-q, [eR; eL], Warray(:,j));
            Warray(:,j+1) = W;
            diagx = diag(W);
            dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

            q = q + dqw;
            qR = qR + dqw(1:14);
            qL = qL + dqw([1:3,15:25]);
```

Appendix B (Continued)

```
        WLNTheta = [WLNTheta; q'];

    end

    Train.(subject).(name).(section).Warray = Warray;
    Train.(subject).(name).(section).BiWLNTheta = WLNTheta;
    Train.(subject).(name).(section).ErrorWLN = (WLNTheta - Theta).^2;

    end

end % Section

end % Trials

end % Subjects

end
```

B.21 SubFunctions\TestBiWLN_Sta.m

% Static Bilateral Weighted Least Norm testing function

% RHBM 2/6/2011 Derek J. Lura

function [Train] = TestBiWLN_Sta(Train)

% Start with the default options

options = optimset;

% Modify options setting

options = optimset(options, 'Display', 'off');

options = optimset(options, 'MaxIter', 5);

options = optimset(options, 'LargeScale', 'off');

options = optimset(options, 'Algorithm', 'active-set');

options = optimset(options, 'PlotFcns', { @optimplotfval });

% Specify Bounds

lb = ones(25,1)*0.001;

ub = ones(25,1);

% Limit to Joints 1-3, R4-6, L4-6 (15-17)

function cost = errorfun(W)

% W = [Wlim(1:6);0.5*ones(8,1);Wlim(7:9);0.5*ones(8,1)];

diagx = diag(W);

cost = 0;

for i=3:size(sections,1)

section = char(sections(i,:));

if isfield(Train.(subject).(name).(section), 'Theta')

Appendix B (Continued)

```
Thetai = Train.(subject).(name).(section).Theta;
sSize = size(Thetai,1);
if (sSize<=15)
    [name, section, 'is to short'];
    continue
end

RThetai = Thetai(:,1:14);
LThetai = Thetai(:,[1:3,15:25]);
LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;

RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

qR = RThetai(1,:);
qL = LThetai(1,:);
q = Thetai(1,:);

WLNTheta = Thetai(1,:);

for j=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

    dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

    q = q + dqw;
    qR = qR + dqw(1:14);
    qL = qL + dqw([1:3,15:25]);
    WLNTheta = [WLNTheta; q];
end

cost = cost+sum(sum((WLNTheta - Thetai).^2));

end
end
end
```


Appendix B (Continued)

```
for s=1:size(fieldnames(Train),1)
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    for t=1:(size(names,1)-2)
        name = char(names(t,:));
        sections = fieldnames(Train.(subject).(name));
        disp([subject, ', ', name]);
        tic
        W = fmincon(@(x) errorfun(x), ones(25,1)*0.5,[],[],[],[],lb,ub,[],options)
        toc
        Train.(subject).(name).StatW = W;
        diagx = diag(W);

        for i=3:size(sections,1)
            section = char(sections(i,:));

            if isfield(Train.(subject).(name).(section), 'Theta')

                Thetai = Train.(subject).(name).(section).Theta;
                sSize = size(Thetai,1);
                if (sSize<=15)
                    [name, section, 'is to short'];
                    continue
                end

                RThetai = Thetai(:,1:14);
                LThetai = Thetai(:, [1:3,15:25]);
                LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;

                RfPos = fkine(RUpperBody, RThetai);
                LfPos = fkine(LUpperBody, LThetai);

                qR = RThetai(1,:);
                qL = LThetai(1,:);
                q = Thetai(1,:);

                StatWLNTheta = Thetai(1,:);

            for j=1:(size(RThetai,1)-1)
```

Appendix B (Continued)

```
JR = jacob0(RUpperBody, qR);
JL = jacob0(LUpperBody, qL);
J = JR;
J(7:12,1:3) = JL(:,1:3);
J(7:12,15:25) = JL(:,4:14);
```

```
eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));
```

```
dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];
```

```
q = q + dqw;
qR = qR + dqw(1:14);
qL = qL + dqw([1:3,15:25]);
StatWLNTheta = [StatWLNTheta; q];
```

end

```
Train.(subject).(name).(section).BiStatWLNTheta = StatWLNTheta;
Train.(subject).(name).(section).ErrorStatWLN = (StatWLNTheta - Thetai).^2;
```

end

end % Section

end % Trials

end % Subjects

end % Test Fucntion

B.22 SubFunctions\TestBiWLN_Sub.m

% Subject Weighted Least Norm testing script

% RHBM 2/6/2011 Derek J. Lura

```
function [Train] = TestBiWLN_Sub(Train)
```

% Start with the default options

```
options = optimset;
```

% Modify options setting

```
options = optimset(options, 'Display', 'off');
```

```
options = optimset(options, 'MaxIter', 5);
```

```
options = optimset(options, 'LargeScale', 'off');
```

```
options = optimset(options, 'Algorithm', 'active-set');
```

```
options = optimset(options, 'PlotFcns', { @optimplotfval });
```

Appendix B (Continued)

% Specify Bounds

```
lb = ones(25,1)*0.001;
```

```
ub = ones(25,1);
```

% Limit to Joints 1-3, R4-6, L4-6 (15-17)

```
function cost = errorfun(W)
```

```
    diagx = diag(W);
```

```
    cost = 0;
```

```
    for ts=1:(size(names,1)-2)
```

```
        name = char(names(ts,:));
```

```
        sections = fieldnames(Train.(subject).(name));
```

```
    for is=3:size(sections,1)
```

```
        section = char(sections(is,:));
```

```
        if isfield(Train.(subject).(name).(section), 'Theta')
```

```
            Thetai = Train.(subject).(name).(section).Theta;
```

```
            sSize = size(Thetai,1);
```

```
            if (sSize<=15)
```

```
                disp([name, section, 'is to short']);
```

```
                continue
```

```
            end
```

```
            RThetai = Thetai(:,1:14);
```

```
            LThetai = Thetai(:,[1:3,15:25]);
```

```
            LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;
```

```
            RfPos = fkine(RUpperBody, RThetai);
```

```
            LfPos = fkine(LUpperBody, LThetai);
```

```
            qR = RThetai(1,:);
```

```
            qL = LThetai(1,:);
```

```
            q = Thetai(1,:);
```

```
            WLNTheta = Thetai(1,:);
```

```
            for js=1:(size(RThetai,1)-1)
```

```
                JR = jacob0(RUpperBody, qR);
```

```
                JL = jacob0(LUpperBody, qL);
```

```
                J = JR;
```

```
                J(7:12,1:3) = JL(:,1:3);
```

```
                J(7:12,15:25) = JL(:,4:14);
```

```
            eR = tr2diff(fkine(RUpperBody, qR), RfPos(:, :, js+1));
```

Appendix B (Continued)

```
eL = tr2diff(fkine(LUpperBody, qL), LfPos(:, :, js+1));

dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

q = q + dqw;
qR = qR + dqw(1:14);
qL = qL + dqw([1:3, 15:25]);
WLNTheta = [WLNTheta; q'];
end

cost = cost+sum(sum((WLNTheta - Thetai).^2));

end
end
end
end

for s=1:size(fieldnames(Train),1)
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    disp(subject);
    tic
    W = fmincon(@(x) errorfun(x), ones(25,1)*0.5,[],[],[],[],lb,ub,[],options)
    toc
    Train.(subject).(name).SubjW = W;
    diagx = diag(W);

    for t=1:(size(names,1)-2)
        name = char(names(t,:));
        sections = fieldnames(Train.(subject).(name));

        for i=3:size(sections,1)
            section = char(sections(i,:));

            if isfield(Train.(subject).(name).(section), 'Theta')

                Thetai = Train.(subject).(name).(section).Theta;
                sSize = size(Thetai,1);
                if (sSize<=15)
```

Appendix B (Continued)

```
    %[name, section, 'is to short'];
    continue
end

RThetai = Thetai(:,1:14);
LThetai = Thetai(:,[1:3,15:25]);
LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;

RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

qR = RThetai(1,:);
qL = LThetai(1,:);
q = Thetai(1,:);

SubjWLNTheta = Thetai(1,:);
BiRSubjWLNTheta = RThetai(1,:);
BiLSubjWLNTheta = LThetai(1,:);

for j=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

    dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

    q = q + dqw;
    qR = qR + dqw(1:14);
    qL = qL + dqw([1:3,15:25]);
    SubjWLNTheta = [SubjWLNTheta; q];
end

Train.(subject).(name).(section).BiSubjWLNTheta = SubjWLNTheta;
Train.(subject).(name).(section).ErrorSubjWLN = (SubjWLNTheta -
Thetai).^2;

end

end % Section
```

Appendix B (Continued)

```
end % Trials
```

```
end % Subjects
```

```
end % Test Fucntion
```

B.23 SubFunctions\TestBiWLN_Tas.m

```
% Task Weighted Least Norm testing script
```

```
% RHBM 2/6/2011 Derek J. Lura
```

```
function [Train, Weights] = TestBiWLN_Tas(Train)
```

```
Dominant = ['R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...  
            'R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...  
            'L'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; ...  
            'R'; 'L'; 'R'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'];
```

```
% Start with the default options
```

```
options = optimset;
```

```
% Modify options setting
```

```
options = optimset(options, 'Display', 'off');
```

```
options = optimset(options, 'MaxIter', 5);
```

```
options = optimset(options, 'LargeScale', 'off');
```

```
options = optimset(options, 'Algorithm', 'active-set');
```

```
options = optimset(options, 'PlotFcns', { @optimplotfval });
```

```
% Specifiy Bounds
```

```
lb = ones(25,1)*0.001;
```

```
ub = ones(25,1);
```

```
% Limit to Joints 1-3, R4-6, L4-6 (15-17)
```

```
function cost = errorfun(W, task)
```

```
cost = 0;
```

```
for ss=1:size(fieldnames(Train),1)
```

```
    subjects = fieldnames(Train);
```

```
    subject = char(subjects(ss,:));
```

```
    names = fieldnames(Train.(subject));
```

```
    if strcmpi(Dominant(10+ss), 'R')
```

```
        Wfixed = W(1:25);
```

```
    elseif strcmpi(Dominant(10+ss), 'L')
```

```
        Wfixed = [W(1:3);
```

```
                  W(15:25);
```

```
                  W(4:14)];
```

```
    end
```

Appendix B (Continued)

```
diagx = diag(W);

RUpperBody = Train.(subject).RUpperBody;
LUpperBody = Train.(subject).LUpperBody;

for ts=1:(size(names,1)-2)
    name = char(names(ts,:));
    sections = fieldnames(Train.(subject).(name));

    if strcmpi(task, 'Brush')&&(~strcmpi(name(1), 'B'))
        continue
    elseif strcmpi(task,
'Drink')&&((~strcmpi(name(1), 'D'))&&(~strcmpi(name(1:2), 'Do'))))
        continue
    elseif strcmpi(task, 'Eat')&&(~strcmpi(name(1), 'E'))
        continue
    elseif strcmpi(task, 'Lift')&&(~strcmpi(name(1), 'L'))
        continue
    elseif strcmpi(task,
'Open')&&( ~(strcmpi(name(1), 'O') || strcmpi(name(1:2), 'Do')) )
        continue
    end

    for is=3:size(sections,1)
        section = char(sections(is,:));
        if isfield(Train.(subject).(name).(section), 'Theta')

            Thetai = Train.(subject).(name).(section).Theta;
            sSize = size(Thetai,1);
            if (sSize<=15)
                %disp([name, section, 'is to short']);
                continue
            end

            RThetai = Thetai(:,1:14);
            LThetai = Thetai(:, [1:3, 15:25]);
            LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;

            RfPos = fkine(RUpperBody, RThetai);
            LfPos = fkine(LUpperBody, LThetai);

            qR = RThetai(1,:);
            qL = LThetai(1,:);
            q = Thetai(1,:);
```

Appendix B (Continued)

```
WLNTheta = Thetai(1,:);

for js=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,js+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,js+1));

    dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

    q = q + dqw;
    qR = qR + dqw(1:14);
    qL = qL + dqw([1:3,15:25]);
    WLNTheta = [WLNTheta; q'];
end

cost = cost+sum(sum((WLNTheta - Thetai).^2));

end
end
end
end
end

disp('Brush')
tic
Weights.Brush.W = fmincon(@(x) errorfun(x, 'Brush'),
ones(25,1)*0.5,[],[],[],[],lb,ub,[],options);
toc
disp('Drink')
Weights.Drink.W = fmincon(@(x) errorfun(x, 'Drink'),
ones(25,1)*0.5,[],[],[],[],lb,ub,[],options);
disp('Eat')
Weights.Eat.W = fmincon(@(x) errorfun(x, 'Eat'),
ones(25,1)*0.5,[],[],[],[],lb,ub,[],options);
disp('Lift')
Weights.Lift.W = fmincon(@(x) errorfun(x, 'Lift'),
ones(25,1)*0.5,[],[],[],[],lb,ub,[],options);
disp('Open')
```


Appendix B (Continued)

```
Weights.Open.W = fmincon(@(x) errorfun(x, 'Open'),  
ones(25,1)*0.5,[],[],[],[],lb,ub,[],options);  
disp('Weights Done')
```

```
for s=1:size(fieldnames(Train),1)  
    subjects = fieldnames(Train);  
    subject = char(subjects(s,:));  
    names = fieldnames(Train.(subject));
```

```
RUpperBody = Train.(subject).RUpperBody;  
LUpperBody = Train.(subject).LUpperBody;
```

```
for t=1:(size(names,1)-2)  
    name = char(names(t,:));  
    sections = fieldnames(Train.(subject).(name));
```

```
if strcmpi(name(1),'B')  
    diagx = diag(Weights.Brush.W);  
elseif strcmpi(name(1),'D')&&strcmpi(name(1:2),'Do')  
    diagx = diag(Weights.Drink.W);  
elseif strcmpi(name(1),'E')  
    diagx = diag(Weights.Eat.W);  
elseif strcmpi(name(1),'L')  
    diagx = diag(Weights.Lift.W);  
elseif (strcmpi(name(1),'O')||strcmpi(name(1:2),'Do'))  
    diagx = diag(Weights.Open.W);  
end
```

```
for i=3:size(sections,1)  
    section = char(sections(i,:));
```

```
if isfield(Train.(subject).(name).(section),'Theta')
```

```
    Thetai = Train.(subject).(name).(section).Theta;  
    sSize = size(Thetai,1);  
    if (sSize<=15)  
        %[name, section, 'is to short'];  
        continue  
    end
```

```
    RThetai = Thetai(:,1:14);  
    LThetai = Thetai(:,[1:3,15:25]);  
    LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;
```

Appendix B (Continued)

```
RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

qR = RThetai(1,:);
qL = LThetai(1,:);
q = Thetai(1,:);

TaskWLNTheta = Thetai(1,:);

for j=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

    dqw = (diagx*J*(J*diagx*J)^-1) * [eR; eL];

    q = q + dqw;
    qR = qR + dqw(1:14);
    qL = qL + dqw([1:3,15:25]);
    TaskWLNTheta = [TaskWLNTheta; q];
end

Train.(subject).(name).(section).BiTaskWLNTheta = TaskWLNTheta;
Train.(subject).(name).(section).ErrorTaskWLN = (TaskWLNTheta -
Thetai).^2;

end

end % Section

end % Trials

end % Subjects

end % Test Fuction

B.24 SubFunctions\TestBiWLN_Glo.m
% Global Weighted Least Norm testing script
% RHBM Derek J. Lura 2/9/2011
% WARNING this function can take several hours to run
```

Appendix B (Continued)

```
% if the number of included subjects is large.
function [Train, W] = TestBiWLN_Glo(Train, W, trainSubjects)

Dominant = ['R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'L'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; ...
            'R'; 'L'; 'R'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'];

% Start with the default options
options = optimset;

% Modify options setting
options = optimset(options, 'Display', 'off');
options = optimset(options, 'MaxIter', 5);
options = optimset(options, 'LargeScale', 'off');
options = optimset(options, 'Algorithm', 'active-set');
options = optimset(options, 'PlotFcns', { @optimplotfval });

% Specify Bounds
lb = ones(25,1)*0.001;
ub = ones(25,1);
count = 0;
tic
% Limit to Joints 1-3, R4-6, L4-6 (15-17)
function cost = errorfun(W)
    cost = 0;
    for ss = trainSubjects

        if strcmpi(Dominant(10+ss), 'R')
            Wfixed = W(1:25);
        elseif strcmpi(Dominant(10+ss), 'L')
            Wfixed = [W(1:3);
                    W(15:25);
                    W(4:14)];
        end

        diagW = diag(Wfixed);

        subjects = fieldnames(Train);
        subject = char(subjects(ss,:));
        names = fieldnames(Train.(subject));

        RUpperBody = Train.(subject).RUpperBody;
        LUpperBody = Train.(subject).LUpperBody;
```

Appendix B (Continued)

```
for ts=1:(size(names,1)-2)
    name = char(names(ts,:));
    sections = fieldnames(Train.(subject).(name));

    for is=3:size(sections,1)
        section = char(sections(is,:));
        if isfield(Train.(subject).(name).(section), 'Theta')

            Thetai = Train.(subject).(name).(section).Theta;
            sSize = size(Thetai,1);
            if (sSize<=15)
                %disp([name, section, 'is to short']);
                continue
            end

            RThetai = Thetai(:,1:14);
            LThetai = Thetai(:,[1:3,15:25]);
            LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;

            RfPos = fkine(RUpperBody, RThetai);
            LfPos = fkine(LUpperBody, LThetai);

            qR = RThetai(1,:);
            qL = LThetai(1,:);
            q = Thetai(1,:);

            WLNTheta = Thetai(1,:);

            for js=1:(size(RThetai,1)-1)
                JR = jacob0(RUpperBody, qR);
                JL = jacob0(LUpperBody, qL);
                J = JR;
                J(7:12,1:3) = JL(:,1:3);
                J(7:12,15:25) = JL(:,4:14);

                eR = tr2diff(fkine(RUpperBody, qR), RfPos(:, :, js+1));
                eL = tr2diff(fkine(LUpperBody, qL), LfPos(:, :, js+1));

                dH = (Thetai(1,:) - q) ./ 25;

                dqw = diag(W * J * (J * diag(W * J)) ^ -1 * [eR; eL]);

                q = q + dqw;
                qR = qR + dqw(1:14);
```

Appendix B (Continued)

```
        qL = qL + dqw([1:3,15:25]);
        WLNTheta = [WLNTheta; q'];
    end

    cost = cost+sum(sum((WLNTheta - Thetai).^2));

    end
end
end
end
count = count+1;
time = toc;
hr = floor(time/3600);
min = floor((time-hr*3600)/60);
sec = floor(time-hr*3600-min*60);
disp([' Count: ',int2str(count), ' Cost: ',int2str(floor(cost)), ' Time: ',int2str(hr),':',int2str(min),':',int2str(sec)]);
end
```

% Global Weights from last run (1/31/2011)

```
Wo = [0.156056640625000;
0.062400390625000;
0.226298828125000;
0.756981573425592;
0.721896484375000;
0.267538381283042;
0.431171875000000;
0.472146484375000;
0.774578125000000;
0.938476562500000;
0.062400390625000;
0.938476562500000;
0.565802734375000;
0.815552734375000;
0.680921875000000;
0.319955078125000;
0.416129956297255;
0.692628906250000;
0.276103209395768;
0.844820312500000;
0.938476562500000;
0.062400390625000;
0.938476562500000;
0.472146484375000;
```

Appendix B (Continued)

```
0.542346181522845];

if nargout>1
    W = fmincon(@(x) errorfun(x), Wo,[],[],[],[],lb,ub,[],options);
end

for s=1:size(fieldnames(Train),1)

    if strcmpi(Dominant(s),'R')
        Wfixed = W(1:25);
    elseif strcmpi(Dominant(s),'L')
        Wfixed = [W(1:3);
                  W(15:25);
                  W(4:14)];
    end

    diagW = diag(Wfixed);

    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    for t=1:(size(names,1)-2)
        name = char(names(t,:));
        sections = fieldnames(Train.(subject).(name));

        for i=3:size(sections,1)
            section = char(sections(i,:));
            if isfield(Train.(subject).(name).(section),'Theta')

                Thetai = Train.(subject).(name).(section).Theta;
                sSize = size(Thetai,1);
                if (sSize<=15)
                    %disp([name, section, 'is to short']);
                    continue
                end

                RThetai = Thetai(:,1:14);
                LThetai = Thetai(:,[1:3,15:25]);
                LThetai(:,3) = LThetai(:,3)+ Train.(subject).(name).(section).dT3;
```

Appendix B (Continued)

```
RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

qR = RThetai(1,:);
qL = LThetai(1,:);
q = Thetai(1,:);

GloWLNTheta = Thetai(1,:);

for j=1:(size(RThetai,1)-1)
    JR = jacob0(RUpperBody, qR);
    JL = jacob0(LUpperBody, qL);
    J = JR;
    J(7:12,1:3) = JL(:,1:3);
    J(7:12,15:25) = JL(:,4:14);

    eR = tr2diff(fkine(RUpperBody, qR), RfPos(:,j+1));
    eL = tr2diff(fkine(LUpperBody, qL), LfPos(:,j+1));

    dqw = diagW*J*(J*diagW*J)^-1 * [eR; eL];

    q = q + dqw;
    qR = qR + dqw(1:14);
    qL = qL + dqw([1:3,15:25]);
    GloWLNTheta = [GloWLNTheta; q];
end
Train.(subject).(name).(section).GloWLNTheta = GloWLNTheta;
Train.(subject).(name).(section).ErrorGloWLNTheta = (GloWLNTheta -
Thetai).^2;

end
end
end
end

end % Test Fuction
```

B.25 SubFunctions\TestBiGP.m

% Probability Density Gradient Projection

% RHBM 2/13/2011

```
function [Train, Rv, Rqt, Lv, Lqt, u] = TestBiGP(Train, T, inc, ROMstr)
```

```
Tini = T;
```

% Set the minimum number of points per interval

```
MC = 1;
```

Appendix B (Continued)

```
function [xp, yp, zp] = RgetEEpos(xi,yi,zi)
for b=1:inc
    if all([Rx(b)<=xi; xi<=Rx(b+1)]);
        xp = b;
    elseif xi<=Rx(1);
        xp = 1;
    elseif Rx(inc)<=xi;
        xp = inc;
    end

    if all([Ry(b)<=yi; yi<=Ry(b+1)]);
        yp = b;
    elseif yi<=Ry(1);
        yp = 1;
    elseif Ry(inc)<=yi;
        yp = inc;
    end

    if all([Rz(b)<=zi; zi<=Rz(b+1)]);
        zp = b;
    elseif zi<=Rz(1);
        zp = 1;
    elseif Rz(inc)<=zi;
        zp = inc;
    end
end
end
```

```
function [xp, yp, zp] = LgetEEpos(xi,yi,zi)
for b=1:inc
    if all([Lx(b)<=xi; xi<=Lx(b+1)]);
        xp = b;
    elseif xi<=Lx(1);
        xp = 1;
    elseif Lx(inc)<=xi;
        xp = inc;
    end

    if all([Ly(b)<=yi; yi<=Ly(b+1)]);
        yp = b;
    elseif yi<=Ly(1);
        yp = 1;
    elseif Ly(inc)<=yi;
        yp = inc;
    end
```


Appendix B (Continued)

```
end

if all([Lz(b)<=zi; zi<=Lz(b+1)]);
    zp = b;
elseif zi<=Lz(1);
    zp = 1;
elseif Lz(inc)<=zi;
    zp = inc;
end
end
end

subjects = fieldnames(Train);
for s=1:size(fieldnames(Train),1)

    subject = char(subjects(s,:))
    names = fieldnames(Train.(subject));
    ROM = ROMstr.(subject);

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    RDH = RUpperBody.dh;
    LDH = LUpperBody.dh;
    dT3 = LDH(3,3)-RDH(3,3);

    if nargin==4

        MinR = ROMstr.(subject)(1:11,1)*ones(1,size(Tini,2));
        MaxR = ROMstr.(subject)(1:11,2)*ones(1,size(Tini,2));

        MinL = ROMstr.(subject)([1:3,15:22],1)*ones(1,size(Tini,2));
        MaxL = ROMstr.(subject)([1:3,15:22],2)*ones(1,size(Tini,2));

        RT = Tini(1:14,all([Tini([1:11],:)>MinR;Tini([1:11],:)<MaxR],1));
        LT = Tini([1:3,15:25],all([Tini([1:3,15:22],:)>MinL;Tini([1:3,15:22],:)<MaxL],1));
    else

        RT = Tini(1:14,:);
        LT = Tini([1:3,15:25],:);
    end

    if (size(RT,2)>1)&&(size(LT,2)>1)
```

Appendix B (Continued)

```
RTr = RT';
LTr = LT';
LTr(:,3) = LTr(:,3) + dT3;

RPh = fkine(RUpperBody, RTr);
LPh = fkine(LUpperBody, LTr);

RP=[];
LP=[];

RP(:,:) = RPh(1:3,4,:);
LP(:,:) = LPh(1:3,4,:);

RPmax = max(RP');
RPmin = min(RP');
RPdif = RPmax - RPmin;
RPinc = (RPdif/inc)-0.001;

LPmax = max(LP');
LPmin = min(LP');
LPdif = LPmax - LPmin;
LPinc = (LPdif/inc)-0.001;

Rx = RPmin(1):RPinc(1):RPmax(1);
Ry = RPmin(2):RPinc(2):RPmax(2);
Rz = RPmin(3):RPinc(3):RPmax(3);

Lx = LPmin(1):LPinc(1):LPmax(1);
Ly = LPmin(2):LPinc(2):LPmax(2);
Lz = LPmin(3):LPinc(3):LPmax(3);

Rv = zeros(14, 99, inc, inc, inc);
Rqt = zeros(14, 99, inc, inc, inc);

Lv = zeros(14, 99, inc, inc, inc);
Lqt = zeros(14, 99, inc, inc, inc);

for p=1:size(T,1)
    [Gf(p,:), Gqi(p,:), u(p)] = ksdensity(T(p,:));
    while any(isnan(diff(Gf(p,:),.^-1)))
        [Gf(p,:), Gqi(p,:), u(p)] = ksdensity(T(p,:), 'width', u(p)*2);
    end
    Gqt(p,:) = (Gqi(p,2:end)+Gqi(p,1:end-1))/2;
    Gv(p,:) = diff(Gf(p,:),.^-1);
```

Appendix B (Continued)

end

for i=1:inc

 for j=1:inc

 for k=1:inc

 Rtheta = RT(:, all([(Rx(i))<=RP(1,:); RP(1,:)<=(Rx(i+1));
 (Ry(j))<=RP(2,:); RP(2,:)<=(Ry(j+1));
 (Rz(k))<=RP(3,:); RP(3,:)<=(Rz(k+1))]));

 Ltheta = LT(:, all([(Lx(i))<=LP(1,:); LP(1,:)<=(Lx(i+1));
 (Ly(j))<=LP(2,:); LP(2,:)<=(Ly(j+1));
 (Lz(k))<=LP(3,:); LP(3,:)<=(Lz(k+1))]));

 Rcount = size(Rtheta,2);

 Lcount = size(Ltheta,2);

 Rconf(i,j,k) = Rcount;

 Lconf(i,j,k) = Lcount;

 if Rcount>=MC;

 f = [];

 qi = [];

 for p=1:14

 [f(p,:), qi(p,:)] = ksdensity(Rtheta(p,:), 'width', u(p));

 if any(isnan(diff(f(p,:).^-1)))

 f(p,:) = Gf(p,:);

 qi(p,:) = Gqi(p,:);

 end

 Rqt(p,:,i,j,k) = (qi(p,2:end)+qi(p,1:end-1))/2;

 if all(Rqt(p,:,i,j,k)==0)

 disp('Error Rqt Zero ');

 [p,i,j,k];

 end

 Rv(p,:,i,j,k) = diff(f(p,:).^-1);

 end

 end

if Lcount>=MC;

 f = [];

 qi = [];

 for p=1:14

Appendix B (Continued)

```
[f(p,:), qi(p,:)] = ksdensity(Ltheta(p,:), 'width', u(p));
if any(isnan(diff(f(p,:).^-1)))
    if p>3
        Lp = p+11;
    else
        Lp = p;
    end
    f(p,:) = Gf(Lp,:);
    qi(p,:) = Gqi(Lp,:);
end

Lqt(p,:,i,j,k) = (qi(p,2:end)+qi(p,1:end-1))/2;
if all(Lqt(p,:,i,j,k)==0)
    disp('Error Lqt Zero ')
    [p,i,j,k]
end

Lv(p,:,i,j,k) = diff(f(p,:).^-1);
end
end

end % Inc k
end % Inc j
end % Inc i

% Maximum gradient vector (for stability of solution)
maxG = 2;

wH = diag(ones(25, 1)*0.05);

for t=1:(size(names,1)-2)
    name = char(names(t,:));
    sections = fieldnames(Train.(subject).(name));

    for i=3:size(sections,1)
        section = char(sections(i,:));

        if isfield(Train.(subject).(name).(section), 'Theta')
            Thetai = Train.(subject).(name).(section).Theta;
            sSize = size(Thetai,1);
            if (sSize<=15)
                [name, section, 'is to short'];
                continue
            end
        end
    end
end
```

Appendix B (Continued)

```
RThetai = Thetai(:,1:14);
LThetai = Thetai(:,[1:3,15:25]);
LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;
```

```
RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);
```

```
Rq = RThetai(1,:);
Lq = LThetai(1,:);
q = Thetai(1,:);
```

```
H = [];
ProbTheta = Thetai(1,:);
```

```
for j=1:(sSize-1)
```

```
    RJ = jacob0(RUpperBody, Rq);
    LJ = jacob0(LUpperBody, Lq);
```

```
    J = RJ;
    J(7:12,1:3) = LJ(:,1:3);
    J(7:12,15:25) = LJ(:,4:14);
```

```
    Re = tr2diff(fkine(RUpperBody, Rq), RfPos(:,j+1));
    Le = tr2diff(fkine(LUpperBody, Lq), LfPos(:,j+1));
```

```
    for p=1:14
```

```
        [xp, yp, zp] = RgetEEpos(RfPos(1,4,j+1), RfPos(2,4,j+1), RfPos(3,4,j+1));
        if (~all(Rqt(p, :, xp, yp, zp)==0))&&(Rconf(xp,yp,zp)>=MC)
            RdH(p) = -interp1(Rqt(p, :, xp, yp, zp), Rv(p, :, xp, yp, zp), q(p), 'spline',
```

```
'extrap');
```

```
        else
```

```
            %disp(['Rqt == 0 or Rcount < ', num2str(MC),'for joint', num2str(p)])
```

```
            % [p, xp, yp, zp]
```

```
            RdH(p) = -interp1(Gqt(p, :), Gv(p, :), q(p), 'spline', 'extrap');
```

```
        end
```

```
    % Error Checking
```

```
    if isnan(RdH(p))
```

```
        disp(['Nan in R interperlation', num2str(p)])
```

```
        [p, xp, yp, zp]
```

```
        RdH(p) = -interp1(Gqt(p, :), Gv(p, :), q(p), 'spline', 'extrap');
```

```
    if isnan(RdH(p))
```

```
        disp(['Nan in R Global interperlation', num2str(p)])
```

Appendix B (Continued)

```
        RdH(p) = 0;
    end
end
% End of Error Checking
end

for p=1:14
    if p>3
        Lp = p+11;
    else
        Lp = p;
    end
    [xp, yp, zp] = LgetEEpos(LfPos(1,4,j+1), LfPos(2,4,j+1), LfPos(3,4,j+1));
    if (~all(Lqt(p, :, xp, yp, zp)==0))&&(Lconf(xp,yp,zp)>=MC)
        LdH(p) = -interp1(Lqt(p, :, xp, yp, zp), Lv(p, :, xp, yp, zp), q(Lp), 'spline',
'extrap');
    else
        % disp(['Lqt == 0 or Lcount < ', num2str(MC),' for joint ', num2str(Lp)])
        % [p, xp, yp, zp]
        LdH(p) = -interp1(Gqt(Lp, :), Gv(Lp, :), q(Lp), 'spline', 'extrap');
    end

    % Error Checking
    if isnan(LdH(p))
        disp(['Nan in L interperlation', num2str(Lp)])
        [p, xp, yp, zp]
        LdH(p) = -interp1(Gqt(Lp, :), Gv(Lp, :), q(Lp), 'spline', 'extrap');
        if isnan(LdH(p))
            disp(['Nan in L Global interperlation', num2str(Lp)])
            LdH(p) = 0;
        end
    end
    % End of Error Checking
end

dH = [RdH(1:3)+LdH(1:3), RdH(4:14), LdH(4:14)];

for p=1:25
    if dH(p) > maxG
        dH(p) = maxG;
    elseif dH(p) < -maxG
        dH(p) = -maxG;
    end
end
```

Appendix B (Continued)

```
H = [H;dH];

dq = J*(J*J')^-1 * [Re; Le] + (eye(25) - J*(J*J')^-1*J) * wH * dH';

q = q + dq;

Rq = Rq + dq(1:14);
Lq = Lq + dq([1:3,15:25]);

ProbTheta = [ProbTheta; q'];

end

Train.(subject).(name).(section).H = H;
Train.(subject).(name).(section).ProbTheta = ProbTheta;
Train.(subject).(name).(section).ErrorProb = (ProbTheta - Thetai).^2;

end % If Right

end % Section

end % Trials
else
disp([subject,'Not enough matching joint angles'])
maxG = 2;
wH = diag(ones(25, 1)*0.05);
for t=1:(size(names,1)-2)
name = char(names(t,:));
sections = fieldnames(Train.(subject).(name));

for i=3:size(sections,1)
section = char(sections(i,:));

if isfield(Train.(subject).(name).(section),'Theta')
Thetai = Train.(subject).(name).(section).Theta;
sSize = size(Thetai,1);
if (sSize<=15)
[name, section, 'is to short'];
continue
end

RThetai = Thetai(:,1:14);
LThetai = Thetai(:,[1:3,15:25]);
LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;
```

Appendix B (Continued)

```
RfPos = fkine(RUpperBody, RThetai);
LfPos = fkine(LUpperBody, LThetai);

Rq = RThetai(1,:);
Lq = LThetai(1,:);
q = Thetai(1,:);

H = [];
ProbTheta = Thetai(1,:);

for j=1:(sSize-1)

    RJ = jacob0(RUpperBody, Rq);
    LJ = jacob0(LUpperBody, Lq);

    J = RJ;
    J(7:12,1:3) = LJ(:,1:3);
    J(7:12,15:25) = LJ(:,4:14);

    Re = tr2diff(fkine(RUpperBody, Rq), RfPos(:,j+1));
    Le = tr2diff(fkine(LUpperBody, Lq), LfPos(:,j+1));

    for p = 1:25;
        dH(p) = -0.05*((ROM(p,2)-ROM(p,1))^2 * (2*q(p)-ROM(p,2)-
ROM(p,1)))/(4*(ROM(p,2)-q(p))^2*(q(p)-ROM(p,1))^2);
    end

    for p=1:25
        if (dH(p) < -maxG)|| (q(p)>ROM(p,2))
            dH(p) = -maxG;
        elseif (dH(p) > maxG)|| (q(p)<ROM(p,1))
            dH(p) = maxG;
        end
    end

    H = [H;dH];

    dq = J*(J*J')^-1 * [Re; Le] + (eye(25) - J*(J*J')^-1*J) * wH * dH';

    q = q + dq;

    Rq = Rq + dq(1:14);
    Lq = Lq + dq([1:3,15:25]);
```


Appendix B (Continued)

```
        ProbTheta = [ProbTheta; q];

    end

    Train.(subject).(name).(section).H = H;
    Train.(subject).(name).(section).ProbTheta = ProbTheta;
    Train.(subject).(name).(section).ErrorProb = (ProbTheta - Thetai).^2;

end % If Right

end % Section

end % Trials
end

end % Subjects

end % function
```

B.26 SubFunctions\TestBiNN.m

% Bilateral Neural Network Testing Algorithgm

% RHBM 2/6/2011

function Train = TestBiNN(Train, T, P, n1)

% Number of neurons in the hidden layer

% n1

% Using training data input P, and output T

% P is an n by i matrix where n is the number of input neurons and i is the
% number of data points

% T is an m by i matrix where m is the number of ouput neurson and i is the
% number of data points

% Create neural network with one hidden layer

Binet = newff(P, T, n1, {'tansig'}, 'trainlm');

% Specify the training function

Binet.trainFcn = 'trainlm';

% Sets the number of training epochs

Binet.trainParam.epochs = 50;

% Specify the memory reduction (use if training set is large)

Binet.trainParam.mem_reduc = 10;

% Train the network with the training data

[Binet] = train(Binet, P, T);

Appendix B (Continued)

```
for s=1:size(fieldnames(Train),1);
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    for t=1:(size(names,1))
        name = char(names(t,:));
        if strcmpi(name, 'RUpperBody')||strcmpi(name, 'LUpperBody')
            continue
        end
        sections = fieldnames(Train.(subject).(name));

        for j=3:size(sections,1)
            section = char(sections(j,:));

            Thetai = Train.(subject).(name).(section).Theta;

            if (size(Thetai,1)<=15)
                disp([name, section, 'is to short']);
                continue
            end

            % Use the network to find Theta
            if strcmpi(subject(1),'B')
                NNTheta = sim(Binet, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP; ones(1,size(Thetai,1))]);
            else
                NNTheta = sim(Binet, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP; ones(1,size(Thetai,1))]);
            end

            % Store results and calculate error squared
            Train.(subject).(name).(section).ThetaNN = NNTheta;
            Train.(subject).(name).(section).ErrorNN = (Thetai-NNTheta).^2;

            RThetaNN = NNTheta(:,1:14);
            LThetaNN = NNTheta(:,[1:3,15:25]);
            LThetaNN(:,3) = LThetaNN(:,3)+ Train.(subject).(name).(section).dT3;

            Train.(subject).(name).(section).RThetaNN = RThetaNN;
            Train.(subject).(name).(section).LThetaNN = LThetaNN;
```

Appendix B (Continued)

```
% Calculate End Effector Position
RfPos = fkine(RUpperBody, RThetaNN);
LfPos = fkine(LUpperBody, LThetaNN);

% Reset end effector position for loop calculations
RP = [];
RP(:, :) = RfPos(1:3,4,:);

% Reset end effector position for loop calculations
LP = [];
LP(:, :) = LfPos(1:3,4,:);

% Calculate the end effector error
Train.(subject).(name).(section).ErrorNN_EE =
[sum((Train.(subject).(name).(section).RP(1:3,:)-RP).^2);
sum((Train.(subject).(name).(section).LP(1:3,:)-LP).^2)];

end

end % Trials

end % Subjects

end

B.27 SubFunctions\TestBiGP_WLN.m
% Probability Density Gradient Projection + Weighted Leat Norm
% RHBM 2/9/2011
function [Train, Rv, Rqt, Lv, Lqt, u] = TestBiGP_WLN(Train, T, inc, Weights)

RT = T(1:14,:);
LTini = T([1:3,15:25],:);

Dominant = ['R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'L'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; ...
            'R'; 'L'; 'R'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'];

% Set the minimum number of points per interval
MC = 1;

function [xp, yp, zp] = RgetEEpos(xi,yi,zi)

for b=1:inc
    if all([Rx(b)<=xi; xi<=Rx(b+1)]);
```

Appendix B (Continued)

```
    xp = b;
elseif xi<=Rx(1);
    xp = 1;
elseif Rx(inc)<=xi;
    xp = inc;
end

if all([Ry(b)<=yi; yi<=Ry(b+1)]);
    yp = b;
elseif yi<=Ry(1);
    yp = 1;
elseif Ry(inc)<=yi;
    yp = inc;
end

if all([Rz(b)<=zi; zi<=Rz(b+1)]);
    zp = b;
elseif zi<=Rz(1);
    zp = 1;
elseif Rz(inc)<=zi;
    zp = inc;
end

end
end

function [xp, yp, zp] = LgetEEpos(xi,yi,zi)

for b=1:inc
    if all([Lx(b)<=xi; xi<=Lx(b+1)]);
        xp = b;
    elseif xi<=Lx(1);
        xp = 1;
    elseif Lx(inc)<=xi;
        xp = inc;
    end

    if all([Ly(b)<=yi; yi<=Ly(b+1)]);
        yp = b;
    elseif yi<=Ly(1);
        yp = 1;
    elseif Ly(inc)<=yi;
        yp = inc;
    end
```

Appendix B (Continued)

```
end

if all([Lz(b)<=zi; zi<=Lz(b+1)]);
    zp = b;
elseif zi<=Lz(1);
    zp = 1;
elseif Lz(inc)<=zi;
    zp = inc;
end
end
end

subjects = fieldnames(Train);
for s=1:size(fieldnames(Train),1)

    subject = char(subjects(s,:))
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
    LUpperBody = Train.(subject).LUpperBody;

    RDH = RUpperBody.dh;
    LDH = LUpperBody.dh;
    dT3 = LDH(3,3)-RDH(3,3);

    LT = LTini;
    LT(:,3) = LT(:,3) + dT3;

    RPh = fkine(RUpperBody, RT);
    LPh = fkine(LUpperBody, LT);

    RP(:, :) = RPh(1:3,4,:);
    LP(:, :) = LPh(1:3,4,:);

    RPmax = max(RP');
    RPmin = min(RP');
    RPdif = RPmax - RPmin;
    RPinc = (RPdif/inc)-0.001;

    LPmax = max(LP');
    LPmin = min(LP');
    LPdif = LPmax - LPmin;
    LPinc = (LPdif/inc)-0.001;
```

Appendix B (Continued)

```
Rx = RPmin(1):RPinc(1):RPmax(1);  
Ry = RPmin(2):RPinc(2):RPmax(2);  
Rz = RPmin(3):RPinc(3):RPmax(3);
```

```
Lx = LPmin(1):LPinc(1):LPmax(1);  
Ly = LPmin(2):LPinc(2):LPmax(2);  
Lz = LPmin(3):LPinc(3):LPmax(3);
```

```
Rv = zeros(14, 99, inc, inc, inc);  
Rqt = zeros(14, 99, inc, inc, inc);
```

```
Lv = zeros(14, 99, inc, inc, inc);  
Lqt = zeros(14, 99, inc, inc, inc);
```

```
for p=1:size(T,1)  
    [Gf(p,:), Gqi(p,:), u(p)] = ksdensity(T(p,:));  
    while any(isnan(diff(Gf(p,:).^-1)))  
        [Gf(p,:), Gqi(p,:), u(p)] = ksdensity(T(p,:), 'width', u(p)*2);  
    end  
    Gqt(p,:) = (Gqi(p,2:end)+Gqi(p,1:end-1))/2;  
    Gv(p,:) = diff(Gf(p,:).^-1);  
end
```

```
for i=1:inc  
    for j=1:inc  
        for k=1:inc
```

```
            Rtheta = T(1:14, all([(Rx(i))<=RP(1,:); RP(1,:)<=(Rx(i+1));  
                                (Ry(j))<=RP(2,:); RP(2,:)<=(Ry(j+1));  
                                (Rz(k))<=RP(3,:); RP(3,:)<=(Rz(k+1))]));
```

```
            Ltheta = T([1:3,15:25], all([(Lx(i))<=LP(1,:); LP(1,:)<=(Lx(i+1));  
                                        (Ly(j))<=LP(2,:); LP(2,:)<=(Ly(j+1));  
                                        (Lz(k))<=LP(3,:); LP(3,:)<=(Lz(k+1))]));
```

```
            Rcount = size(Rtheta,2);  
            Lcount = size(Ltheta,2);
```

```
            Rconf(i,j,k) = Rcount;  
            Lconf(i,j,k) = Lcount;
```

```
        if Rcount>=MC;  
            f = [];  
            qi = [];
```

Appendix B (Continued)

```
for p=1:14
    [f(p,:), qi(p,:)] = ksdensity(Rtheta(p,:), 'width', u(p));
    if any(isnan(diff(f(p,:).^-1)))
        disp('NaN in density function generation')
        f(p,:) = Gf(p,:);
        qi(p,:) = Gqi(p,:);
    end

    Rqt(p,:,i,j,k) = (qi(p,2:end)+qi(p,1:end-1))/2;
    if all(Rqt(p,:,i,j,k)==0)
        disp('Error Rqt Zero ')
        [p,i,j,k];
    end

    Rv(p,:,i,j,k) = diff(f(p,:).^-1);
end
end

if Lcount>=MC;
    f = [];
    qi = [];
    for p=1:14
        [f(p,:), qi(p,:)] = ksdensity(Ltheta(p,:), 'width', u(p));
        if any(isnan(diff(f(p,:).^-1)))
            disp('NaN in density function generation')
            if p>3
                Lp = p+11;
            else
                Lp = p;
            end
            f(p,:) = Gf(Lp,:);
            qi(p,:) = Gqi(Lp,:);
        end

        Lqt(p,:,i,j,k) = (qi(p,2:end)+qi(p,1:end-1))/2;
        if all(Lqt(p,:,i,j,k)==0)
            disp('Error Lqt Zero ')
            [p,i,j,k]
        end

        Lv(p,:,i,j,k) = diff(f(p,:).^-1);
    end
end
```

Appendix B (Continued)

```
    end % Inc k
  end % Inc j
end % Inc i

% Maximum gradient vector (for stability of solution)
maxG = 2;

if strcmpi(Dominant(s), 'R')
  Wfixed = Weights(1:25);
elseif strcmpi(Dominant(s), 'L')
  Wfixed = [Weights(1:3);
            Weights(15:25);
            Weights(4:14)];
end

W = diag(Wfixed);

wH = diag(ones(25, 1)*0.05);

for t=1:(size(names,1)-2)
  name = char(names(t,:));
  sections = fieldnames(Train.(subject).(name));

  for i=3:size(sections,1)
    section = char(sections(i,:));

    if isfield(Train.(subject).(name).(section), 'Theta')
      Thetai = Train.(subject).(name).(section).Theta;
      sSize = size(Thetai,1);
      if (sSize<=15)
        [name, section, 'is to short'];
        continue
      end

      RThetai = Thetai(:,1:14);
      LThetai = Thetai(:,[1:3,15:25]);
      LThetai(:,3) = LThetai(:,3) + Train.(subject).(name).(section).dT3;

      RfPos = fkine(RUpperBody, RThetai);
      LfPos = fkine(LUpperBody, LThetai);

      Rq = RThetai(1,:);
      Lq = LThetai(1,:);
      q = Thetai(1,:);
    end
  end
end
```


Appendix B (Continued)

```
H = [];  
ProbTheta = Thetai(1,:);  
  
for j=1:(sSize-1)  
  
    RJ = jacob0(RUpperBody, Rq);  
    LJ = jacob0(LUpperBody, Lq);  
  
    J = RJ;  
    J(7:12,1:3) = LJ(:,1:3);  
    J(7:12,15:25) = LJ(:,4:14);  
  
    Re = tr2diff(fkine(RUpperBody, Rq), RfPos(:, :, j+1));  
    Le = tr2diff(fkine(LUpperBody, Lq), LfPos(:, :, j+1));  
  
    [xp, yp, zp] = RgetEEpos(RfPos(1,4,j+1), RfPos(2,4,j+1), RfPos(3,4,j+1));  
    for p=1:14  
  
        if (~all(Rqt(p, :, xp, yp, zp)==0))&&(Rconf(xp,yp,zp)>=MC)  
            RdH(p) = -interp1(Rqt(p, :, xp, yp, zp), Rv(p, :, xp, yp, zp), q(p), 'spline',  
'extrap');  
        else  
            % disp(['Rqt == 0 or Rcount < ', num2str(MC), 'for joint', num2str(p)])  
            % [p, xp, yp, zp]  
            RdH(p) = -interp1(Gqt(p, :), Gv(p, :), q(p), 'spline', 'extrap');  
        end  
  
        % Error Checking  
        if isnan(RdH(p))  
            disp(['Nan in R interpolation', num2str(p)])  
            RdH(p) = 0;  
        end  
        % End of Error Checking  
    end  
  
    [xp, yp, zp] = LgetEEpos(LfPos(1,4,j+1), LfPos(2,4,j+1), LfPos(3,4,j+1));  
    for p=1:14  
        if p>3  
            Lp = p+11;  
        else  
            Lp = p;  
        end  
  
        if (~all(Lqt(p, :, xp, yp, zp)==0))&&(Lconf(xp,yp,zp)>=MC)
```

Appendix B (Continued)

```
LdH(p) = -interp1(Lqt(p, :, xp, yp, zp), Lv(p, :, xp, yp, zp), q(Lp), 'spline',
'extrap');
else
    % disp(['Lqt == 0 or Lcount < ', num2str(MC),' for joint ', num2str(Lp)])
    % [p, xp, yp, zp]
    LdH(p) = -interp1(Gqt(Lp, :), Gv(Lp, :), q(Lp), 'spline', 'extrap');
end

% Error Checking
if isnan(LdH(p))
    disp(['Nan in L interpolation', num2str(Lp)])
    LdH(p) = 0;
end
end

dH = [RdH(1:3)+LdH(1:3), RdH(4:14), LdH(4:14)];

for p=1:25;
    if dH(p) > maxG
        dH(p) = maxG;
    elseif dH(p) < -maxG
        dH(p) = -maxG;
    end
end

H = [H;dH];

dq = W*J*(J*W*J)^-1 * [Re; Le] + (eye(25) - pinv(J)*J) * wH * dH';

q = q + dq;

Rq = Rq + dq(1:14);
Lq = Lq + dq([1:3,15:25]);

ProbTheta = [ProbTheta; q];

end

Train.(subject).(name).(section).H = H;
Train.(subject).(name).(section).ProbTheta = ProbTheta;
Train.(subject).(name).(section).ErrorProb = (ProbTheta - Thetai).^2;

end % Is field
```

Appendix B (Continued)

```
end % Section

end % Trials

end % Subjects

end % function

B.28 SubFunctions\TestBiNN_WLN.m
% Bilateral Neural Network + Weighted Least Norm Testing Algorithm
% RHBM 2/9/2011
function Train = TestBiNN_WLN(Train, T, P, n1, Weights)

Dominant = ['R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'R'; 'R'; 'L'; 'R'; 'R'; 'L'; 'R'; 'R'; 'R'; 'R'; ...
            'L'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; ...
            'R'; 'L'; 'R'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'; 'X'];

% Number of neurons in the hidden layer
% n1
% Using training data input P, and output T
% P is an n by i matrix where n is the number of input neurons and i is the
% number of data points
% T is an m by i matrix where m is the number of output neuron and i is the
% number of data points

% Create neural network with one hidden layer
Binet = newff(P, T, n1, {'tansig'}, 'trainlm');

% Specify the training function
Binet.trainFcn = 'trainlm';
% Sets the number of training epochs
Binet.trainParam.epochs = 50;
% Specify the memory reduction (use if training set is large)
Binet.trainParam.mem_reduc = 1;

% Train the network with the training data
[Binet] = train(Binet, P, T);

for s=1:size(fieldnames(Train),1);
    subjects = fieldnames(Train);
    subject = char(subjects(s,:));
    names = fieldnames(Train.(subject));

    RUpperBody = Train.(subject).RUpperBody;
```

Appendix B (Continued)

```
LUpperBody = Train.(subject).LUpperBody;

if strcmpi(Dominant(s),'R')
    Wfixed = Weights(1:25);
elseif strcmpi(Dominant(s),'L')
    Wfixed = [Weights(1:3);
             Weights(15:25);
             Weights(4:14)];
end

W = diag(Wfixed);

for t=1:(size(names,1))
    name = char(names(t,:));
    if strcmpi(name, 'RUpperBody')||strcmpi(name, 'LUpperBody')
        continue
    end
    sections = fieldnames(Train.(subject).(name));

    for j=3:size(sections,1)
        section = char(sections(j,:));

        % Use the network to find Theta
        NNTheta = sim(Binet, [Train.(subject).(name).(section).RP;
Train.(subject).(name).(section).LP]);

        % Filter the Neural Network Results
        ThetaNN_WLN = WMAfilter(11, NNTheta);

        % Calculate right and left arm joint angles
        RThetaNN_WLN = ThetaNN_WLN(:,1:14);
        LThetaNN_WLN = ThetaNN_WLN(:,[1:3,15:25]);
        LThetaNN_WLN(:,3) = LThetaNN_WLN(:,3)+
Train.(subject).(name).(section).dT3;

        % Calculate testing data right and left arm joint angles
        Thetai = Train.(subject).(name).(section).Theta;
        RTheta = Thetai(:,1:14);
        LTheta = Thetai(:,[1:3,15:25]);
        LTheta(:,3) = LTheta(:,3)+ Train.(subject).(name).(section).dT3;

        % Calculate Forward Kinematic End Effector Position from filtered Neural Net
        RfPos = fkine(RUpperBody, RTheta);
        LfPos = fkine(LUpperBody, LTheta);
```

Appendix B (Continued)

```
tol = 0.01;

for i = 1:size(Thetai,1)
    dxR = tr2diff(fkine(RUpperBody, RThetaNN_WLN(i,:),), RfPos(:,i));
    dxL = tr2diff(fkine(LUpperBody, LThetaNN_WLN(i,:),), LfPos(:,i));
    dx = [dxR; dxL];

    % Repeat Loop Until End Effector Error is less than tol.
    while sum(abs(dx))>tol
        dxR = tr2diff(fkine(RUpperBody, RThetaNN_WLN(i,:),), RfPos(:,i));
        dxL = tr2diff(fkine(LUpperBody, LThetaNN_WLN(i,:),), LfPos(:,i));
        dx = [dxR; dxL];

        RJ = jacob0(RUpperBody, RThetaNN_WLN(i,:));
        LJ = jacob0(LUpperBody, LThetaNN_WLN(i,:));

        J = RJ;
        J(7:12,1:3) = LJ(:,1:3);
        J(7:12,15:25) = LJ(:,4:14);

        dqw = W*J*(J*W*J)^-1 * dx;

        RThetaNN_WLN(i,:) = RThetaNN_WLN(i,:) + dqw(1:14)';
        LThetaNN_WLN(i,:) = LThetaNN_WLN(i,:) + dqw([1:3,15:25])';
        ThetaNN_WLN(i,:) = ThetaNN_WLN(i,:) + dqw';
    end
end

% Store results and calculate error squared
Train.(subject).(name).(section).ThetaNN_WLN = ThetaNN_WLN;
Train.(subject).(name).(section).ErrorNN_WLN = (Thetai-ThetaNN_WLN).^2;

end

end % Trials

end % Subjects

end
```