

University of Southern Queensland  
Faculty of Engineering & Surveying

**A Biomechanical Analysis for Improved Ergonomics in  
Metal Detector Products using Trajectory Optimisation**

A dissertation submitted by

B. Smith

in fulfilment of the requirements of

**ENG4112 Research Project**

towards the degree of

**Bachelor of Mechanical Engineering**

Submitted: October, 2010

# Abstract

Along with enabling technologies that drive function, ergonomics has become an increasingly important factor in the mechanical design of a product. This is particularly relevant to the assessment of equipment that is used over extended periods of time. In many large hand-held devices, balance is an important aspect of ergonomics and mismatched inertial properties will likely result in user discomfort and injury. Organizations have a responsibility to understand the full impact of their products whilst ensuring satisfactory performance.

The focus of this study is to underpin the dynamics of human-product interaction with respect to metal detector products and to provide an analysis tool to better understand the impact of a design on the end user. This report details existing literature, the technical approach to the study, objectives, and methodologies. Presented is both a quantitative and qualitative analysis of the impacts of macro-movement ergonomics (good and bad) on the end-users of metal-detector products by modelling dynamics of the human-product interaction.

University of Southern Queensland  
Faculty of Engineering and Surveying

<b>ENG4111/2 <i>Research Project</i></b>
--

### **Limitations of Use**

The Council of the University of Southern Queensland, its Faculty of Engineering and Surveying, and the staff of the University of Southern Queensland, do not accept any responsibility for the truth, accuracy or completeness of material contained within or associated with this dissertation.

Persons using all or any part of this material do so at their own risk, and not at the risk of the Council of the University of Southern Queensland, its Faculty of Engineering and Surveying or the staff of the University of Southern Queensland.

This dissertation reports an educational exercise and has no purpose or validity beyond this exercise. The sole purpose of the course pair entitled “Research Project” is to contribute to the overall education within the student’s chosen degree program. This document, the associated hardware, software, drawings, and other material set out in the associated appendices should not be used for any other purpose: if they are so used, it is entirely at the risk of the user.

**Prof F Bullen**

Dean

Faculty of Engineering and Surveying

# Certification of Dissertation

I certify that the ideas, designs and experimental work, results, analyses and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.

I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

B. SMITH

0050059964

---

Signature

---

Date

# Acknowledgments

This thesis is as much the work of my wife Jessica, two sons Harrison and Lachlan and extended family for the enduring support and encouragement they have offered along the way.

Thanks is also due to my work colleagues for the valuable technical advice they have given. In particular, special acknowledgement goes to Dr. Eng-Leng Mah for his assistance with data acquisition.

Finally, I would like to thank my supervisor, Dr. Ahmad Sharifian for his excellent academic support and guidance throughout the project.

B. SMITH

*University of Southern Queensland*

*October 2010*

# Contents

<b>Abstract</b>	<b>i</b>
<b>Acknowledgments</b>	<b>iv</b>
<b>List of Figures</b>	<b>x</b>
<b>List of Tables</b>	<b>xiv</b>
<b>Chapter 1 Introduction and Literature Review</b>	<b>1</b>
1.0.1 Background . . . . .	1
1.1 Review on Biomechanical Models . . . . .	3
1.2 Review on Numerical Methods for Biomechanics . . . . .	4
1.3 Review on Motion Capture Technologies . . . . .	6
1.3.1 Measuring Uncertainty: Optimal State Estimation . . . . .	7
1.4 Dissertation Overview . . . . .	7
<b>Chapter 2 Scope and Objectives</b>	<b>9</b>
2.1 Chapter Overview . . . . .	9

<b>CONTENTS</b>	<b>vi</b>
2.1.1 Hypothesis . . . . .	9
2.1.2 Specific Project Objectives . . . . .	10
2.2 Detailed Specifications . . . . .	10
2.2.1 Biomechanical Model . . . . .	10
2.2.2 Hardware . . . . .	11
2.2.3 Numerical Tools and Software . . . . .	12
<b>Chapter 3 Theory and Methodology</b>	<b>13</b>
3.1 Chapter Overview . . . . .	13
3.2 Kinematics . . . . .	13
3.2.1 Joint Definitions . . . . .	13
3.2.2 Homogenous Transformations . . . . .	14
3.3 Pseudo Motion Capture . . . . .	17
3.3.1 Experimental Methodology . . . . .	18
3.3.2 Force-Time Synchronisation . . . . .	19
3.4 Muscle-Actuator Dynamics . . . . .	21
3.4.1 Force-Velocity and Tension-Length Relationships . . . . .	21
3.4.2 Activation Dynamics . . . . .	22
3.4.3 Total Muscle Force . . . . .	22
3.4.4 PWM, Muscle Grouping, Parameters and Attachments . . . . .	23
3.4.5 State-Dependant Torque . . . . .	24

---

3.5	Rigid Body Dynamics . . . . .	26
3.5.1	Lagrangian Formulation . . . . .	27
3.5.2	State-Space Model . . . . .	28
<b>Chapter 4 Numerical Methods and Software Implementation</b>		<b>29</b>
4.1	Chapter Overview . . . . .	29
4.2	MBSysTran Implementation . . . . .	30
4.3	Trajectory Optimisation . . . . .	32
4.3.1	Objective Function . . . . .	33
4.3.2	Discretisation . . . . .	35
4.3.3	Constraints and Boundary Conditions . . . . .	35
4.3.4	Gauss Pseudo-Spectral Optimisation Software (GPOPS) . . . . .	37
4.3.5	Assessment Criteria . . . . .	39
4.4	Data Acquisition . . . . .	39
<b>Chapter 5 Hardware Design and Selection</b>		<b>40</b>
5.1	Chapter Overview . . . . .	40
5.2	Force-Gauge Network . . . . .	40
5.2.1	Resolving Orthogonal Components . . . . .	40
5.2.2	Sensor Selection . . . . .	43
5.3	Dummy Metal Detector . . . . .	44



---

5.4	Data Acquisition Hardware . . . . .	45
<b>Chapter 6 Results</b>		<b>48</b>
6.1	Chapter Overview . . . . .	48
6.2	Experiment 1 Results . . . . .	49
6.2.1	Shoulder Complex Activations . . . . .	49
6.2.2	Elbow Complex Activations . . . . .	50
6.2.3	Wrist Complex Activations . . . . .	51
6.3	Experiment 2 Results . . . . .	52
6.3.1	Shoulder Complex Activations . . . . .	52
6.3.2	Elbow Complex Activations . . . . .	53
6.3.3	Wrist Complex Activations . . . . .	54
6.4	Product Performance: Muscle-Activation Comparisons . . . . .	55
6.5	Sampling Window Effects . . . . .	56
<b>Chapter 7 Conclusions and Further Work</b>		<b>58</b>
7.1	Achievement of Project Objectives . . . . .	58
7.2	Future Work . . . . .	59
7.2.1	Measures of Accuracy . . . . .	59
7.2.2	Improvement of the Optimal Control Solution Procedure . . . . .	59
7.2.3	Implementation of Goal-Orientated Movement . . . . .	60

<b>CONTENTS</b>	<b>ix</b>
7.2.4 Assessment of Coil Movement Under Impact and Over Terrain Variations . . . . .	60
7.2.5 Implementation of Inertial Motion Sensing . . . . .	60
<b>References</b>	<b>63</b>
<b>Appendix A Project Specification</b>	<b>66</b>
<b>Appendix B Lumped Muscle Parameters</b>	<b>69</b>
<b>Appendix C Symbolic Functions of Moment Arm and Muscle Unit Vec- tors for DELT1</b>	<b>72</b>
<b>Appendix D Modified Symbolic Functions for the Shoulder, Elbow and Wrist Complex</b>	<b>76</b>
<b>Appendix E DAE Functions for the Shoulder, Elbow and Wrist Com- plex</b>	<b>84</b>
<b>Appendix F Datasheet of the Integrated Force-Gauges</b>	<b>91</b>
<b>Appendix G Datasheets of the Data-Acquisition System</b>	<b>93</b>
<b>Appendix H OCSP Convergence Statistics</b>	<b>97</b>

# List of Figures

1.1	Horizontal swing cycle and elevation of an electromagnetic coil above the ground. (C) Copyright Minelab Electronics 2010. . . . .	2
1.2	The core elements of a typical metal-detector product. (C) Copyright Minelab Electronics 2010. . . . .	3
1.3	Basic redundant musculotendon example over a single joint. . . . .	4
2.1	The biomechanical model simplified from Holzbaur, Murray & Delp (2005) used in this study. Only two musculotendon units are shown for simplicity. . . . .	11
2.2	A system diagram of the numerical approach used in this study and that which may be implemented in the future. . . . .	12
3.1	The 7DOF kinematic configuration of the interconnected shoulder, elbow and wrist complexes of the arm extremity. The links connecting joints within a complex have zero length. The links connecting complexes, in this case the upper arm and forearm, have lengths $L1$ and $L2$ . . . . .	14
3.2	The experimental procedure involves sampling a number of swing cycles between predefined target positions. The force-time histories at the handle and armrest are recorded and post-processed using MATLAB and LABVIEW. . . . .	18

3.3	Typical data for the force-time histories measured at the handle. The red lines indicate the window of time over which the data will be further processed. . . . .	19
3.4	A trigonometric fit (blue) of typical force-time (green) measurement at the handle. The resulting shoulder articulation (red) is <i>antiphase</i> to the fit. . . . .	20
3.5	A musculoskeletal model of 50 musculo-tendon compartments taken from Holzbaaur et al. (2005) (left) reduced to 11 (right). . . . .	24
3.6	Two muscle attachment points, one in the proximal and one in the distal frame, define the muscle line-of-action (left). The orthogonal distance from the joint centre to the muscle line-of-action is defined as the moment-arm. The vector diagram (right) represents the line-of-action and moment-arm about the joint. . . . .	25
4.1	MBSysPad graphical editor is used for constructing rigid-body models (CEREM 2008). The above model is that of the shoulder complex connected to the upper-arm with three mutually orthogonal rotary joints. . . . .	31
4.2	Flowchart of the MATLAB implementation of GPOPS . . . . .	38
4.3	A screenshot of the LABVIEW application for recording and saving of the force-time histories at the handle and armrest. . . . .	39
5.1	Integration of the force sensor network on the dummy detector. Forces are measured orthogonal to the hand and armrest. The unknown components are shown as dashed. . . . .	41
5.2	The unknown x force component at the armrest is found by taking moments of $\mathbf{F}_{1_x}$ and $\mathbf{F}_{2_x}$ about the product center-of-gravity. . . . .	41
5.3	The unknown y force component at the armrest is found by taking moments of $\mathbf{F}_{1_y}$ and $\mathbf{F}_{2_y}$ about the product center-of-gravity. . . . .	42

---

5.4	Off-axis (orthogonal) force components result in bending moments about the sensor base. . . . .	44
5.5	The LCM300 miniature load cell from Futek (Futek Advanced Sensor Technology 2010). . . . .	44
5.6	The NI USB 6210 (National Instruments Corporation n.d.) with 256kS/s sampling rate, 16 analog inputs, 4 digital inputs and 4 digital outputs. .	45
5.7	Itemised view of the top half of the metal-detector dummy. Refer to table 5.1 for a material list. . . . .	46
5.8	Itemised view of the bottom of the metal-detector dummy. Refer to table 5.2 for a material list. . . . .	47
6.1	Required muscle activations for the lumped muscles of the shoulder complex (experiment #1). . . . .	49
6.2	Required muscle activations for the lumped muscles of the elbow complex (experiment # 1). . . . .	50
6.3	Required muscle activations for the lumped muscles of the wrist complex (experiment # 1). . . . .	51
6.4	Required muscle activations for the lumped muscles of the shoulder complex (experiment #2). . . . .	52
6.5	Required muscle activations for the lumped muscles of the elbow complex (experiment # 2). . . . .	53
6.6	Required muscle activations for the lumped muscles of the wrist complex (experiment # 2). . . . .	54
6.7	Time histories of forces measured at the armrest(top) and handle(bottom). The red bars indicate the position and width of the sampling window. .	56

---

6.8	The same force-time data but the sampling window has shifted across to include the radical transients experienced at the armrest. . . . .	56
7.1	Flowchart of extended Kalman filter (EKF) with recursive Newton-Euler algorithm in-the-loop. . . . .	61

# List of Tables

5.1	Material list for upper half of dummy detector as shown in figure 5.7. . .	46
5.2	Material list for lower half of dummy detector as shown in figure 5.8. . .	47
6.1	Inertial configurations of two systems of equal mass but different arrangement. . . . .	49
6.2	The following table shows a comparison of evaluated activation integrals that form a basis for ergonomic assessment. . . . .	55
B.1	Muscle attachment data (Holzbaur et al. 2005) for major muscles used as lumped actuators. The attachment point coordinates (in meters) are with respect to the proximal and distal reference frames as indicated. . .	70

# Chapter 1

## Introduction and Literature Review

During the early stages of development, a number of constraints arising from functional, operational, cost and manufacturing requirements determine the course of a design. In many cases, the operational constraints (ergonomics) have traditionally had the least design influence while functional constraints (performance) have received the most focus. Ergonomics data is probably the most extensively utilized by the armed forces in the USA under MIL STD 1472 (Beavis & Slade 2003). However, the approach is only qualitative and a broader argument for benefits from improved ergonomics lacks quantitative backing.

This study aims to provide both a quantitative and qualitative analysis of the impacts of macro-movement ergonomics (good and bad) on the end-users of metal-detector products by modelling the dynamics of the human-product interaction.

### **1.0.1 Background**

Two significant markets exist for metal detector products. The first is a military market for mine-clearing and humanitarian assistance and the second is for the hobby and professional prospecting of coin, treasure and relics. In each market, the products are used



for extensive periods of time by a wide range of people in various environments. Proper use of the product requires repetitive and consistent movement of an electromagnetic coil over changing terrain. Soldiers may have to operate in harsh, hostile environments with extreme ranges of temperature for shifts of up to 12 hours at a time. The use of a metal detector for this extent of time will most likely result in repetitive strain injury (RSI) in the wrist or forearm and fatigue in the shoulder. In the case of a military application, fatigue induced by extensive field service may compromise the outcome of the operation if, as a result, explosive targets are missed with disastrous and tragic consequences. The effect of this strain and fatigue is for the subject to subsequently *hurry* the horizontal coil movement and also deviate from the ideal elevation of the coil above the ground. A representation of this movement is shown below in figure 1.1. Similarly, in the case of hobby or professional prospecting, targets of value will be missed if the trajectory of the detector is compromised as the user tires.



Figure 1.1: Horizontal swing cycle and elevation of an electromagnetic coil above the ground.

(C) Copyright Mimetab Electronics 2010.

Irrespective of the enabling detection technology in each case, the products have to be operated in a prescribed manner for maximum effectiveness. Thus, indirectly, the balance and weight of the product affects a measurement of performance with respect to the number of targets located. If a product is poorly balanced, field-time may be voluntarily reduced or user performance is compromised resulting in fewer target findings. Informal participation by field-testers in the assessment of prototypes has shown that the response to a product may be transient and subjective and initial perceptions do not necessarily hold for the full duration of the trial. This implies that a short ergonomic assessment upfront will not suffice in most cases for the purposes of design and that lengthy field evaluations are required. Therefore, a design and

analysis system that accurately models or predicts a response to the product early in the development phase would be of major benefit to the organisation and, ultimately, the end-user.

The core elements of a metal detector are shown in figure 1.2.



Figure 1.2: The core elements of a typical metal-detector product. (C) Copyright Minelab Electronics 2010.

## 1.1 Review on Biomechanical Models

Two factors that play a major role in the ergonomic interface to a metal-detector are the muscle-tendon metabolics (or work required) and joint loading. Field-testers have pin-pointed areas of discomfort in the arm and shoulder after extensive product use. As a quantitative assessment of the ergonomics, this study proposes that the distribution of upper extremity muscle activity required for a given cycle of movement may be used as a key indicator along with the joint loadings in the wrist, elbow and shoulder.

A vast array of literature exists with regard to musculoskeletal systems and modelling, the bulk of which is focussed towards neuroprosthesis and simulation of the effects of surgical procedure. Studies in functional neuromuscular stimulation (FNS) have exploited the electrical activation dynamics of muscles for control of movement (Chizeck, Crago & Kofman 1988). Substantial studies of the integration of Hill-type muscles and

tendons have been undertaken in a broader biomechanical scope (Zajak 1989). However literature focused on the direct application of these models to product ergonomics is relatively sparse. What is required is a model of broad applicability that relates externally applied forces and joint movements to muscle activity. Of particular relevance to this study is a comprehensive upper extremity model (Holzbaur et al. 2005) that has been comprehensively digitized for integration within a computational analysis program (Delp et al).

## 1.2 Review on Numerical Methods for Biomechanics

Dynamic simulation is a powerful method of investigating the interaction between muscles, bones and joints. However, due to the complexity of the muscle activation dynamics (Zajak 1989) and the redundancy of muscle groups, determining a set of muscle activations for a required movement pattern under loading is a complex and challenging task. A basic example of muscle redundancy over a single joint is represented by the planar arrangement in figure 1.3.

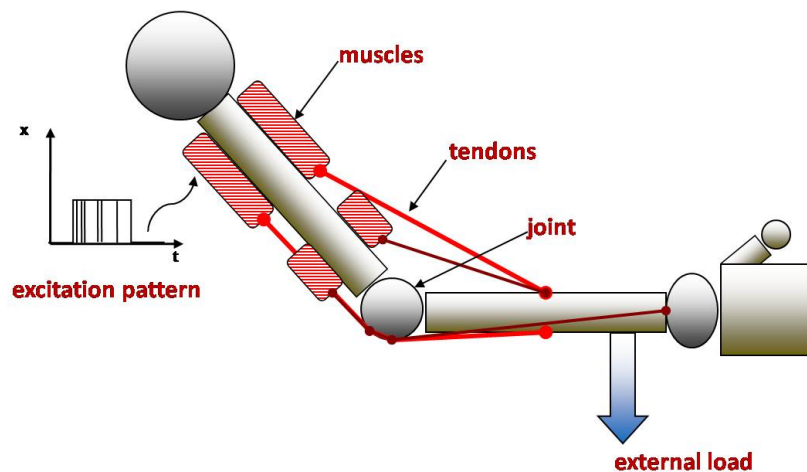


Figure 1.3: Basic redundant musculotendon example over a single joint.

A forward dynamics simulation will reveal the resultant motion trajectory from a given set of actuation forces and external loads whereas inverse dynamics will determine the joint torques from a given trajectory. In this study we attempt to find the desired muscle actuations for observed movement and external loading using forward dynamics and optimization. The required muscle activity is then compared by varying anthropometry

and different metal-detector configurations where observations of movement are made experimentally for each case. The two differing approaches to solving the redundant forward dynamics problem are static and dynamic optimization methods.

Despite these different optimization approaches, they both have common elements of a biomechanical forward simulation that include muscle activation and general multi-body system dynamics. The equations that define these subsets of the overall dynamics are summarized in chapter 3. The dynamic optimization approach formulates an optimal control problem (OCP) from the state equations where a functional integral cost function is minimized and a set of control and trajectory constraints are satisfied (Menegaldo, Fleury & Weber 2003); (Kaplan & Heegard 2002).

The static optimization approach is to solve for a set of muscle activations by considering only the steady-state forces that arise once each muscle system has reached equilibrium (Thelen, Anderson & Delp 2003). Actuator redundancy may be resolved by limiting activation and simultaneously enforcing the system acceleration constraints. The primary advantage of the static optimization method is the speed in which muscle-activated simulations of movement are computationally achieved. However, the limitations of this approach include latent inaccuracies concerning more rapid movements due to the delay between muscle excitations and the resultant muscle forces. Secondly, performance criteria can only be evaluated at each time-step and measures of performance over a range of time or movement (e.g. metabolic cost) cannot be done with static optimization.

It is for these reasons that a dynamic optimization approach will be adopted in this study for flexibility at the expense of computational cost. This flexibility will allow for future analysis of transient events like impacts, total mechanical energy expenditure and movement of the metal-detector coil through varying elements (e.g. sand, water, shrub etc).

---

## 1.3 Review on Motion Capture Technologies

Essential to this study is the data related to the movement observed during the use of different metal-detector configurations. By recording the actual product-arm trajectories, unknown influences will be accounted for in the simulations. Influences of movement may include fatigue, noise and environmental distractions. Real data is valuable as it would otherwise be difficult to hypothesize the change in movement with fatigue and other factors.

It is important to record the load applied to the hand and arm during the extent of movement. When a metal detector is coupled to a human arm through a handle and armrest-strap assembly, a hybrid kinematic chain is formed. The forward dynamic algorithms become a little more complicated (Featherstone & Orin 2008, 35-65) for closed chain sub-systems due to the system being overdetermined. There also exists uncertainty in the armrest and hand couplings due to wobble and soft-tissue effects, and relying on the hybrid kinematics and product inertial characteristics alone opens up the possibility for inaccuracies in the model. In contrast, any external loading from the surroundings that would otherwise be difficult to predict, are easily accounted for in the simulations if they are directly measured. It is for these reasons that the product is *de-coupled* by way of direct load-cell measurement in this study. The result is an open, serial link system with the measured forces applied orthogonally to the hand and arm and this dynamic arrangement is far easier to model for the purposes of this study (Vidyasagar, Hutchinson & Spong 2006).

There are many commercially available systems that implement optical motion capture technology. Motion capture systems of notable quality and accuracy are those from Vicon Motion Systems (Vicon n.d.) who develop state-of-the-art optical motion capture systems. Despite these systems being highly accurate, the difficulty with this technology is its high cost, the need for complex support and lack of portability. For this study, an alternative to optical motion capture had to be implemented due to cost constraints and field portability requirements. The two most viable alternatives in terms of cost and applicability are inertial measurement units (IMUs) and magnetometers. Since magnetometers are affected by the proximity of ferrous materials locally disturbing the

Earth's magnetic field (Mayagoitia, Nene & Veltink 2002), multi-axis IMUs were initially selected for the motion capture experiments. In this study, due to time-constraints the motion of interest has instead been synthesized from force-time histories as explained in chapter 3. However the implementation of IMU sensors in the future still remains viable.

### 1.3.1 Measuring Uncertainty: Optimal State Estimation

For future work, accelerometers may be attached to non-rigid areas on the arm and hand. As such the measurements may be affected by latent oscillations arising from the skin and tissue beneath. In addition to this, inherent characteristics of the sensors give rise to drift and noise in the measurements. These two noise sources are classified as process and measurement noise respectively and since a dynamic simulation may involve many integration routines, these noise sources may become a problem as the integration errors compound over time.

A digital signal processing (DSP) technique which will be used in the future to accurately estimate the joint motion from noisy accelerometer measurements is that of model-based inertial motion sensing using an extended Kalman filter (EKF) in combination with a Lagrangian dynamics model. This approach is motivated by similar past studies in state estimation of planar, multi-link pendulum systems in the context of human motion capture (Music, Kamnik & Munih 2008); (Zhou & Hu 2005). The details of the implementation are discussed in chapter 7.

## 1.4 Dissertation Overview

This dissertation is organized as follows:

**Chapter 2** Details the scope and specific objectives of the study. The specification submission at current issue is included in Appendix A.

**Chapter 3** Details the technical methodology and experimental design employed for

the study.

**Chapter 4** Details the hardware design including construction of a configurable metal-detector dummy, sensor network and data acquisition equipment.

**Chapter 5** Details the numerical methods and software tools implemented for this study.

**Chapter 6** Discusses the experimental results obtained thus far.

**Chapter 7** Provides conclusions obtained from the study and experimental results.

# Chapter 2

## Scope and Objectives

### 2.1 Chapter Overview

This chapter addresses the project deliverables and detailed system specifications for a suitable biomechanical model, hardware, numerical tools employed in software, data storage and experimental guidelines. The project specification at current issue may be found in Appendix A.

#### 2.1.1 Hypothesis

The underlying hypothesis in this study is two-fold and presented as follows:-

1. For a given (captured or measured) cycle of movement of a metal-detector, the *distribution* of muscle activity required for that movement forms a key assessment of the ergonomic performance of the product.
2. Collectively, the elements making up a product are fixed and constrained in mass due to the characteristics of the enabling technologies. However, there exists an ideal *arrangement* of elements about the location of the product centre of gravity that allows for an optimal inertial configuration for maximum comfort of use.



### 2.1.2 Specific Project Objectives

The hypothesis is addressed with the following key deliverables:-

1. The design and implementation of a biomechanical test and analysis platform.
2. The initiation of an ergonomic performance database that contains muscle activation data and is indexed by system mass and inertial configurations for that mass. The database is to be of a format that may easily be used as a design tool in future product development.

## 2.2 Detailed Specifications

### 2.2.1 Biomechanical Model

The biomechanical model consists of muscles, tendons, bones, a shoulder fixed joint, elbow and wrist joint. Simplified muscles are used with specific parameters and tendon insertion points taken from an existing upper extremity model (Holzbaur et al. 2005). The model of Holzbaur et al. (2005) consists of 50 musculo-tendon compartments and 12 degrees of freedom (DOF). For the purposes of this study the model to be used is simplified by reducing the number of musculo-tendon units to 11 *lumped* muscle groups and the number of degrees of freedom to 7 (3 in the shoulder, 2 in the elbow and 2 in the wrist). The simplified biomechanical model is shown in figure 2.1.

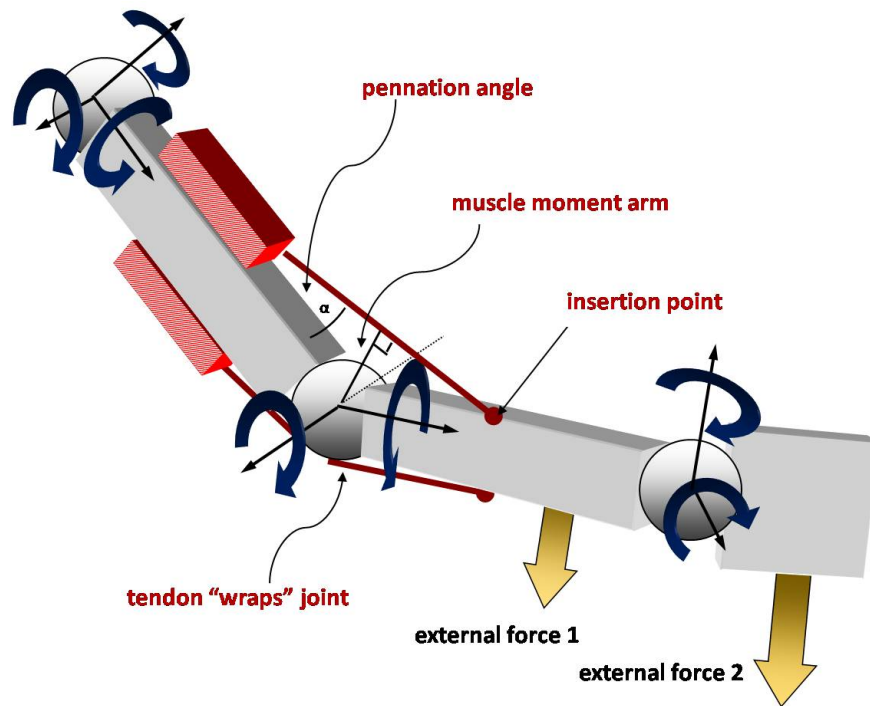


Figure 2.1: The biomechanical model simplified from Holzbaur et al. (2005) used in this study. Only two musculotendon units are shown for simplicity.

### 2.2.2 Hardware

The hardware requirements for the project include the following,

1. A dummy metal-detector with integrated force-sensor network. The dummy metal-detector is to have a modular design for quick turnaround of different inertial configurations.
2. Data acquisition equipment which is to be bus-powered for portability and with adequate sampling capability.

## 2.2.3 Numerical Tools and Software

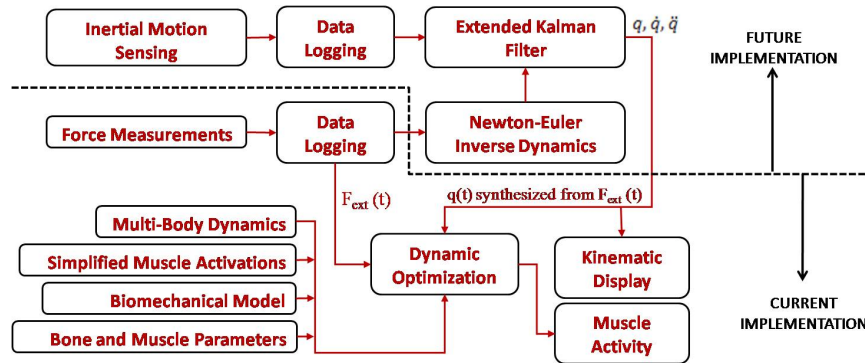


Figure 2.2: A system diagram of the numerical approach used in this study and that which may be implemented in the future.

The figure above shows the system diagram representing the numerical approach to be used in the study. A dynamic optimisation tool is to be used to resolve muscle actuations for measured (or synthesized) movement.

A graphical user interface front-end application with a 3D display of the links and muscles is required for visualization. This application also must allow for the initialization of experimental parameters. It must also provide graphical output plots of the trajectories and actuations as solved by the dynamic optimisation module.

# Chapter 3

## Theory and Methodology

### 3.1 Chapter Overview

This chapter focuses on the underlying theory employed in this study. It starts off with an explanation of a suitable kinematic model and details the use of homogenous transformations. A discussion of a *pseudo motion capture* technique using force-time data is then presented. Muscle-actuator dynamics is briefly discussed and finally the relevant rigid-body formalisms, using Lagrangian dynamics, is introduced.

### 3.2 Kinematics

#### 3.2.1 Joint Definitions

Each compound joint in the kinematic model for this study is made up of a series of two or more rotational joints, the constraints of which are *holonomic*. For a constraint to be holonomic it must be expressible as a function of the form (Vidyasagar et al. 2006, 197)

$$f(q_1, q_2, \dots, q_n) = 0 \tag{3.1}$$

In equation 3.1,  $q_1, q_2, \dots, q_n$  are generalized coordinates and the function  $f$  depends only on these coordinates and time. It does not depend on the *time-derivatives* (or velocities) of the joints.

Each compound joint forms an orthogonal set of two or three planar, rotary joints as depicted by the kinematic configuration with 7 degrees of freedom shown in figure 3.1.

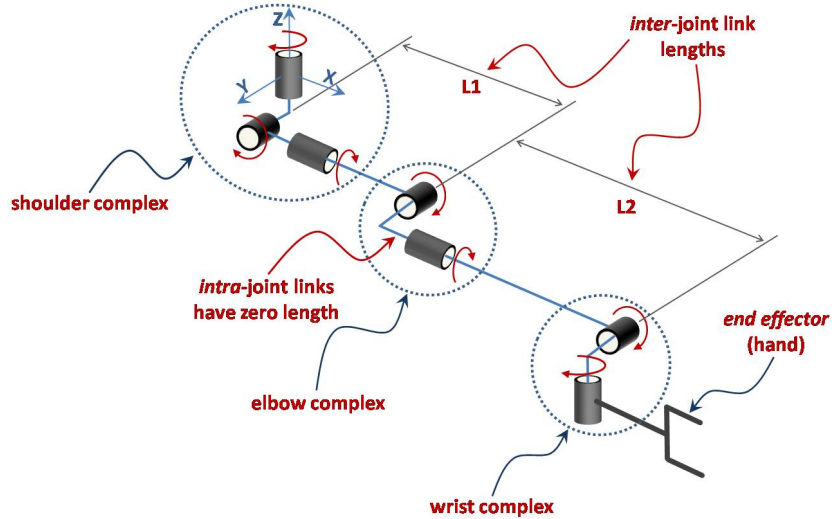


Figure 3.1: The 7DOF kinematic configuration of the interconnected shoulder, elbow and wrist complexes of the arm extremity. The links connecting joints within a complex have zero length. The links connecting complexes, in this case the upper arm and forearm, have lengths  $L1$  and  $L2$ .

### 3.2.2 Homogenous Transformations

Angular joint articulations in this model are described with the convention of roll, pitch and yaw. The shoulder complex involves a complete set of roll, pitch and yaw angles ( $\theta$ ,  $\phi$  and  $\psi$ ) whereas the elbow involves pitch and yaw and the wrist involves yaw and roll articulations. A homogenous transformation matrix contains both a rotational and translational part and is expressed in the form

$$\begin{bmatrix} \mathbf{R}_{\phi\theta\psi} & \mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix} \quad (3.2)$$

In matrix 3.2,  $\mathbf{R}_{\gamma\theta\psi}$  is a 3x3 rotational transformation matrix of the form (Vidyasagar et al. 2006, 57)

$$\begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix} \quad (3.3)$$

where,

$$\begin{aligned} s_\theta &= \sin \theta & c_\theta &= \cos \theta \\ s_\phi &= \sin \phi & c_\phi &= \cos \phi \\ s_\psi &= \sin \psi & c_\psi &= \cos \psi \end{aligned}$$

and  $\mathbf{d}$  is a vector  $(L_x \ L_y \ L_z)^\top$  representing the translation with respect to a local coordinate frame. We let the transformation of the shoulder complex be  $T_1$  as per 3.3 with  $\theta_1$ ,  $\theta_2$  and  $\theta_3$  for roll, pitch and yaw respectively such that

$$T_1 = \begin{bmatrix} c_{\theta_3} c_{\theta_2} & -s_{\theta_3} c_{\theta_1} + c_{\theta_3} s_{\theta_2} s_{\theta_1} & s_{\theta_3} s_{\theta_1} + c_{\theta_3} s_{\theta_2} c_{\theta_1} & 0 \\ s_{\theta_3} c_{\theta_2} & c_{\theta_3} c_{\theta_1} + s_{\theta_3} s_{\theta_2} s_{\theta_1} & -c_{\theta_3} s_{\theta_1} + s_{\theta_3} s_{\theta_2} c_{\theta_1} & 0 \\ -s_{\theta_2} & c_{\theta_2} s_{\theta_1} & c_{\theta_2} c_{\theta_1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

For the elbow joint, we have only roll and pitch. We let the transformation of the elbow complex be  $T_2$ . Substituting zero into the yaw angle we obtain

$$T_2 = \begin{bmatrix} c_{\theta_5} & s_{\theta_5} s_{\theta_4} & s_{\theta_5} c_{\theta_4} & L_1 \\ 0 & c_{\theta_4} & -s_{\theta_4} & 0 \\ -s_{\theta_5} & c_{\theta_5} s_{\theta_4} & c_{\theta_5} c_{\theta_4} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

Similarly, for the wrist joint, we have only pitch and yaw. We let the transformation

of the elbow complex be  $T_3$ . Substituting zero into the roll angle we obtain

$$T_3 = \begin{bmatrix} c_{\theta_7}c_{\theta_6} & -s_{\theta_7} & c_{\theta_7}s_{\theta_6} & L_2 \\ s_{\theta_7}c_{\theta_6} & c_{\theta_7} & s_{\theta_7}s_{\theta_6} & 0 \\ -s_{\theta_6} & 0 & c_{\theta_6} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.6)$$

To find the position of a point  $\mathbf{P}$  anywhere on the upper arm with respect to the shoulder (or global) fixed frame  $\mathbf{O}_s$  we apply the homogenous transformation

$$\begin{bmatrix} \mathbf{P}_O \\ 1 \end{bmatrix} = T_1 \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \quad (3.7)$$

To find the position of a point  $\mathbf{P}$  anywhere on the lower arm with respect to the shoulder (or global) fixed frame  $\mathbf{O}_s$  we apply a *series* of homogenous transformations

$$\begin{bmatrix} \mathbf{P}_O \\ 1 \end{bmatrix} = T_1 T_2 \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \quad (3.8)$$

Finally, to find the position of a point  $\mathbf{P}$  anywhere on the hand (or end effector) with respect to the shoulder (or global) fixed frame  $\mathbf{O}_s$  we apply the series of homogenous transformations

$$\begin{bmatrix} \mathbf{P}_O \\ 1 \end{bmatrix} = T_1 T_2 T_3 \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \quad (3.9)$$

In applying the product of homogenous transformations from 3.4 to 3.6 as shown above

in 3.7 to 3.9, it is possible to find the position, with respect to the global frame, of a point on any of the links in the model. This is important for the purposes of setting an initial pose and representing the position and orientation of any bodies within the model for visualisation within a single, 3D coordinate frame (such as a MATLAB graph).

In the context of this model, the term *proximal* refers to entities closer to the grounded shoulder joint or global origin. The term *distal* refers to entities farthest from the global origin. Using this convention, a joint separates a proximal from a distal coordinate frame. In section 3.4.5, it will be shown how an *inverse* transformation may be applied to transform coordinates in a proximal frame to coordinates in the distal frame. In so doing, with the knowledge of individual muscle insertion points (refer figure 2.1), it is possible to find the muscle line of action as well as the moment arm to the joint centre.

### 3.3 Pseudo Motion Capture

For the purposes of this study, it has been assumed that the movement of the metal detector coil is, for the most part, consistent. Furthermore, this movement needs to conform to an ideal horizontal swing speed as well as a constant, ideal elevation off the ground. With this in mind, it should be possible to reasonably synthesise the movement of the product and joint articulations. It has also been further assumed that the bulk movement or *swing* arises from the shoulder articulation and the elbow and wrist joints are kept relatively static.

In applying this motion synthesis and to make sense from the experimental force-time histories, it is necessary to synchronise the shoulder articulations with force data measured at the handle. As will be shown in this section, this may be done by adjusting the swing phase and period of the synthesised movement to match that of the measurements.



### 3.3.1 Experimental Methodology

Before discussing any post-processing of data, it is necessary to first describe the experimental methodology implemented in this study. The experimental process is shown in figure 3.2. A subject stands with the product attached to the arm in an initial pose. Markers are placed on an imaginary ground-arc and subtend an angle about the shoulder vertical axis. This angle defines the range of shoulder articulation which in this kinematic model may be defined as  $[\theta_{2_{min}}; \theta_{2_{max}}]$ .

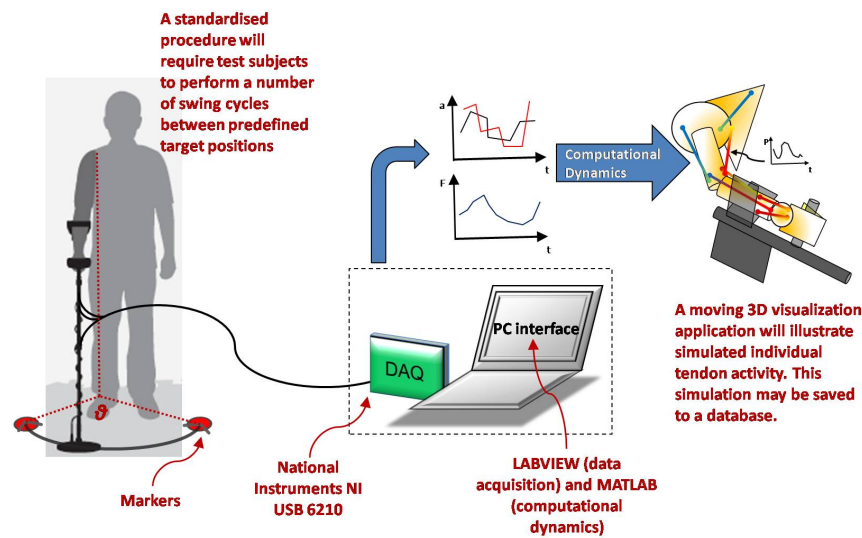


Figure 3.2: The experimental procedure involves sampling a number of swing cycles between predefined target positions. The force-time histories at the handle and armrest are recorded and post-processed using MATLAB and LABVIEW.

The metal-detector dummy is adjusted to a particular mass and inertial configuration in question. A subject then gently swings the dummy in cycles, tracing an imaginary arc between the two ground markers with the coil for approximately 15 seconds. During this time, force-time measurements at the handle and arm-rest are recorded with the use of a NI USB 6210 data acquisition unit and a LABVIEW software application. The data is saved as a CSV delimited file for further post-processing in MATLAB later on (refer to chapter 4 for details).

### 3.3.2 Force-Time Synchronisation

From the knowledge of  $\theta_{2_{min}}$  and  $\theta_{2_{max}}$  and the force-time history at the handle, it is possible to synthesize  $\theta_2$  shoulder articulation. The remaining shoulder joints ( $\theta_1$  and  $\theta_3$ ) are kept static with a value from the initial pose as are all the elbow and wrist joints. Regarding this force-time data, it may be observed that the signal obtained is largely periodic and sinusoidal (3.3). Only a windowed portion of the data is processed due to the lengthy convergence of the optimisation procedure that would otherwise result later on.

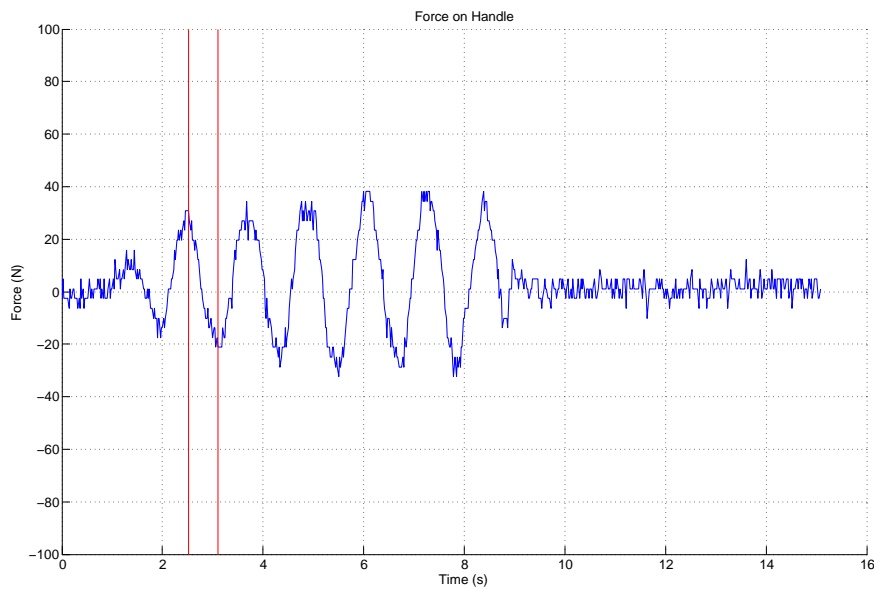


Figure 3.3: Typical data for the force-time histories measured at the handle. The red lines indicate the window of time over which the data will be further processed.

A trigonometric curve-fit is applied to the windowed data as shown in figure 3.4. The force-time data is approximated by the trigonometric function

$$F(t)_{hand} \approx A \sin(\alpha t + \beta) \quad (3.10)$$

where  $t$  is time,  $A$  is the amplitude of the  $F(t)_{hand}$  signal,  $\alpha$  is the period and  $\beta$  is the phase. A force is associated with acceleration and to obtain a linear approximation of the corresponding displacement ( $s_{lin}$ ) the data needs to be integrated twice and scaled

by an inverse mass factor ( $m^{-1}$ ) as follows

$$s_{lin} = \frac{1}{m} \int \left( \int F_{hand} dt \right) dt \quad (3.11)$$

Since the original  $F(t)$  data is noisy and susceptible to drift, integration using the manner shown above results in a radical divergence of the associated displacement-time function. Instead, using the trigonometric fit as the integrand results in the expected, theoretical displacement or *antiphase* of 3.10,

$$\theta_2 \approx D \sin(\alpha t + \beta + \pi) \quad (3.12)$$

where  $D$  is the amplitude of displacement defined as  $(\theta_{2_{max}} - \theta_{2_{min}})$  known *a priori* from the ground-marker positions in the experimental setup. The parameters of period ( $\alpha$ ) and phase ( $\beta$ ) are adjusted as needed to best fit the force measurements in a data-acquisition post-processor (refer to chapter 4 for details).

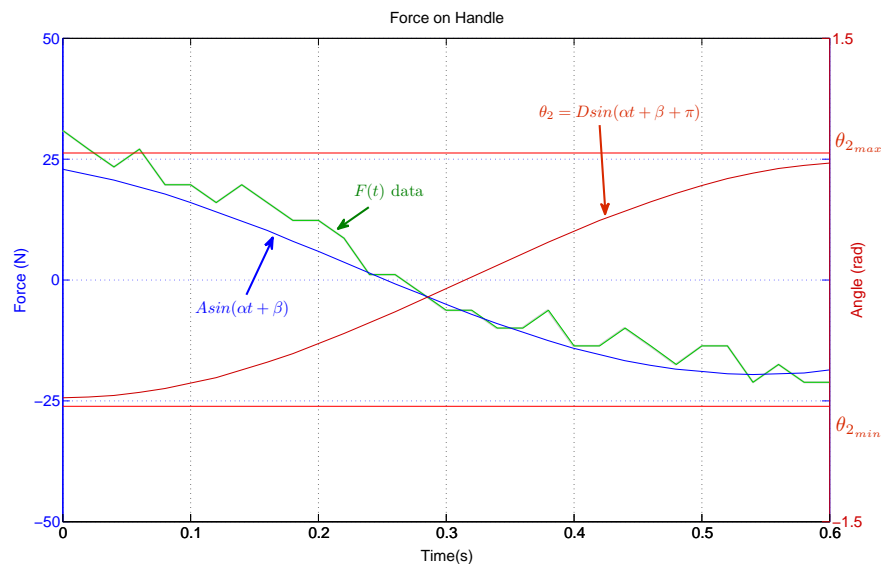


Figure 3.4: A trigonometric fit (blue) of typical force-time (green) measurement at the handle. The resulting shoulder articulation (red) is *antiphase* to the fit.

For most of the experiments, the windowed time-frame lasts for the duration of a single

half-cycle of coil movement, starting at one marker and ending at the other.

## 3.4 Muscle-Actuator Dynamics

### 3.4.1 Force-Velocity and Tension-Length Relationships

Muscle actuator relationships are comprehensively summarised in (Stelzer & von Stryk 2006). The two most significant relationships are force-velocity and tension-length derived from the Hill type muscle model. The force-velocity relationship is defined as

$$f_{FV}(v) = \begin{cases} \frac{1 - \frac{v}{v_{max}}}{1 + \frac{v_{max}c_3 v}{v}} & \text{if } v \leq 0 \\ \frac{1 - 1.33 \frac{v_{max}c_4}{v}}{1 - \frac{v}{v_{max}c_4}} & \text{if } v > 0 \end{cases} \quad (3.13)$$

The tension-length relationship is defined as

$$f_{TL}(l) = \begin{cases} e^{-\frac{1}{c_1} \left(1 - \frac{l}{1.1l_0}\right)^3} & \text{if } l \leq 1.1l_0 \\ e^{-\frac{1}{c_2} \left(\frac{l}{1.1l_0} - 1\right)^3} & \text{if } l > 1.1l_0 \end{cases} \quad (3.14)$$

In the above equations,  $v_{max}$  is the maximum contraction velocity,  $l_0$  is the optimal musculo-tendon length and  $c_1$ ,  $c_2$ ,  $c_3$  and  $c_4$  are parameters specific to the individual muscle. These can be found by referring to a comprehensive musculoskeletal model such as that compiled by Holzbaaur et al. (2005). As discussed in section xx, with knowledge of the individual insertion points, the muscle length can be found for any given joint state ( $q$ ) using an inverse homogenous transformation and so a collective function for the actuator lengths is

$$f(l) = l(\mathbf{q}) \quad (3.15)$$

The muscle velocity is simply the time-derivative of muscle length function and so these are collectively described as

$$f(v) = \frac{\partial f(l)}{\partial t} \quad (3.16)$$

### 3.4.2 Activation Dynamics

The force exerted by muscles is not instantaneous. When a muscle is excited the concentration of calcium ions increases until the muscle finally exerts a force (Stelzer & von Stryk 2006). If the muscle excitation is  $u$ , the calcium ion concentration  $\gamma$  and with the selection of suitable parameters of  $b_1$ ,  $b_2$  and  $b_3$ , this property may be described by the first-order ODE as

$$\dot{\gamma} = b_2(b_3u - \gamma) \quad (3.17)$$

The relationship of the calcium ion concentration to the force exerted by the muscle is given by the following equation

$$f_{AD}(\gamma(u)) = \frac{(b_1\gamma(u))^3}{1 + (b_1\gamma(u))^3} \quad (3.18)$$

### 3.4.3 Total Muscle Force

By combining equations 3.13 to 3.18 the total muscle force at any state is represented by the following expression

$$F(\gamma, f(l), f(v)) = f_{iso-max} f_{AD}(\gamma) f_{TL}(l) f_{FV}(v) + f_{PE}(l) + f_{DE}(v) \quad (3.19)$$

where  $f_{iso-max}$  is the muscle peak isometric force,  $f_{PE}$  is the passive elastic muscle force and  $f_{DE}$  is the passive damping force. The passive elements,  $f_{PE}$  and  $f_{DE}$  are described by

$$f_{PE}(l) = k_1(e^{k_2(l - k_3)} - 1) + k_4(e^{k_5(l - k_6)} - 1) \quad \text{and} \quad f_{DE}(v) = k_0v$$

where the parameters  $k_1$  to  $k_6$  and  $f_{iso-max}$  are specific to the individual muscle.

#### 3.4.4 PWM, Muscle Grouping, Parameters and Attachments

The complete set of equations from 3.13 to 3.19 were implemented initially in a solution procedure for the optimal control problem to be discussed in chapter 4. However due to their complexity and the non-linearity of the kinematic transformations, convergence times are unyieldly. It is for this reason that the muscle-actuator dynamics have been greatly simplified for the purposes of this study.

The simplification that has been undertaken to reduce the complexity of the muscle-actuator system has been to substitute the state-dependant product of force-velocity, tension-length and chemical activation with a sinusoidal pulse-width modulated (PWM) actuation scheme. With regards to equation 3.19, for a single muscle-actuator the total muscle force may then be approximated as

$$F(\gamma, f(l), f(v)) \approx u f_{iso-max} \sin \kappa t, \quad 0 \leq u \leq 1 \quad (3.20)$$

where  $u$  is a control variable (found in the optimal control solution) and  $\kappa$  is the period of actuations.

A further simplification, as mentioned in chapter 2, has been to reduce the number of musculo-tendon groups from fifty to eleven as shown in figure 3.5. This has been done by grouping the muscles and selecting a suitable lumped parameter of peak isometric force ( $f_{iso-max}$ ) for each muscle group. The muscle attachment points are specified with respect to the coordinate frame of the link on which they are attached. For the lumped muscle groups, the attachment points of the major muscle in the group has been selected. The attachment data from Holzbaur et al. (2005) has been used in this model for muscle placement and a table of the grouping can be found in Appendix B.

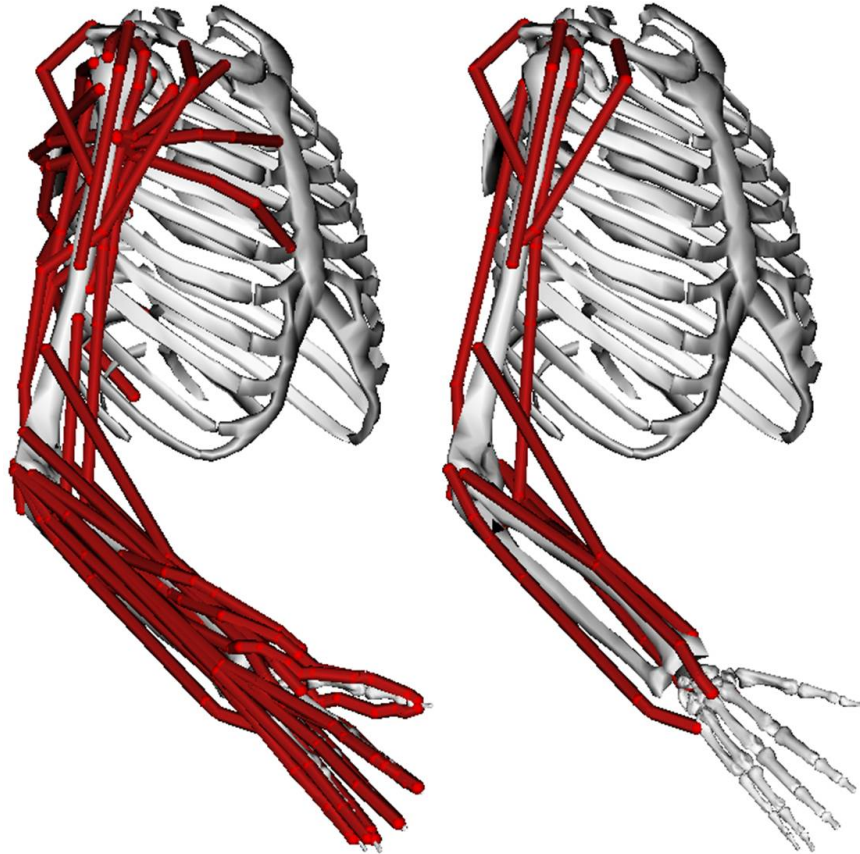


Figure 3.5: A musculoskeletal model of 50 musculo-tendon compartments taken from Holzbaur et al. (2005) (left) reduced to 11 (right).

### 3.4.5 State-Dependant Torque

As mentioned in section 3.2.2, using an inverse homogenous transformation, it is possible to express the proximal coordinates of a muscular attachment point in a distal coordinate frame. In this way, the muscle line of action is succinctly calculated and, furthermore, the moment-arm about the joint may be found using an orthogonal projection from the joint centre to the line of action as illustrated in 3.6. The joint centre is the origin of the distal coordinate frame. To transform the coordinates of a point from a proximal frame to a distal frame the following inverse homogenous transformation

applies

$$\begin{bmatrix} \mathbf{P}_{distal} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_n^T & -\mathbf{R}_n^T \mathbf{T}_n \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{proximal} \\ 1 \end{bmatrix} \quad (3.21)$$

In equation 3.21,  $R_n^T$  and  $T_n$  are the 3x3 inverse-rotational part and 1x3 translational part respectively from any of the transformation matrices 3.4, 3.5 and 3.6. The choice of the transformation matrix ( $T_n$ ) will depend on the joint in question (shoulder, elbow or wrist). The rotational part of these matrices is easily invertible using the transpose since it is *symmetric positive definite*.

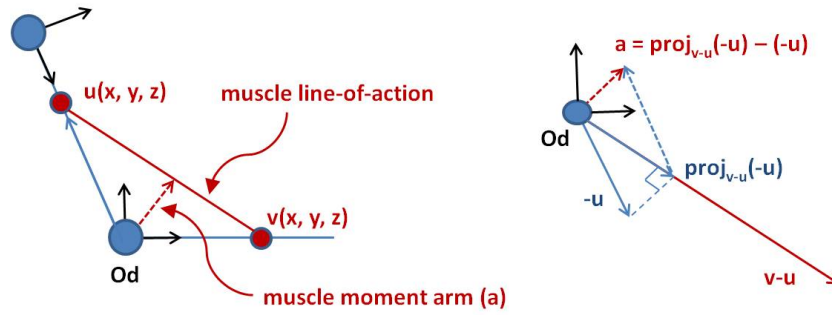


Figure 3.6: Two muscle attachment points, one in the proximal and one in the distal frame, define the muscle line-of-action (left). The orthogonal distance from the joint centre to the muscle line-of-action is defined as the moment-arm. The vector diagram (right) represents the line-of-action and moment-arm about the joint.

If we define the transformed proximal muscle attachment point as a vector  $\mathbf{u}$  and the distal attachment point as  $\mathbf{v}$ , then the muscle line-of-action vector is simply  $(\mathbf{v} - \mathbf{u})$ . The orthogonal projection of  $\mathbf{u}$  on the muscle line-of-action is defined as

$$proj_{\mathbf{v}-\mathbf{u}}(-\mathbf{u}) = \frac{-\mathbf{u} \cdot (\mathbf{v} - \mathbf{u})}{(\mathbf{v} - \mathbf{u}) \cdot (\mathbf{v} - \mathbf{u})} (\mathbf{v} - \mathbf{u}) \quad (3.22)$$

and the moment-arm vector is thus

$$\mathbf{a} = proj_{\mathbf{v}-\mathbf{u}}(-\mathbf{u}) + \mathbf{u} \quad (3.23)$$



The torque applied by the muscle at the joint is the cross-product of the muscle force vector and moment arm,

$$\tau = \left( F(\gamma, f(l), f(v)) \frac{\mathbf{v} - \mathbf{u}}{\|\mathbf{v} - \mathbf{u}\|} \right) \times \mathbf{a} \quad (3.24)$$

where  $\frac{\mathbf{v} - \mathbf{u}}{\|\mathbf{v} - \mathbf{u}\|}$  is the muscle line-of-action unit vector. The vectors  $\mathbf{u}$  and  $\mathbf{v}$  depend on the angle of the joint and hence the moment arm will change as the joint angle changes. It is for this reason that torque in equation 3.24 is state dependant. For the numerical implementation, the moment-arm and muscle line-of-action unit vector are calculated as functions *off-line* with the use of equations 3.21, 3.22, 3.23 and 3.24 and a symbolic math tool. The moment-arm function  $\mathbf{a}$  and muscle-line-of action unit vector  $\mathbf{V}$  function for a given muscle are described with

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} f_{a_x}(\theta_1, \theta_2, \dots, \theta_7) \\ f_{a_y}(\theta_1, \theta_2, \dots, \theta_7) \\ f_{a_z}(\theta_1, \theta_2, \dots, \theta_7) \end{bmatrix} \quad \text{and} \quad \mathbf{V} = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} f_{V_x}(\theta_1, \theta_2, \dots, \theta_7) \\ f_{V_y}(\theta_1, \theta_2, \dots, \theta_7) \\ f_{V_z}(\theta_1, \theta_2, \dots, \theta_7) \end{bmatrix} \quad (3.25)$$

By observing the above equations, it is clear that the vector functions  $f_{\mathbf{V}}$  and  $f_{\mathbf{a}}$  are dependant on the joint angles. Due to the kinematic transformations, each component of these vectors may be a lengthy trigonometric function and contributes significantly to increased convergence time of the optimal control solution procedure. It may be the subject of future work to further optimise these components for speedier processing. The off-line MATLAB code to be produce the symbolic entries  $\mathbf{a}$  and  $\mathbf{V}$  is listed in Appendix C.

### 3.5 Rigid Body Dynamics

The focus of this section will be on the equations of motion that govern the system at hand. The methods employed are very similar to those used in robotics and reference is made to Vidyasagar et al. (2006) for the derivations of the Lagrangian equations.

From the Lagrangian equation a state-space model may be derived for later use in an optimal control problem.

### 3.5.1 Lagrangian Formulation

The Euler-Lagrange equation of motion is expressed as

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau_i \quad i = 1 \dots n \quad \text{and} \quad L = K - P \quad (3.26)$$

for  $n$  general coordinates where  $K$  and  $P$  are the system kinetic and potential energy respectively. From equation 3.26 the well-known equation for rigid body dynamics may be derived as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} = \boldsymbol{\tau} - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{G}(\mathbf{q}) \quad (3.27)$$

where

- $\mathbf{M}$  is the mass matrix
- $\mathbf{q}$  are the joint angles
- $\boldsymbol{\tau}$  are the total torques
- $\mathbf{C}$  are the Coriolis and centrifugal forces
- $\mathbf{G}$  is the gravitational force

The derivation of the mass matrix for a 7DOF system using inertia terms and jacobians as well as *Christoffel symbols*<sup>1</sup> for the Coriolis and centrifugal terms is a lengthy and complex task and will not be undertaken here. Instead a *symbolic generator* is used to find the mass matrix for this system as described in chapter 4.

<sup>1</sup>For a full derivation of the mass matrix for some simpler configurations refer to Vidyasagar et al. (2006, 210-216).

## 3.5.2 State-Space Model

To facilitate the formulation of the optimal control problem the equation of 3.27 is re-arranged and cast into a state-space model as follows

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \\ \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix} = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{2n} \\ M^{-1}(x_1, \dots, x_n) \left( \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} - [\mathbf{C}] - [\mathbf{G}] \right) \end{bmatrix} \quad (3.28)$$

where

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix}$$

are the joint velocities and joint accelerations respectively. The inverse mass matrix,  $\mathbf{M}^{-1}$  is a function of joint angles. As mentioned, the procedure to find  $\mathbf{M}$  and its inverse is discussed in chapter 4. The matrix,  $[\tau_1 \ \dots \ \tau_n]^T$ , represents the total torques applied at the joints. The total torque at a joint includes the vector sum of torques arising from muscle activity (since more than one muscle acts on a joint) as well as the torques arising from the forces measured at the handle and armrest.

## Chapter 4

# Numerical Methods and Software Implementation

### 4.1 Chapter Overview

The focus of this chapter will be on the numerical methods and software tools implemented in the study. Off-line processing with the symbolic toolbox from MATLAB has been extensively used to find the vector functions of the moment-arms and muscle line-of-action. These functions are called per-time-step in the optimal control solution procedure (OCSP) where the joint angles are provided as inputs to the functions.

A powerful symbolic processor from *Robotran* has been used to find the mass-matrix function which also accepts as input the joint angles and is called per-time-step in the OCSP.

Finally the implementation of the Gauss Pseudo-Spectral Optimisation software (GPOPS) for use in the OCSP, boundary constraints and nodal resolution is discussed.

## 4.2 MBSysTran Implementation

From the bottom half of the state-space equation 3.28 we have,

$$\begin{bmatrix} \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix} = M^{-1}(x_1, \dots, x_n) \left( \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} - [\mathbf{C}] - [\mathbf{G}] \right) \quad (4.1)$$

where, upon rearrangement,

$$M(x_1, \dots, x_n) \begin{bmatrix} \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} - [\mathbf{C}] - [\mathbf{G}] \quad (4.2)$$

By making the following substitution into 4.2,

$$\begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} - [\mathbf{C}] - [\mathbf{G}] = -\bar{\mathbf{c}}(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g})$$

we have,

$$M(x_1, \dots, x_n) \begin{bmatrix} \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix} = -\bar{\mathbf{c}}(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g}) \quad (4.3)$$

where  $\bar{\mathbf{c}}$  is a collective vector function of total torques ( $\tau$ ), joint displacements and velocities ( $\mathbf{q}$  and  $\dot{\mathbf{q}}$ ), body parameters of mass, centre-of-mass and inertia ( $\delta$ ) and gravity ( $\mathbf{g}$ ).

With respect to equations 3.28 and 4.3, the functions of  $\mathbf{M}$  and  $\bar{\mathbf{c}}$  are required for the forward solution of the system. More specifically, the *inverse* of  $\mathbf{M}$  and  $\bar{\mathbf{c}}$  need

to be known for use in the OCSF. The classical assembly of these two functions from the Lagrangian formulation involves the use of joint and velocity jacobians along with *Christoffel symbols* and for a 7DOF system this requires unyieldly calculations with the possibility for errors to be made if done manually. Initially, an attempt was made to use the symbolic toolbox from MATLAB to systematically compile  $\mathbf{M}$  but this resulted in matrix entries of 100 or more trigonometric terms thereby rendering it impractical for inversion per time-step in the OCSF.

After further investigation, an on-line symbolic preprocessor was discovered (CEREM 2008), in the form of MBSysTran, for the purposes of compiling optimal symbolic functions  $\mathbf{M}$  and  $\bar{\mathbf{c}}$  and useful in the analysis of rigid-body dynamics and control systems. MBSysTran runs on a remote server under the ownership of MECA, UCL in Belgium. Compilation of optimal symbolic functions for use in robotics has been the topic of extensive research by Professors Paul Fisette and Jean-Claude Samin (Samin & Fisette 2003).

A graphical editor is used to construct a rigid-body model. Figure 4.1 shows the construction of the shoulder and upper-arm with three rotary joints in *MBSysPad*.

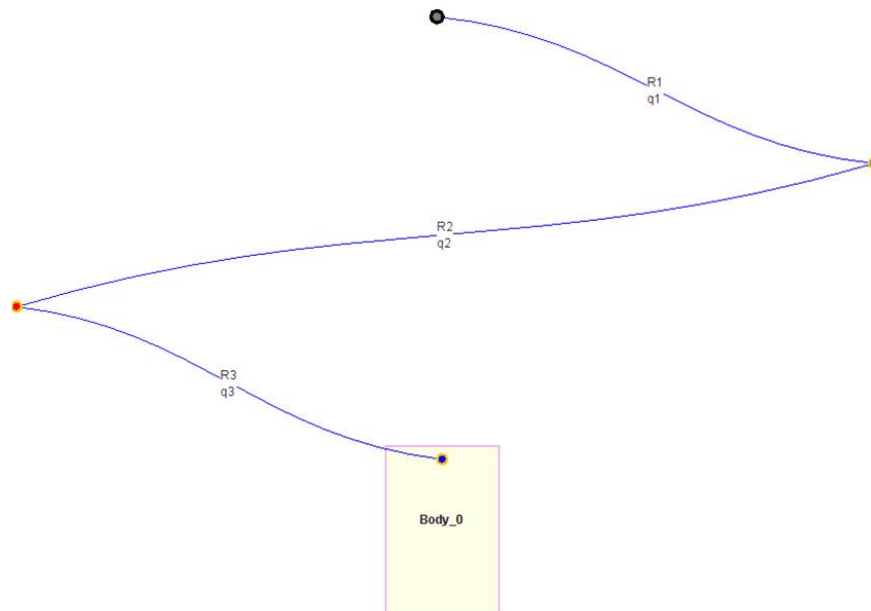


Figure 4.1: MBSysPad graphical editor is used for constructing rigid-body models (CEREM 2008). The above model is that of the shoulder complex connected to the upper-arm with three mutually orthogonal rotary joints.

From within the MBSysPad application environment a client request is sent to a server machine at UCL. If successful, the server uploads the compiled symbolic MATLAB file to the client machine. The file includes a function that returns the  $\mathbf{M}$  and  $\bar{\mathbf{c}}$  matrices. This has been done for the shoulder, elbow and wrist and the code has then been manually customised for implementation in the OCSP. Refer to Appendix D for the MATLAB code of the shoulder, elbow and wrist symbolic functions.

At each time-step of the OCSP, total torques and joint states are passed to the symbolic functions as inputs. The customisation of these functions has been an extension of the output to

$$\mathbf{O}_{sym}(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g}) = \mathbf{M}^{-1} \left[ -\bar{\mathbf{c}}^T(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g}) \right] \quad (4.4)$$

where  $\mathbf{M}^{-1}$  is of dimension  $nxn$ ,  $\bar{\mathbf{c}}^T$  is  $nx1$  and the inversion of  $\mathbf{M}$  is performed efficiently using Cholesky decomposition. Substitution of this output (4.4) into the state-space system of (3.28) yields

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \\ \dot{x}_{n+1} \\ \vdots \\ \dot{x}_{2n} \end{bmatrix} = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{2n} \\ \mathbf{O}_{sym}(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g}) \end{bmatrix} \quad (4.5)$$

The above forms the shell of a direct algebraic equation (DAE) as required by the OCSP.

### 4.3 Trajectory Optimisation

The term *trajectory optimisation*, used in the context of this study, is to find the required muscle activations that result in movement as close as possible to the *measured* (or

*synthesised*) movement. The muscle actuators need to be fired in an optimal manner such that the resulting movement of the product resembles that which has been measured (or synthesised) or, more technically, the forward simulation of the system has to represent the synthesised joint displacements.

### 4.3.1 Objective Function

Since more than one muscle-actuator acts on a joint, the system exhibits a high level of redundancy. Optimisation procedures allow for the resolution of such systems given a cost or objective function. There exists extensive literature, the likes of (Kaplan & Heegard 2002), (Stelzer & von Stryk 2006), (Menegaldo et al. 2003) and (Delp, Anderson, Arnold, Loan, Habib, Chand, Guendelman & Thelen 2007), covering optimisation approaches and the choice of objective function.

For goal-orientated motion, (eg. achieving maximum jump height, baseball pitching, metabolic cost minimisation etc), a dynamic optimisation procedure is generally best suited although computationally intensive. In this study, a dynamic optimisation procedure is undertaken on a greatly simplified system to reproduce measured movement. Using this approach also allows for the future implementation of movement prescription. For example, perhaps there exists an optimal swing that may be *prescribed* for the purpose of minimising muscle activity.

As represented in the state-space model of (3.28), the state variables,  $\mathbf{x}(t)$ , describe the system kinematics. The control variables,  $\mathbf{u}(t)$ , are used in the PWM scheme of muscle actuation as per equation 3.20 and are related to the state variables by differential equations. The main purpose of the optimal control problem is to find the functions,  $\mathbf{u}(t)$ , which minimise a cost function,  $J$ , over the time interval ( $t_0 \leq t \leq t_f$ ) according to

$$J = \phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) dt \quad (4.6)$$



where, in the context of the state-space model,

$$\begin{aligned}\dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{x}(t_0) &= \mathbf{x}_0\end{aligned}$$

The initial state,  $\mathbf{x}(t_0)$ , the start time,  $t_0$ , and the end time,  $t_f$  are specified. The solution to the optimal control problem requires finding  $\{\mathbf{x}^{op}(t), \mathbf{u}^{op}(t)\}$  such that the cost function,  $J$ , is minimised. The terms of  $\phi$  and  $\int L dt$  are known as the *Mayer* and *Lagrange* terms respectively. For goal-orientated motion the Mayer term is important, but for minimising the error between forward solution states and measured states (as is the case in this study) the Mayer term reduces to zero and the cost function,  $J$ , becomes

$$\begin{aligned}J &= \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) dt \\ &= \int_{t_0}^{t_f} (\mathbf{x} - \mathbf{x}_{op})^2 dt\end{aligned}\tag{4.7}$$

For the minimisation of  $J$ , the following conditions must be satisfied

$$\begin{aligned}\frac{\partial H}{\partial \mathbf{u}} &= \frac{\partial L}{\partial \mathbf{u}} + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T \lambda = \mathbf{0} \\ \dot{\lambda} &= -\frac{\partial H}{\partial \mathbf{x}} = -\frac{\partial L}{\partial \mathbf{x}} - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T \lambda\end{aligned}\tag{4.8}$$

$$\dot{\mathbf{x}}(t) - \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) = \mathbf{0}$$

where  $\lambda(t)$  are the *adjoint variables*, and the *Hamiltonian*,  $H$ , is a scalar function defined as

$$H(t) = L(\mathbf{x}(t), \mathbf{u}(t), t) + \lambda(t) \cdot \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)\tag{4.9}$$

The fundamental theorem of optimal control is the *Pontryagin minimum principle*,

which states that for the cost function,  $J$ , to be minimised, the Hamiltonian must be a minimum at the stationary solutions  $\{\mathbf{x}^{op}, \mathbf{u}^{op}\}$ .

### 4.3.2 Discretisation

The state variables are discretised over a time interval into  $n_j$  nodal values. An extended state vector,  $\mathbf{y}$ , is introduced and it is composed of generalised coordinates,  $\mathbf{q}$ , quasi-velocities,  $\mathbf{s}$ , and time derivatives of the quasi-velocities,  $\dot{\mathbf{s}}$ . This is in contrast to the state vector,  $\mathbf{x}$ , which contains only the generalised coordinates and quasi-velocities. At node  $a$ , the extended state vector,  $\mathbf{y}$ , is evaluated at time,  $t_a$ ,

$$\mathbf{y}(t_a) = \{\mathbf{q}(t_a), \mathbf{s}(t_a), \dot{\mathbf{s}}(t_a)\} \quad (a = 1, \dots, n_j) \quad (4.10)$$

$$\mathbf{Y} = \{\mathbf{y}_a(t)\}_{a=1}^{n_j}$$

where the vector  $Y$  is a collection of discrete values of the state variables at every node. The number of nodes has an effect on the solution convergence time. In this study, for  $0 \leq t \leq 0.6s$ , a suitable nodal count is  $n_j = 20$  which allows for a reasonable compromise between practical time constraints and solution accuracy.

### 4.3.3 Constraints and Boundary Conditions

Before implementation of the OCSP, limits have to be assigned to the state variable,  $\mathbf{x}$ , and control variable,  $\mathbf{u}$ . Limits for the control variable are

$$0 \leq u_i \leq 1 \quad i = (1, \dots, c_{ac}) \quad (4.11)$$

In the above constraint,  $c_{ac}$  is the total actuator count, which for this study is 11.

Limits for state variables specific to the shoulder complex are

$$\begin{aligned}
(x_1^{i_{pose}} - \delta_{x_1}) &\leq x_1 \leq (x_1^{i_{pose}} + \delta_{x_1}), & -\pi &\leq x_1^{i_{pose}} \leq \pi \\
x_2^{\theta_{2min}} &\leq x_2 \leq x_2^{\theta_{2max}}, & -\pi &\leq x_2^{\theta_2} \leq \pi \\
(x_3^{i_{pose}} - \delta_{x_3}) &\leq x_3 \leq (x_3^{i_{pose}} + \delta_{x_3}), & -\pi &\leq x_3^{i_{pose}} \leq \pi \\
-\delta_{\dot{x}_1} &\leq \dot{x}_1 \leq \delta_{\dot{x}_1} \\
-\pi \text{ rad.s}^{-1} &\leq \dot{x}_2 \leq \pi \text{ rad.s}^{-1} \\
-\delta_{\dot{x}_3} &\leq \dot{x}_3 \leq \delta_{\dot{x}_3}
\end{aligned} \tag{4.12}$$

Limits for the state variables specific to the elbow complex are

$$\begin{aligned}
(x_4^{i_{pose}} - \delta_{x_4}) &\leq x_4 \leq (x_4^{i_{pose}} + \delta_{x_4}), & -\pi &\leq x_4^{i_{pose}} \leq \pi \\
(x_5^{i_{pose}} - \delta_{x_5}) &\leq x_5 \leq (x_5^{i_{pose}} + \delta_{x_5}), & -\pi &\leq x_5^{i_{pose}} \leq \pi \\
-\delta_{\dot{x}_4} &\leq \dot{x}_4 \leq \delta_{\dot{x}_4} \\
-\delta_{\dot{x}_5} &\leq \dot{x}_5 \leq \delta_{\dot{x}_5}
\end{aligned} \tag{4.13}$$

and, finally, for the wrist complex the constraints are

$$\begin{aligned}
(x_6^{i_{pose}} - \delta_{x_6}) &\leq x_6 \leq (x_6^{i_{pose}} + \delta_{x_6}), & -\pi &\leq x_6^{i_{pose}} \leq \pi \\
(x_7^{i_{pose}} - \delta_{x_7}) &\leq x_7 \leq (x_7^{i_{pose}} + \delta_{x_7}), & -\pi &\leq x_7^{i_{pose}} \leq \pi \\
-\delta_{\dot{x}_6} &\leq \dot{x}_6 \leq \delta_{\dot{x}_6} \\
-\delta_{\dot{x}_7} &\leq \dot{x}_7 \leq \delta_{\dot{x}_7}
\end{aligned} \tag{4.14}$$

In the above constraint settings of 4.12, 4.13 and 4.14,  $\delta$  and  $\dot{\delta}$  relaxes the constraints and

$$0 \text{ rad} \leq \delta \leq 0.1 \text{ rad} \quad \text{and} \quad 0 \text{ rad.s}^{-1} \leq \dot{\delta} \leq 0.5 \text{ rad.s}^{-1} \tag{4.15}$$

allow for some numerical dampening in the solution procedure. Also,  $\mathbf{x}^{i_{pose}}$  is the collection of states from an initial pose of the user and  $[x_2^{\theta_{2min}}, x_2^{\theta_{2max}}]$  is the range of shoulder articulation in  $x_2$  that produces movement of the coil between marker positions.

#### 4.3.4 Gauss Pseudo-Spectral Optimisation Software (GPOPS)

A robust and well documented MATLAB toolbox for the solution of optimal control problems is the open-source Gauss Pseudo-Spectral Optimisation Software that has been implemented for this study. The algorithms have been developed for single or multiple-phase optimal control problems (Rao, Benson, Darby, Patterson, Francolin & Huntington 2010).

The GPOPS algorithms implement the Gauss and Radau hp-adaptive pseudospectral methods. In these methods the state is approximated using Lagrange polynomials and the time derivative of the state is collocated at either the Legendre-Gauss or Legendre-Gauss-Radau points. The continuous-time optimal control problem is then transcribed to a finite-dimensional nonlinear programming problem (NLP) and the NLP is solved using well known software tools for nonlinear optimization (SNOPT).

GPOPS uses a convenient multi-level structured interface to specify information in each phase of the problem although only a single phase is required for the problem undertaken in this study. With regard to function, all user-defined functions (e.g., differential-algebraic equations, boundary conditions, cost functional, and phase-connect conditions) have structured arrays as inputs, thus enabling the use of compact and efficient MATLAB code. The objective of GPOPS is to encourage an open collaboration between those in academia, industry, and government by providing an optimal control software in an open-source format that can be freely modified and adapted to the needs of a particular user.

MATLAB code snippets of the GPOPS setup routines, differential algebraic equations (DAE) functions and cost-functional routines for the shoulder, elbow and wrist complex may be found in Appendix E. The symbolic function  $\mathbf{O}_{sym}(\tau, \mathbf{q}, \dot{\mathbf{q}}, \delta, \mathbf{g})$  from 4.4 is called per time-step within the DAE function. This is the most significant deviation from the conventional structure of the DAE structure and the successful integration of the GPOPS and MBSysTran technologies has been the most notable achievement of the project. A flow diagram of the MATLAB implementation is shown in figure 4.2.

One drawback of this method is, where GPOPS allows for analytic jacobians and hes-

sians to be used for the cost-functional gradient, these are not used in the implementation for this study. Instead differentiation is performed by numerical finite-differencing within SNOPT, a third-party NLP library. Furthermore, the compact and efficient structures of GPOPS make use of MATLAB abstract objects for which the MBSys-Tran customised symbolic function is, as yet, unsuitable. Using the SNOPT routines allows for a conventional functional call of  $\mathbf{O}_{sym}$  per time-step but results in much longer solution convergence times. It may be the focus of future work to further improve the integration of these software technologies to make better use of the full power and speed of GPOPS.

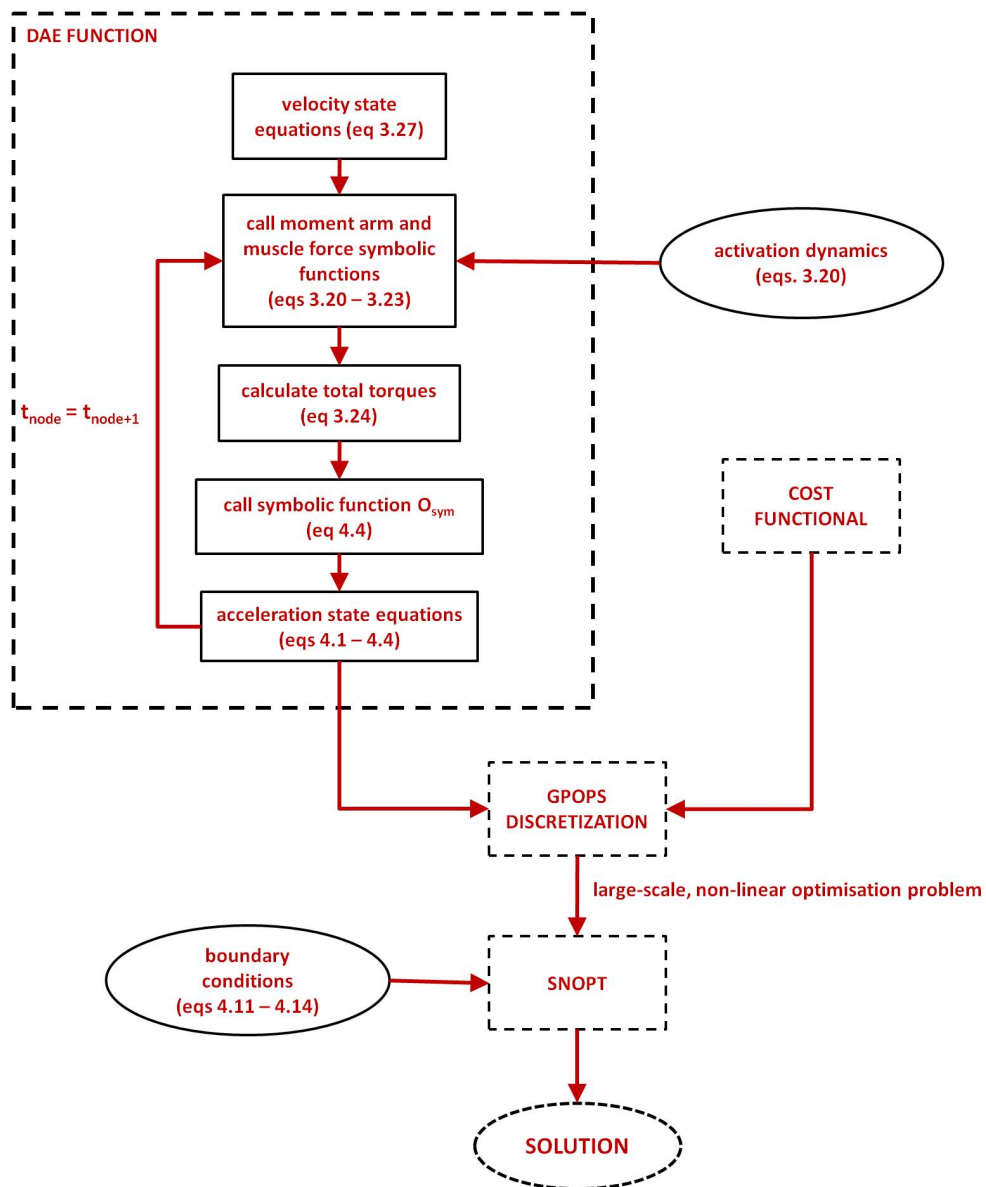


Figure 4.2: Flowchart of the MATLAB implementation of GPOPS

### 4.3.5 Assessment Criteria

In order to quantify the level of individual muscle activity required, the assessment undertaken in this study is based on an integral function of the control variable  $u$ ,

$$c = \int_{t_0}^{t_f} u dt \quad (4.16)$$

or, simply put, quantitative activity is the area under the  $u(t)$  curve for an individual muscle. An assessment on the *distribution* of all muscle activity will arise out of a collective regard for the integral of all  $u(t)$  curves.

## 4.4 Data Acquisition

The analog signals from the force gauges at the handle and arm-rest are sampled via a NI-USB 6210 data acquisition unit from Analog Devices. A LABVIEW executable application has been written that provides a user interface as shown in figure 4.3 for the purposes of recording and saving data in the form of a CSV delimited file for further processing in MATLAB. A calibration certificate has been provided by the supplier of the force gauges and, in sampling the data, the LABVIEW application also scales the data accordingly.

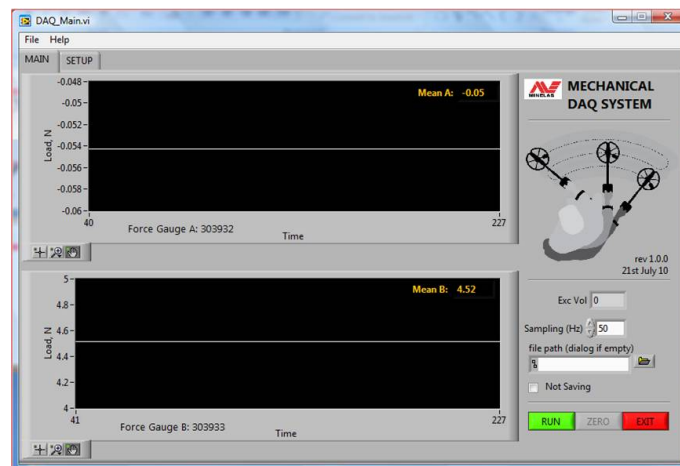


Figure 4.3: A screenshot of the LABVIEW application for recording and saving of the force-time histories at the handle and armrest.

# Chapter 5

## Hardware Design and Selection

### 5.1 Chapter Overview

The focus of this chapter will be on the design and selection of hardware required for the experimental processes undertaken in this study. Core elements of the hardware include a dummy metal-detector unit with configurable inertia, an integrated force sensor network and data acquisition equipment for sampling of the analog signals from the force gauges. The budgeted cost for hardware at the start of the project was \$5000 but the actual expenditure came to about half of this amount.

### 5.2 Force-Gauge Network

#### 5.2.1 Resolving Orthogonal Components

Essential to the processes of pseudo motion capture (chapter 3) and forward simulation (chapter 4) has been the integration of a force sensor network to measure the applied loads on the hand and forearm. More specifically, the measured force-time histories are applied to these points on the decoupled, open kinematic system to find the counter-torques included the equations of motion.

In this study, single forces are measured orthogonal to the hand and forearm as shown in figure 5.1. The unknown orthogonal components are resolved by taking moments about the center-of-gravity of the detector as illustrated in figures 5.2 and 5.3.

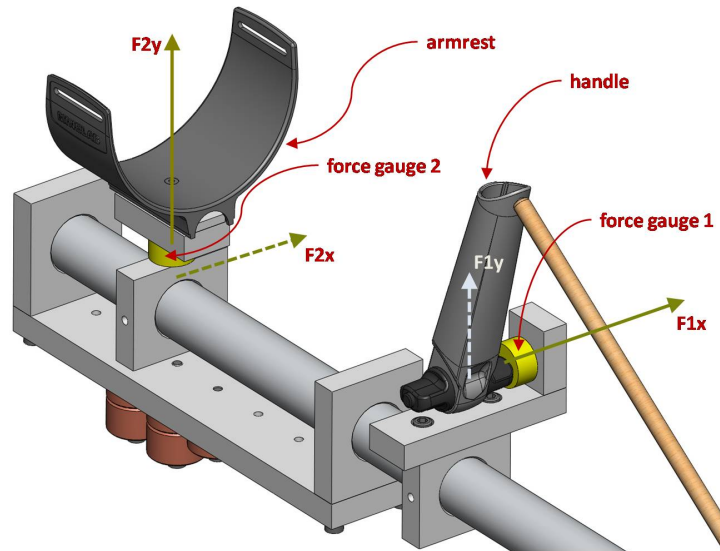


Figure 5.1: Integration of the force sensor network on the dummy detector. Forces are measured orthogonal to the hand and armrest. The unknown components are shown as dashed.

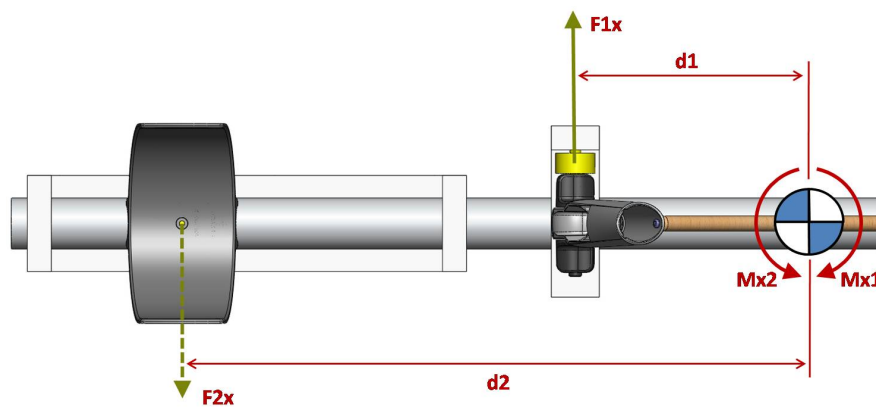


Figure 5.2: The unknown x force component at the armrest is found by taking moments of  $F_{1x}$  and  $F_{2x}$  about the product center-of-gravity.



The  $x$  component of  $F_2$  is found as follows,

$$\begin{aligned}
 \sum M_x &\approx 0 \\
 \Rightarrow F_{1_x}d_1 + F_{2_x}d_2 &\approx 0 \\
 \Rightarrow F_{2_x} &\approx -\frac{F_{1_x}d_1}{d_2}
 \end{aligned} \tag{5.1}$$

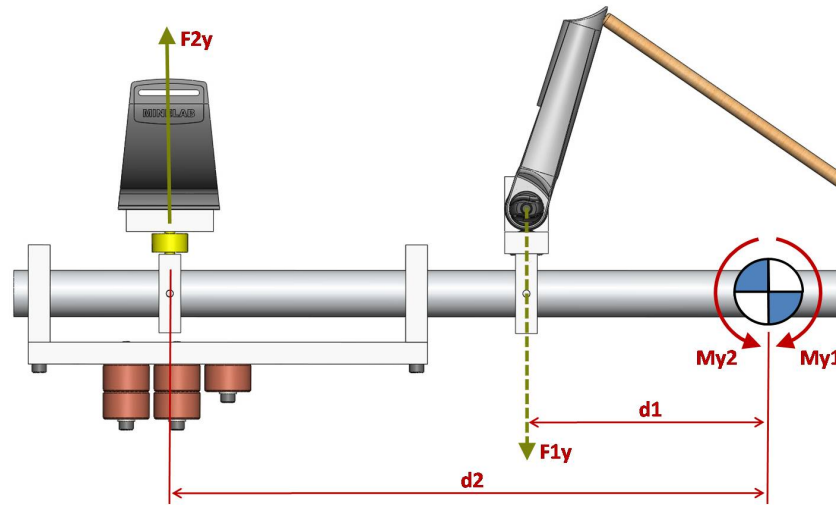


Figure 5.3: The unknown  $y$  force component at the armrest is found by taking moments of  $F_{1_y}$  and  $F_{2_y}$  about the product center-of-gravity.

Similarly the  $y$  component of  $F_1$  is found as follows (refer to figure 5.3),

$$\begin{aligned}
 \sum M_y &\approx 0 \\
 \Rightarrow F_{1_y}d_1 + F_{2_y}d_2 &\approx 0 \\
 \Rightarrow F_{2_y} &\approx -\frac{F_{1_y}d_1}{d_2}
 \end{aligned} \tag{5.2}$$

Equations 5.1 and 5.2 assume the movement of the metal-detector to be rectilinear. In reality, there do exist some small rotations about the center-of-gravity since the coil is moved along an imaginary arc. Hence, it is for this reason that the resolved orthogonal components are only an approximation. The curvature of movement is large however, and to approximate the movement as linear is a reasonable assumption for the purpose

of this study.

An alternative is to use multi-axis force gauges at the handle and armrest. After an extensive search, however, it was found that this type of force-sensor is not readily available, very costly and of an impractical size to be unobtrusively integrated into the construction of the metal-detector dummy.

### 5.2.2 Sensor Selection

The requirement of the force sensors are for robustness, accuracy and cost-effectiveness. The sensors selected that meet these criteria are manufactured by Futek (Futek Advanced Sensor Technology 2010), of which the specifications and calibration sheets are shown in Appendix F. Besides the impressive quality of this component it is also perfectly sized for compact integration into the metal-detector dummy construction (figure 5.5).

An initial concern about the use of uni-axial force gauges is the effect of off-axis, orthogonal force components resulting in bending moments about the base of the sensor as shown in figure 5.4. It was anticipated that this bending moment may illicit an error in the axial force measurement and that, as such, a means of decoupling the orthogonal force components may be required. An example of this is the sensor used in the handle. The forces of interest are axial to the sensor (orthogonal to the hand) but there exists an off-axis force in the partial weight of the detector.

However, after some initial testing of the sensors, it was found that orthogonal off-axis force components did not result in any significant measurement error. This has been put down to the rigidity of the sensor casing and the mounting of the internal piezo components.

It was also initially estimated that peak forces as a result of movement may be in the order of around 100N - 150N and, therefore, the force-sensor selected has a peak measurable force rating of 22kg (220N). However, impacts of the coil have not been taken into account and experiments on impacts still remain to be done in order to establish the magnitude of such a force. It is not yet known whether the dynamic

properties of the selected force sensors are suitable for impacts. The effects of impact on the test system may be a topic for further investigation in the future.

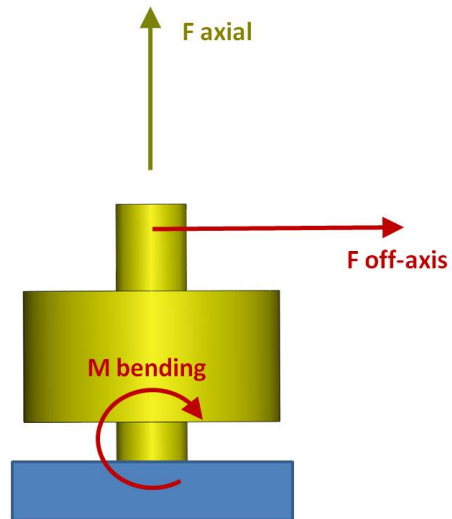


Figure 5.4: Off-axis (orthogonal) force components result in bending moments about the sensor base.

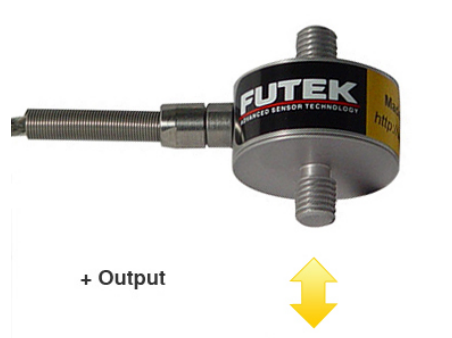


Figure 5.5: The LCM300 miniature load cell from Futek (Futek Advanced Sensor Technology 2010).

### 5.3 Dummy Metal Detector

A requirement of the metal-detector dummy is for it to have modular components for the purpose of positional adjustment. This allows for the quick turnaround of various

inertial configurations in question. All the materials used are relatively inexpensive and some of the components already exist in the form of injection-molded parts. Figures 5.7 and 5.8 show itemised views of the dummy components. The dummy allows for a total system mass range of 1.5kg - 3.0kg which easily accomodates the mass emulation of any detector product currently available.

Custom fabricated mounting blocks machined from high density polyethylene (HDPE) have tapped holes for the use of machine screws that allow for ease of assembly and positional adjustment. A circular plate, emulating the coil, has been machined out of aluminium. It also has tapped holes for the assembly of weights if required to emulate different coil masses.

Universal cylindrical weights have been machined out of copper for placement on the coil and upper-half of the dummy. In this way the system mass may be easily adjusted.

## 5.4 Data Acquisition Hardware

The choice of data acquisition equipment has been driven by quality and compatibility with interface software. The unit of choice for this project is the NI USB 6210, shown in figure 5.6 from National Instruments (National Instruments Corporation n.d.) which is low cost, USB bus powered (which makes it conveniently portable with a laptop) and compatible with LABVIEW. The detailed specifications of the NI USB 6210 may be found in Appendix G. It has ample sampling cabability at 256kS/s, 16 analog inputs, 4 digital inputs and 4 digital outputs.



Figure 5.6: The NI USB 6210 (National Instruments Corporation n.d.) with 256kS/s sampling rate, 16 analog inputs, 4 digital inputs and 4 digital outputs.

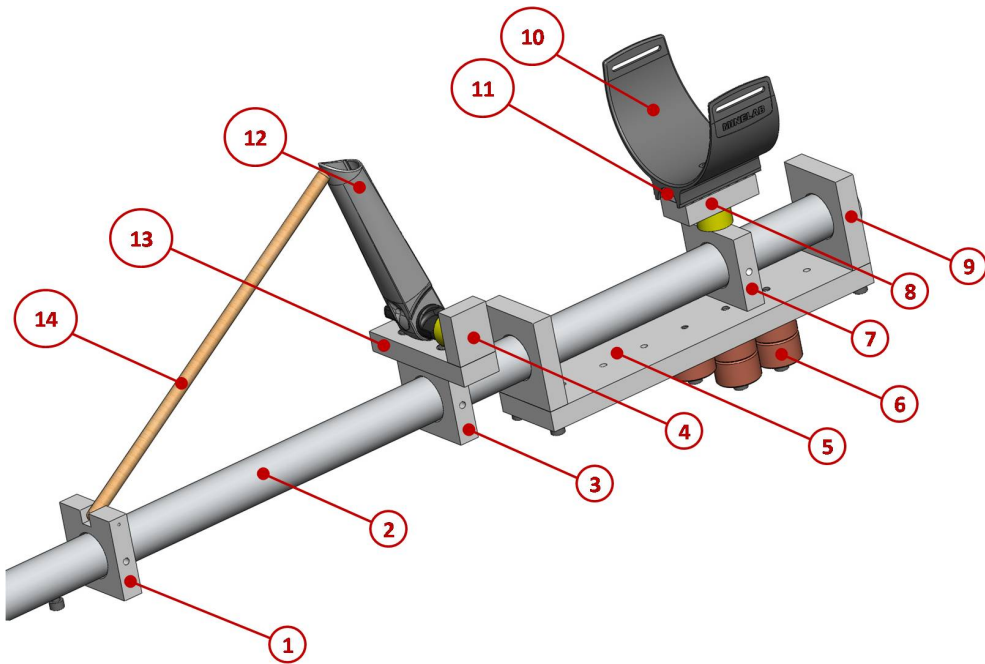


Figure 5.7: Itemised view of the top half of the metal-detector dummy. Refer to table 5.1 for a material list.

Table 5.1: Material list for upper half of dummy detector as shown in figure 5.7.

Item	Description	Material
1	staypole mounting block	15mm x 60mm x 60mm HDPE
2	upper shaft	dia.32mm x 1.1mm x 790mm 6061 aluminium
3	handle mounting block	15mm x 60mm x 60mm HDPE
4	handle bracket piece #1	15mm x 30mm x 38mm HDPE
5	control box base block	15mm x 275mm x 60mm HDPE
6	universal mass (0.127kg)	dia.32mm x 19mm copper
7	armrest mounting block	15mm x 54mm x 60mm HDPE
8	armrest bracket piece #1	15mm x 30mm x 60mm HDPE
9	control box mounting block	15mm x 60mm x 67mm HDPE (2 off)
10	armrest, Xterra	injection moulded ABC PC
11	armrest bracket piece #2	5mm x 24mm x 60 HDPE
12	handle yoke	injection moulded ABS PC
13	handle bracket piece #2	15mm x 30mm x 107mm HDPE
14	staypole	dia.9.5mm x 290mm timber (balsa)

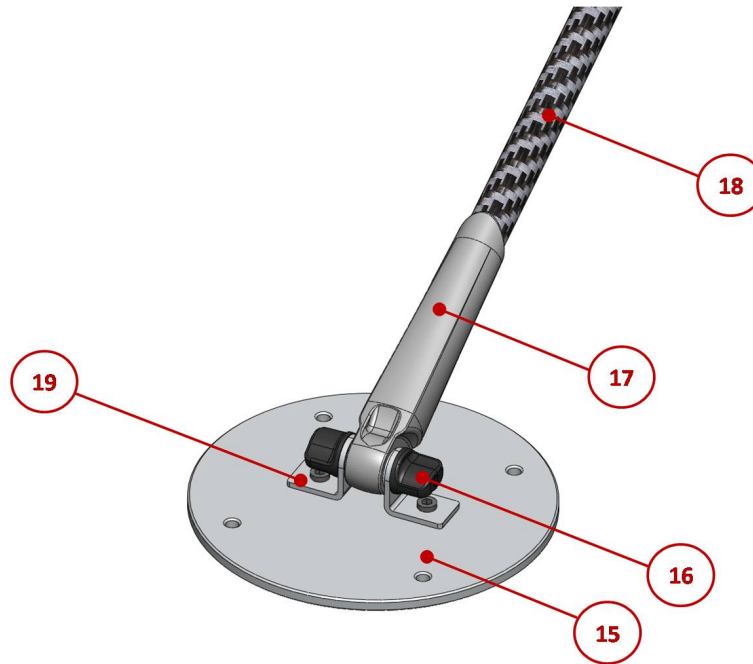


Figure 5.8: Itemised view of the bottom of the metal-detector dummy. Refer to table 5.2 for a material list.

Table 5.2: Material list for lower half of dummy detector as shown in figure 5.8.

Item	Description	Material
15	coil plate	dia.188mm x 6.5mm 6061 aluminium
16	clevis nut	injection moulded ABS PC
17	coil yoke	injection moulded ABS PC
18	lower shaft	Thornel mat VMA carbon fibre
19	coil braket	channel sq.32mm x 3mm x 23mm 6061 aluminium (2 off)

# Chapter 6

## Results

### 6.1 Chapter Overview

This chapter discusses the experimental results to date. These experiments and the evolution of the database with logged ergonomic performance is ongoing. Two experiments have been undertaken thus far on systems both of total mass 2.5kg but of different inertial configurations. The results show that the simulated distribution of muscle activity is indeed different between the two.

For both experiments marker positions were placed at  $-20deg$  and  $20deg$  from the stationary pose. The human subject that undertook the experiment is of 50th percentile build with no known disabilities.

A convergence trace and statistics of the optimal control solution are provided in Appendix H. Above each plot of simulated activations is the evaluated integral of activations used as assessment criteria for this study. A summary table of evaluated integrals is provided for the comparison of configurations 6.2.

Table 6.1: Inertial configurations of two systems of equal mass but different arrangement.

System Mass	Configuration No.	$I_{xx}$ ( $kg.m^2$ )	$I_{yy}$ ( $kg.m^2$ )	$I_{zz}$ ( $kg.m^2$ )
2.5kg	1	0.01	1.09	1.09
2.5kg	2	0.01	1.05	1.05

## 6.2 Experiment 1 Results

The inertial properties of the dummy metal detector for experiment 1 have been set up as shown in table B.1.

### 6.2.1 Shoulder Complex Activations

After experiment 1, the simulated activations for the lumped muscle group of the shoulder complex are shown in the following figure.

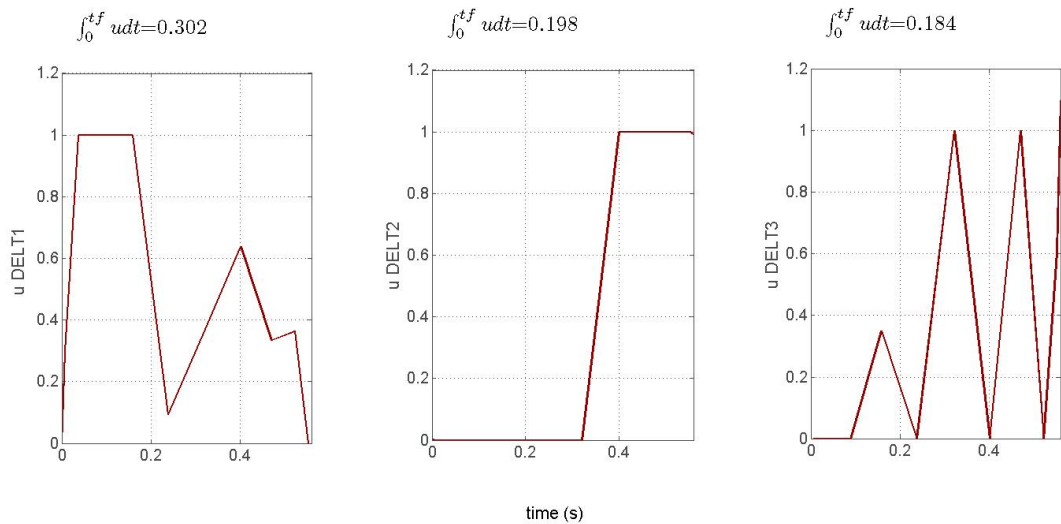


Figure 6.1: Required muscle activations for the lumped muscles of the shoulder complex (experiment #1).



## 6.2.2 Elbow Complex Activations

After experiment 1, the simulated activations for the lumped muscle group of the elbow complex are shown in the following figure.

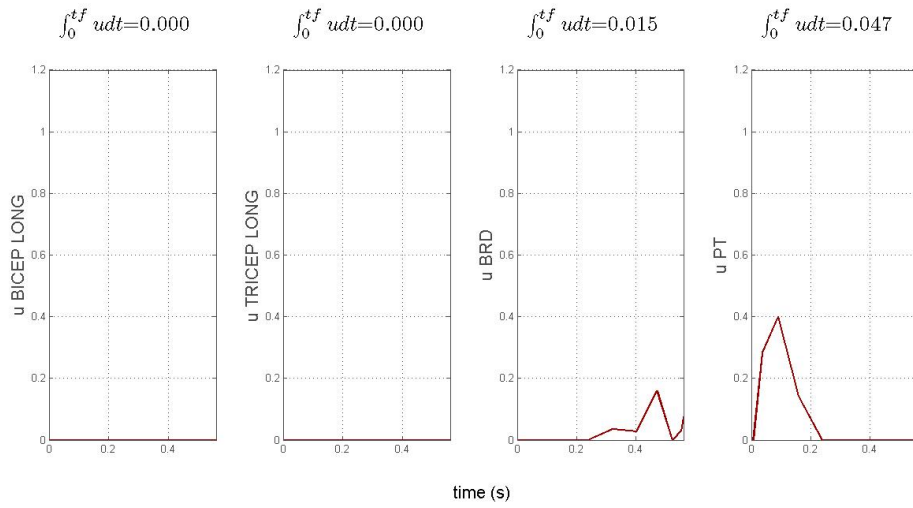


Figure 6.2: Required muscle activations for the lumped muscles of the elbow complex (experiment # 1).

## 6.2.3 Wrist Complex Activations

After experiment 1, the simulated activations for the lumped muscle group of the wrist complex are shown in the following figure.

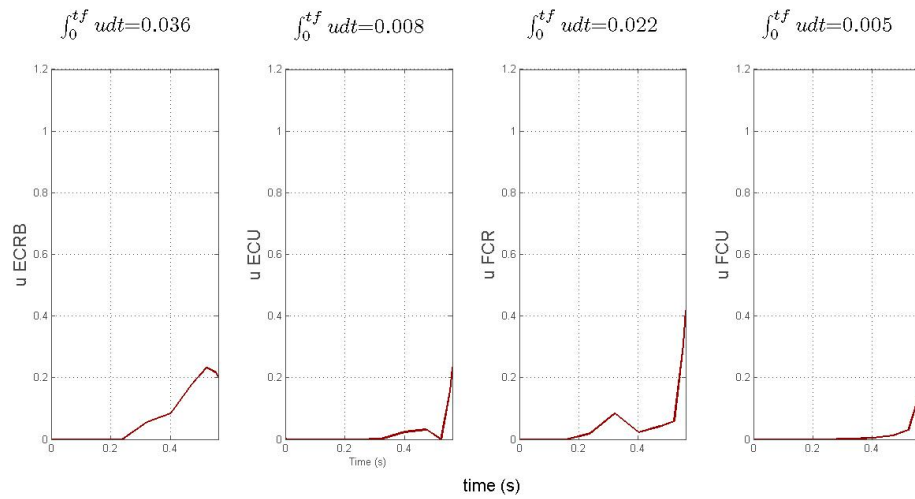


Figure 6.3: Required muscle activations for the lumped muscles of the wrist complex (experiment # 1).

## 6.3 Experiment 2 Results

The inertial properties of the dummy metal detector for experiment 2 have been set up as shown in table B.1.

### 6.3.1 Shoulder Complex Activations

After experiment 2, the simulated activations for the lumped muscle group of the shoulder complex are shown in the following figure.

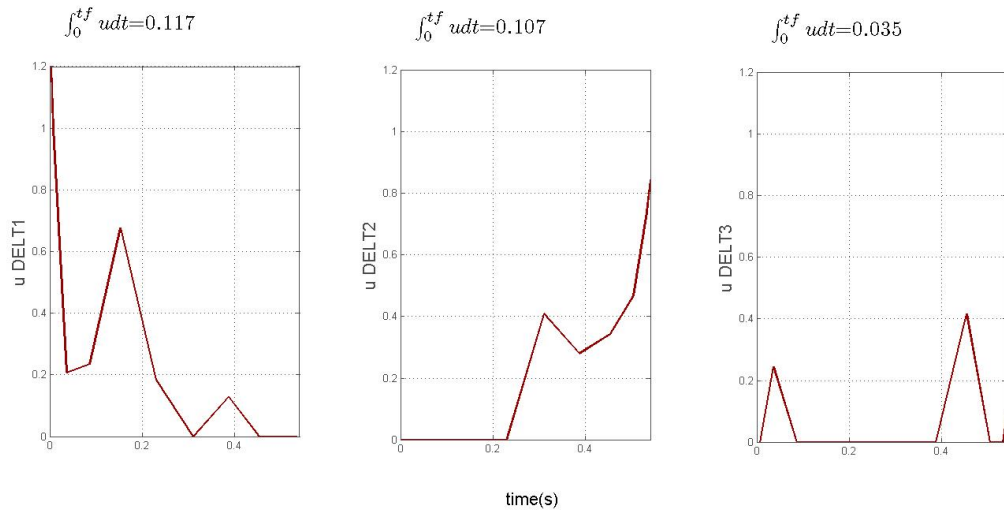


Figure 6.4: Required muscle activations for the lumped muscles of the shoulder complex (experiment #2).

## 6.3.2 Elbow Complex Activations

After experiment 2, the simulated activations for the lumped muscle group of the elbow complex are shown in the following figure.

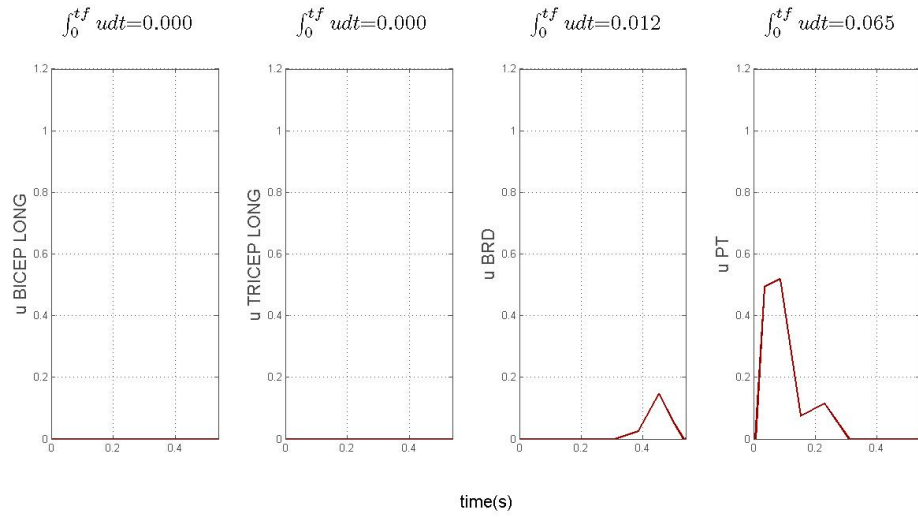


Figure 6.5: Required muscle activations for the lumped muscles of the elbow complex (experiment # 2).

### 6.3.3 Wrist Complex Activations

After experiment 1, the simulated activations for the lumped muscle group of the wrist complex are shown in the following figure.

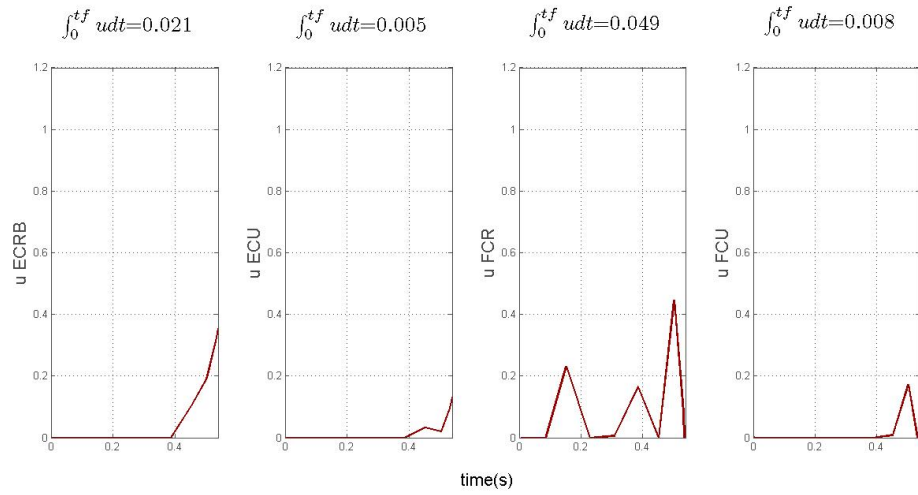


Figure 6.6: Required muscle activations for the lumped muscles of the wrist complex (experiment # 2).

## 6.4 Product Performance: Muscle-Activation Comparisons

As can be seen from table 6.2, the most notable difference is in the level of activations required in the shoulder complex. In summary, the first inertial configuration,  $\mathbf{I}_{\text{config}_1}$ , may indeed result in shoulder fatigue at a faster rate. This has been confirmed, on a cognitive level, by the test subject after swinging the dummy off-line for around 10 minutes. As an example then, if the system mass has to be constrained at 2.5 kg, on this basis one would select an inertial configuration of  $\mathbf{I}_{\text{config}_1}$  for the purposes of design. There do exist, of course, many other inertial configurations for a system of this mass and these will be tested on an ongoing basis.

Table 6.2: The following table shows a comparison of evaluated activation integrals that form a basis for ergonomic assessment.

Mass	Muscle Group	$\mathbf{I}_{\text{config}_1} : \int_{t_0}^{t_f} udt$	$\mathbf{I}_{\text{config}_2} : \int_{t_0}^{t_f} udt$
2.5kg	DELT1	<b>0.302</b>	<b>0.117</b>
	DELT2	<b>0.198</b>	<b>0.107</b>
	DELT3	<b>0.184</b>	<b>0.035</b>
	BICEP LONG	<b>0.000</b>	<b>0.000</b>
	TRICEP LONG	<b>0.000</b>	<b>0.000</b>
	BRD	<b>0.015</b>	<b>0.012</b>
	PT	<b>0.047</b>	<b>0.065</b>
	ECRB	<b>0.036</b>	<b>0.021</b>
	ECU	<b>0.008</b>	<b>0.005</b>
	FCR	<b>0.022</b>	<b>0.049</b>
FCU	<b>0.005</b>	<b>0.008</b>	

The arrangement of elements within the product is left up to the discretion of the design process to converge on the inertial configuration that is most comfortable for the system mass constraint.

## 6.5 Sampling Window Effects

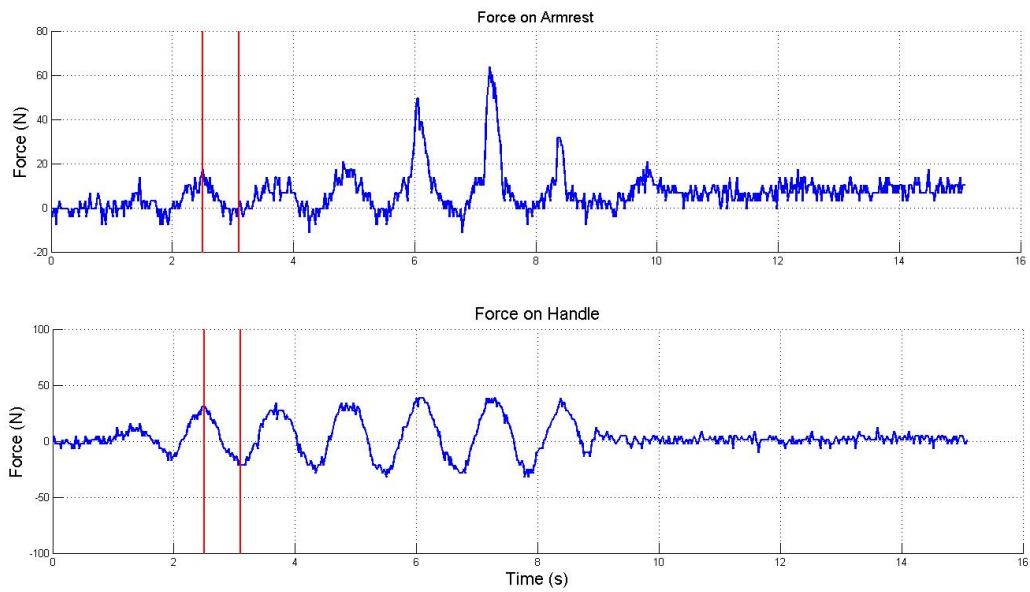


Figure 6.7: Time histories of forces measured at the armrest(top) and handle(bottom). The red bars indicate the position and width of the sampling window.

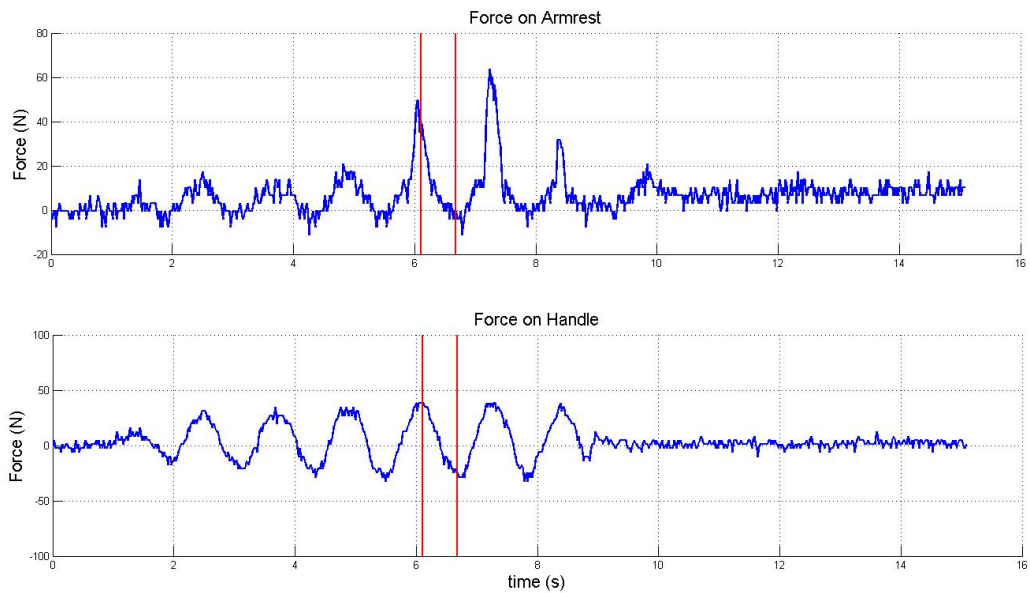


Figure 6.8: The same force-time data but the sampling window has shifted across to include the radical transients experienced at the armrest.

Only about 0.6s of force-time data is sampled for the simulations carried out thus far and this sampling window has been placed at a different position within the same data span for comparison. In figure 6.8 radical transient forces are transmitted through the armrest as a result of the the test subject momentarily engaging in a more erratic swing motion. The force-time data used from the windowing of radical transients does result in markedly different results for the activations required in the shoulder complex.

It is thus important for the test subject to maintain consistent movement throughout the experiments. For this study the position of sampling window should be selected to encapsulate the force-time histories of conventional swing motion. The study may be extended into the future to assess the effects of erratic movement or impacts on the required muscle activations.



## Chapter 7

# Conclusions and Further Work

This chapter focuses on the achievements of the study undertaken and summarises the potential topics for further investigation.

### 7.1 Achievement of Project Objectives

The total mass of a metal-detector product is constrained by the enabling technologies employed by the individual components that make up the system. There is however, flexibility in the positioning of these components that allows for a variation of inertial configurations. The hypothesis established at the beginning of this study proposes that the distribution of muscle activity required to swing the product is a key indicator of ergonomic performance. An attempt to prove this hypothesis was carried out with the achievement of the following objectives:-

1. The design and implementation of a biomechanical test and analysis platform.
2. The initiation of an ergonomic performance database that contains muscle activation data and is indexed by system mass and inertial configurations for that mass. The database is to be of a format that may easily be used as a design tool in future product development.

It has been shown through experiment and user feedback that the simulation of required muscle activity does provide an accurate indication of associated comfort and product ergonomics. However, many more configurations still need to be tested, not just on a 50th percentile subject, but on subjects of varying anthropometries. As such, this study only provides an initiation to the vast combinations of anthropometries and system configurations that form the user-product interface.

## 7.2 Future Work

### 7.2.1 Measures of Accuracy

Thus far, the only measure of simulation accuracy lies in feedback from the user with regards to comfort (or discomfort). This stems from a cognitive response of the user to the dummy metal-detector set to an inertial configuration in question.

A better measure of accuracy may be undertaken with the use of electromyography (EMG). The use of such a system does require a lot more time and resources but will provide valuable insight into the accuracy of the system. Electrode patches are to be located on the upper-extremity targeting the electro-neural behaviour that drives specific muscles. EMG signals may then be compared with simulated muscle activity.

### 7.2.2 Improvement of the Optimal Control Solution Procedure

As mentioned in chapter 3, the symbolic functions defining actuations within the open-kinematic chain (equations 3.25) are rather complicated, involving matrix entries of many trigonometric terms. This does affect the time required for the optimal control solution procedure (especially when implementing numerical finite differencing for the cost-functional gradient). Simplifying the kinematics with the use of Denavit-Hartenberg parameters (Vidyasagar et al. 2006) may be one way of achieving this. Furthermore, the implementation of analytical Hessians and Jacobians, although time-consuming initially, may improve the speed and accuracy of the optimal control solution procedure.

The modified MBSysTran symbolic function, in its current form, is not suitable for use with compact and efficient MATLAB abstract objects of the GPOPS structure. The integration of this symbolic function may be improved in order to make full use of the power and speed of GPOPS.

### 7.2.3 Implementation of Goal-Orientated Movement

In this study we have only been concerned with measured movement and, as such, have disregarded the,  $\phi(\mathbf{x}(t_f))$  *Mayer term* of equation 4.6. However the implementation of GPOPS allows for a scalar entry of this term and thus the optimal control solution for goal-orientated movement is possible. An example of this would be to assign a scalar value of metabolic cost to this term and, in so doing, *prescribe* movement that results in minimum fatigue.

### 7.2.4 Assessment of Coil Movement Under Impact and Over Terrain Variations

In reality, the metal-detector coil is passed over different types of grass, gravel, sand and through water. This may impose many different forces on the arm and it would be useful to understand how terrain variations affect the required muscle activity and if any design improvements can be made with regard to these environmental affects.

### 7.2.5 Implementation of Inertial Motion Sensing

Pseudo motion capture has been used in this study for the sake of time and simplicity (refer to 3). However, it may be useful to implement inertial motion sensing (IMS) into the system. A proposal is to attach IMS units to the hand, forearm and upperarm and to use data fusion for the purposes of accurate motion capture. This has the advantage of detecting subtle movements that may contribute to the long-term fatigue of the user. It also opens up the possibility for ambulatory study since it may be integrated into products in a compact manner for the purposes of field trials.

The proposed motion capture implements an extended Kalman filter coupled with a Newton-Euler inverse dynamics algorithm similar to a process used in a past study for inertial motion sensing of human body motion in sit-to-stand movement (Music et al. 2008). A stochastic process with non-linear state space description and measurement equations is represented as

$$\begin{aligned}\bar{\mathbf{x}}_{k+1} &= f(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \bar{\mathbf{w}}_k) \\ \bar{\mathbf{z}}_k &= \mathbf{h}(\bar{\mathbf{x}}_k, \bar{\mathbf{v}}_k)\end{aligned}$$

where  $f$  relates the state vector  $\bar{\mathbf{x}}$  and the input  $\bar{\mathbf{u}}$  at time-step  $k$  to the state vector at time step  $k+1$ . The measurement vector  $\mathbf{h}$  relates the state vector to the measurement vector  $\mathbf{z}_k$ . Process noise and measurement noise are represented by vectors  $\bar{\mathbf{w}}_k$  and  $\bar{\mathbf{v}}_k$  respectively, and for the purpose of future study could be assumed uncorrelated and Gaussian distributed with zero mean.

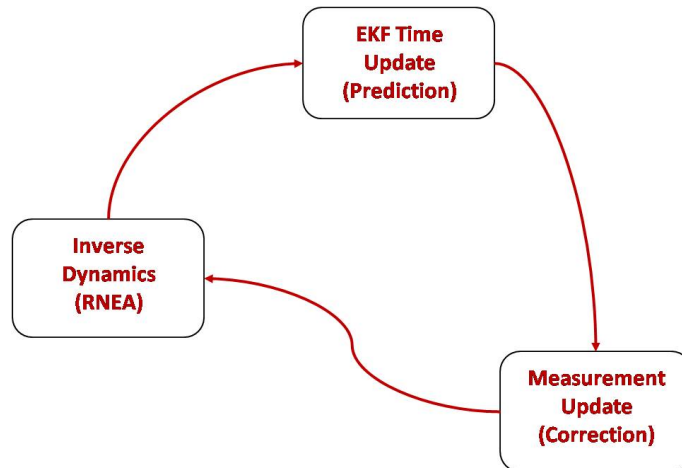


Figure 7.1: Flowchart of extended Kalman filter (EKF) with recursive Newton-Euler algorithm in-the-loop.

For four accelerometers the state and measurement vectors may be represented as

$$\mathbf{x}_k = \begin{bmatrix} \theta_1 \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \vdots \\ \theta_n \\ \dot{\theta}_n \\ \ddot{\theta}_n \end{bmatrix}, \quad \mathbf{z}_k = \begin{bmatrix} \theta_1 \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \vdots \\ \theta_n \\ \dot{\theta}_n \\ \ddot{\theta}_n \\ a_{1r} \\ a_{1t} \\ \vdots \\ a_{4r} \\ a_{4t} \\ M_1 \\ \vdots \\ M_k \end{bmatrix} \quad (7.1)$$

Where  $\theta_i$ ,  $\dot{\theta}_i$ ,  $\ddot{\theta}_i$  is the angular displacement, velocity and acceleration of joint  $i$  respectively,  $a_{j_t}$  and  $a_{j_r}$  are the tangential and radial components of accelerometer measurements respectively and  $M_i$  is the moment about joint  $i$ . Elements of  $z_k$  may be found by direct sensor measurement except for the  $M$  (moment) elements which are found indirectly via an inverse dynamics calculation from the estimated kinematic values and external force-time data.

The recursive Newton-Euler algorithm (RNEA), which calculates required joint torques from given kinematic values and externally applied forces (Featherstone & Orin 2008), is used per-time-step in the EKF loop as shown in figure 7.1.

# References

- Beavis, D. & Slade, I. M. (2003), 'Ergonomics - costs and benefits', *Journal of Applied Ergonomics* **34**, 413–418.
- CEREM, U. C. d. L. (2008), 'Multibody Systems Online', <http://www.robotran.be/>.
- Chizeck, H. J., Crago, P. E. & Kofman, L. S. (1988), 'Robust Closed-Loop Control of Isometric Muscle Force using Pulsewidth Modulation', *IEEE Transactions on Biomedical Engineering* **35**(7), 510–517.
- Delp, S. L., Anderson, F. C., Arnold, A. S., Loan, P., Habib, A., Chand, J. T., Guendelman, E. & Thelen, D. G. (2007), 'OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement', *IEEE Transactions on Biomedical Engineering* **54**(11), 1940–1950.
- Featherstone, R. & Orin, D. E. (2008), Dynamics, in 'Springer Handbook of Robotics', Springer - Heidelberg, Berlin, pp. 35–65. Edited by B. Siciliano and O. Khatib.
- Futek Advanced Sensor Technology (2010), 'Mini Tension and Compression Load Cell', <http://www.futek.com/product.aspx?stock=FSH02630>.
- Holzbour, K. R. S., Murray, W. M. & Delp, S. L. (2005), 'A Model of the Upper Extremity for Simulating Musculoskeletal Surgery and Analyzing Neuromuscular Control', *Annals of Biomedical Engineering* pp. 829–840.
- Kaplan, M. L. & Heegard, J. H. (2002), 'Second-Order Optimal Control Algorithm for Complex Systems', *International Journal for Numerical Methods in Engineering* (53), 2043–2060.

- Mayagoitia, R. E., Nene, A. V. & Veltink, P. (2002), 'Accelerometer and Rate Gyroscope Measurement of Kinematics: An Inexpensive Alternative to Optical Motion Analysis Systems', *Journal of Biomechanics Vol. 35* **35**(3), 537–542.
- Menegaldo, L. L., Fleury, A. T. & Weber, H. I. (2003), 'Biomechanical Modeling and Optimal Control of Human Posture', *Journal of Biomechanics* (36), 1701–1712.
- Music, J., Kamnik, R. & Munih, M. (2008), 'Model Based Inertial Sensing of Human Body Motion Kinematics in Sit-to-Stand Movement', *Simulation Modelling Practice and Theory* **16**, 933–944.
- National Instruments Corporation (n.d.), 'Nhi USB 6210'.
- Rao, A. V., Benson, D. A., Darby, C. L., Patterson, M. A., Francolin, C. & Huntington, G. T. (2010), 'Algorithm:GPOPS, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems using the Gauss Pseudospectral method', *ACM Transactions on Mathematical Software* **37**(2), 221–239.
- Rao, A. V., Benson, D., Darby, C. L., Patterson, M., Francolin, C., Sanders, I. & Huntington, G. T. (2010), 'GPOPS: Open-Source General Pseudospectral Optimal Control Software', <http://www.gpops.org/>.
- Samin, J. & Fisette, P. (2003), *Symbolic Modeling of Multibody Systems*, Springer.
- Stelzer, M. & von Stryk, O. (2006), 'Efficient Forward Dynamics Simulation and Optimization of Human Body Dynamics', *ZAMM - Journal of Applied Mathematics and Mechanics* **86**, 828–840.
- Thelen, D. G., Anderson, F. C. & Delp, S. L. (2003), 'Generating Dynamic Simulations of Movement using Computed Muscle Control', *Journal of Biomechanics* (36), 321–328.
- Vicon (n.d.), 'Motion Capture Systems from Vicon', <http://www.vicon.com/>.
- Vidyasagar, M., Hutchinson, S. & Spong, M. W. (2006), *Robot Dynamics and Control*, John Wiley and Sons - Hoboken, NJ.
- Zajack, F. E. (1989), Muscle and Tendon: Properties, Models, Scaling and Application to Biomechanics and Motor Control, *in* 'Critical Reviews in Biomedical Engineering', CRC Press - Boca Raton, pp. 35–65. Edited by J. R. Bourne.

- 
- Zhou, H. & Hu, H. (2005), Kinematic Model Aided Inertial Motion Tracking of Human Upper Limb, *in* 'Proceedings of the 2005 IEEE International Conference on Information Acquisition', Hong Kong, Macua, China, pp. 150–155.



Appendix A

Project Specification

University of Southern Queensland  
FACULTY OF ENGINEERING AND SURVEYING

**ENG4111/4112 Research Project**  
**PROJECT SPECIFICATION**

FOR: **Brendan SMITH**

TOPIC: A BIOMECHANICAL ANALYSIS FOR IMPROVED  
ERGONOMICS IN METAL DETECTOR PRODUCTS USING  
TRAJECTORY OPTIMIZATION

SUPERVISOR: Dr. Ahmad Sharifian

SPONSERSHIP: Minelab Electronics Division, Codan Pty Ltd

PROJECT MANAGER: Paul Congdon (Minelab Electronics)

PROJECT AIM: 1) To develop a biomechanical test and analysis system for the  
improvement of metal detector ergonomics with respect to the  
human hand-arm-shoulder extremity.  
2) To initiate a database of optimal product inertias for given  
product masses.

CONFIDENTIALITY: Not for public distribution


**PROGRAMME (Issue B, 2 August 2010): -**

- 1) Design a simple sensor network model of two force gauges for force capture.
- 2) Develop a theoretical state space model of a 7 degree of freedom biomechanical hand-arm-shoulder system based on lagrangian dynamics.
- 3) Utilize a symbolic tool for processing state-dependant inertia and coriolis matrices in part of a Matlab implementation.
- 4) Research past studies in biomechanics to investigate muscle models and simplify to include in an optimal control problem (OCP).
- 5) Formulate OCP and utilize trajectory optimization (GPOPS and SNOPT) in a Matlab implementation with 3D graphical user interface (GUI).
- 6) Design a standardized biomechanical experiment and perform on various human subjects.
- 7) Design and construct a metal detector dummy with configurable inertia, integrated sensor network and data acquisition hardware.

- 8) Design a standardized biomechanical experiment and perform on various human subjects.

If time permits:

- 9) Introduce terrain variations (e.g. thick grass, stubble etc) that would induce different reaction forces on the coil-and-shaft assembly.

  
\_\_\_\_\_ (student)  
Date: 02/03/10

AGREED \_\_\_\_\_ (project manager)  
Date:

\_\_\_\_\_ (supervisor)  
Date:

Examiner / Co-Examiner: \_\_\_\_\_

## Appendix B

# Lumped Muscle Parameters

Table B.1: Muscle attachment data (Holzbaur et al. 2005) for major muscles used as lumped actuators. The attachment point coordinates (in meters) are with respect to the proximal and distal reference frames as indicated.

Muscle	Abbreviation	Peak Force(N)	Attachment Points [x y z]
<i>Anterior Deltoid</i>	DELT1	142.6	prox 0.00896 -0.11883 0.00585 prox 0.01623 -0.11033 0.00412 dist 0.04347 -0.03252 0.00099 dist -0.01400 0.01106 0.08021
<i>Middle Deltoid</i>	DELT2	1142.6	prox 0.00461-0.13611 0.00560 prox 0.02720 0.00535 0.04807 dist 0.00005 0.00294 0.02233 dist -0.01078 -0.00034 0.00620
<i>Posterior Deltoid</i>	DELT3	259.9	prox -0.05573 0.00122 -0.02512 prox -0.07247 -0.03285 -0.03285 dist 0.00206 -0.07602 0.01045
<i>Tricep Long</i>	TRICEP LONG	798.5	prox -0.04565 -0.04073 -0.01377 prox -0.02714 -0.11441 -0.00664 prox -0.03184 -0.22637 -0.01217 prox -0.01743 -0.26757 -0.01208 dist -0.02190 0.01046 -0.00078
<i>Bicep Long</i>	BICEP LONG	624.3	prox -0.03123 -0.02353 -0.01305 prox -0.02094 -0.01309 -0.00461 prox 0.02131 0.01793 0.01028 prox 0.02378 -0.00511 0.01201 prox 0.01345 -0.02827 0.00136 dist 0.01068 -0.07736 -0.00165 dist 0.01703 -0.12125 0.00024 dist 0.02280 -0.17540 -0.00630 dist -0.00200 -0.03750 -0.00200
<i>Brachioradialis</i>	BRD	261.3	prox -0.00980 -0.19963 0.00223 dist 0.03577 -0.12742 0.02315 dist 0.04190 -0.22100 0.02240

<b>Muscle</b>	<b>Abbreviation</b>	<b>Peak Force(N)</b>	<b>Attachment Points [x y z]</b>
<i>Pronator teres</i>	PT	566.2	prox 0.02240 -0.27590 -0.03650 prox 0.00846 -0.03373 -0.01432 prox 0.00464 -0.00026 -0.00214 dist 0.02360 -0.09340 0.00940 dist 0.02540 -0.10880 0.01980
<i>Extensor CRB</i>	ECRB	100.5	prox 0.01349 -0.29048 0.01698 dist 0.02905 -0.13086 0.02385 dist 0.03549 -0.22805 0.03937 dist 0.00452 0.02788 0.00112
<i>Extensor CU</i>	ECU	93.2	prox 0.00083 -0.28955 0.01880 prox -0.01391 -0.03201 0.02947 prox -0.01705 -0.05428 0.02868 dist -0.01793 -0.09573 0.03278 dist -0.01421 -0.22696 0.03481 dist -0.00061 0.02067 0.00294
<i>Flexor CR</i>	FCR	74	prox 0.00758 -0.27806 -0.03705 prox 0.02110 -0.21943 0.00127 dist -0.00345 0.01918 -0.00921
<i>Flexor CU</i>	FCU	128.9	prox 0.00219 -0.27740 -0.03880 prox 0.00549 0.00000 0.00000 dist 0.01082 -0.22327 0.00969 dist 0.00154 0.01703 -0.00330

## Appendix C

# Symbolic Functions of Moment Arm and Muscle Unit Vectors for DELTA1

## Muscle Moment Arm Entries

```
dataMus{1}.rxfunc = @(q1, q2, q3) ...
    ((20*cos(q3)*cos(q2) -
70*sin(q3)*cos(q2)+20*sin(q2)) * (100+20*cos(q3)*cos(q2) -...
    70*sin(q3)*cos(q2)+20*sin(q2)) + (-
20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*sin(q1) -...
    70*cos(q3)*cos(q1)-70*sin(q3)*sin(q2)*sin(q1)-20*cos(q2)*sin(q1)) * (10-
20*sin(q3)*...
    cos(q1)+20*cos(q3)*sin(q2)*sin(q1)-70*cos(q3)*cos(q1) -
70*sin(q3)*sin(q2)*sin(q1)-20*...

cos(q2)*sin(q1)) + (20*sin(q3)*sin(q1)+20*cos(q3)*sin(q2)*cos(q1)+70*cos(q3)
*sin(q1)-70*...
    sin(q3)*sin(q2)*cos(q1) -
20*cos(q2)*cos(q1)) * (10+20*sin(q3)*sin(q1)+20*cos(q3)*sin(q2)*...
    cos(q1)+70*cos(q3)*sin(q1)-70*sin(q3)*sin(q2)*cos(q1) -
20*cos(q2)*cos(q1)) / ((100+20*cos(q3)*...
    cos(q2)-70*sin(q3)*cos(q2)+20*sin(q2))^2+(10-
20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*sin(q1)-70*...
    cos(q3)*cos(q1)-70*sin(q3)*sin(q2)*sin(q1) -
20*cos(q2)*sin(q1))^2+(10+20*sin(q3)*sin(q1)+20*...
    cos(q3)*sin(q2)*cos(q1)+70*cos(q3)*sin(q1)-70*sin(q3)*sin(q2)*cos(q1) -
20*cos(q2)*cos(q1))^2)*...
    (100+20*cos(q3)*cos(q2)-70*sin(q3)*cos(q2)+20*sin(q2)) -
20*cos(q3)*cos(q2)+70*sin(q3)*cos(q2)-20*sin(q2);

dataMus{1}.ryfunc = @(q1, q2, q3) ...
    ((20*cos(q3)*cos(q2) -
70*sin(q3)*cos(q2)+20*sin(q2)) * (100+20*cos(q3)*cos(q2) -
70*sin(q3)*cos(q2)+...
    20*sin(q2)) + (-20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*sin(q1) -
70*cos(q3)*cos(q1)-70*sin(q3)*sin(q2)*...
    sin(q1)-20*cos(q2)*sin(q1)) * (10-
20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*sin(q1)-70*cos(q3)*cos(q1)-70*...
    sin(q3)*sin(q2)*sin(q1) -
20*cos(q2)*sin(q1)) + (20*sin(q3)*sin(q1)+20*cos(q3)*sin(q2)*cos(q1)+70*cos(
q3)*...
    sin(q1)-70*sin(q3)*sin(q2)*cos(q1) -
20*cos(q2)*cos(q1)) * (10+20*sin(q3)*sin(q1)+20*cos(q3)*sin(q2)*cos(q1)+...
    70*cos(q3)*sin(q1)-70*sin(q3)*sin(q2)*cos(q1) -
20*cos(q2)*cos(q1)) / ((100+20*cos(q3)*cos(q2)-70*sin(q3)*...
    cos(q2)+20*sin(q2))^2+(10-
20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*sin(q1)-70*cos(q3)*cos(q1) -
70*sin(q3)*...
    sin(q2)*sin(q1) -
20*cos(q2)*sin(q1))^2+(10+20*sin(q3)*sin(q1)+20*cos(q3)*sin(q2)*cos(q1)+70
*cos(q3)*...
    sin(q1)-70*sin(q3)*sin(q2)*cos(q1)-20*cos(q2)*cos(q1))^2 * (10-
20*sin(q3)*cos(q1)+20*cos(q3)*sin(q2)*...
    sin(q1)-70*cos(q3)*cos(q1)-70*sin(q3)*sin(q2)*sin(q1) -
20*cos(q2)*sin(q1)) + 20*sin(q3)*cos(q1)-20*...
```



$\cos(q_3) \sin(q_2) \sin(q_1) + 70 \cos(q_3) \cos(q_1) + 70 \sin(q_3) \sin(q_2) \sin(q_1) + 20 \cos(q_2) \sin(q_1);$

```
dataMus{1}.rzfunc = @(q1, q2, q3) ...  
    ((20*cos(q3)*cos(q2) -  
70*sin(q3)*cos(q2) + 20*sin(q2)) * (100 + 20*cos(q3)*cos(q2) -  
70*sin(q3)*cos(q2) + 20*...  
    sin(q2)) + (-20*sin(q3)*cos(q1) + 20*cos(q3)*sin(q2)*sin(q1) -  
70*cos(q3)*cos(q1) - 70*sin(q3)*sin(q2)*...  
    sin(q1) - 20*cos(q2)*sin(q1)) * (10 -  
20*sin(q3)*cos(q1) + 20*cos(q3)*sin(q2)*sin(q1) - 70*cos(q3)*cos(q1) - 70*...  
    sin(q3)*sin(q2)*sin(q1) -  
20*cos(q2)*sin(q1)) + (20*sin(q3)*sin(q1) + 20*cos(q3)*sin(q2)*cos(q1) + 70*cos(q3)*...  
    sin(q1) - 70*sin(q3)*sin(q2)*cos(q1) -  
20*cos(q2)*cos(q1)) * (10 + 20*sin(q3)*sin(q1) + 20*cos(q3)*sin(q2)*...  
    cos(q1) + 70*cos(q3)*sin(q1) - 70*sin(q3)*sin(q2)*cos(q1) -  
20*cos(q2)*cos(q1)) / ((100 + 20*cos(q3)*cos(q2) - 70*...  
    sin(q3)*cos(q2) + 20*sin(q2))^2 + (10 -  
20*sin(q3)*cos(q1) + 20*cos(q3)*sin(q2)*sin(q1) - 70*cos(q3)*cos(q1) -  
70*sin(q3)*...  
    sin(q2)*sin(q1) -  
20*cos(q2)*sin(q1))^2 + (10 + 20*sin(q3)*sin(q1) + 20*cos(q3)*sin(q2)*cos(q1) + 70  
*cos(q3)*sin(q1) - 70*...  
    sin(q3)*sin(q2)*cos(q1) -  
20*cos(q2)*cos(q1))^2 * (10 + 20*sin(q3)*sin(q1) + 20*cos(q3)*sin(q2)*cos(q1) + 7  
0*cos(q3)*...  
    sin(q1) - 70*sin(q3)*sin(q2)*cos(q1) - 20*cos(q2)*cos(q1)) -  
20*sin(q3)*sin(q1) - 20*cos(q3)*sin(q2)*cos(q1) - 70*...  
    cos(q3)*sin(q1) + 70*sin(q3)*sin(q2)*cos(q1) + 20*cos(q2)*cos(q1);
```

```
dataMus{1}.mxfunc = @(q1, q2, q3) ...  
    (-20*cos(q3)*cos(q2) + 70*sin(q3)*cos(q2) - 20*sin(q2) - 100) / ((-  
20*cos(q3)*cos(q2) + 70*sin(q3)*...  
    cos(q2) - 20*sin(q2) - 100)^2 + (20*sin(q3)*cos(q1) -  
20*cos(q3)*sin(q2)*sin(q1) + 70*cos(q3)*cos(q1) + ...  
    70*sin(q3)*sin(q2)*sin(q1) + 20*cos(q2)*sin(q1) - 10)^2 + (-  
20*sin(q3)*sin(q1) - 20*cos(q3)*sin(q2)*...  
    cos(q1) -  
70*cos(q3)*sin(q1) + 70*sin(q3)*sin(q2)*cos(q1) + 20*cos(q2)*cos(q1) -  
10)^2)^(1/2);
```

## Muscle Unit Vector Entries

```
dataMus{1}.mxfunc = @(q1, q2, q3) ...
    (-20*cos(q3)*cos(q2)+70*sin(q3)*cos(q2)-20*sin(q2)-100)/((-
20*cos(q3)*cos(q2)+70*sin(q3)*...
    cos(q2)-20*sin(q2)-100)^2+(20*sin(q3)*cos(q1)-
20*cos(q3)*sin(q2)*sin(q1)+70*cos(q3)*cos(q1)+...
    70*sin(q3)*sin(q2)*sin(q1)+20*cos(q2)*sin(q1)-10)^2+(-
20*sin(q3)*sin(q1)-20*cos(q3)*sin(q2)*...
    cos(q1)-
70*cos(q3)*sin(q1)+70*sin(q3)*sin(q2)*cos(q1)+20*cos(q2)*cos(q1)-
10)^2)^(1/2);

dataMus{1}.myfunc = @(q1, q2, q3) ...
    (20*sin(q3)*cos(q1)-
20*cos(q3)*sin(q2)*sin(q1)+70*cos(q3)*cos(q1)+70*sin(q3)*sin(q2)*sin(q1)+.
..
    20*cos(q2)*sin(q1)-10)/((-20*cos(q3)*cos(q2)+70*sin(q3)*cos(q2)-
20*sin(q2)-100)^2+(20*sin(q3)*...
    cos(q1)-
20*cos(q3)*sin(q2)*sin(q1)+70*cos(q3)*cos(q1)+70*sin(q3)*sin(q2)*sin(q1)+2
0*cos(q2)*...
    sin(q1)-10)^2+(-20*sin(q3)*sin(q1)-20*cos(q3)*sin(q2)*cos(q1)-
70*cos(q3)*sin(q1)+70*sin(q3)*...
    sin(q2)*cos(q1)+20*cos(q2)*cos(q1)-10)^2)^(1/2);

dataMus{1}.mzfunc = @(q1, q2, q3) ...
    (-20*sin(q3)*sin(q1)-20*cos(q3)*sin(q2)*cos(q1)-
70*cos(q3)*sin(q1)+70*sin(q3)*sin(q2)*cos(q1)+...
    20*cos(q2)*cos(q1)-10)/((-20*cos(q3)*cos(q2)+70*sin(q3)*cos(q2)-
20*sin(q2)-100)^2+(20*sin(q3)*...
    cos(q1)-
20*cos(q3)*sin(q2)*sin(q1)+70*cos(q3)*cos(q1)+70*sin(q3)*sin(q2)*sin(q1)+2
0*cos(q2)*...
    sin(q1)-10)^2+(-20*sin(q3)*sin(q1)-20*cos(q3)*sin(q2)*cos(q1)-
70*cos(q3)*sin(q1)+70*sin(q3)*...
    sin(q2)*cos(q1)+20*cos(q2)*cos(q1)-10)^2)^(1/2);
```

## Appendix D

# Modified Symbolic Functions for the Shoulder, Elbow and Wrist Complex

## Modified Shoulder Symbolic Function

```
function [iM] = dirdyna_shoulder (q, qd, ...
    frcx1, frcy1, frcz1, ...
    trqx1, trqy1, trqz1)

global I1xx I1yy I1zz;
global X1 Z1;
global modeldata;

s.g = [0; 0; -9.8];
% mass of static model
ms = modeldata.m1+modeldata.m2+modeldata.m3; %kg
s.m = [0 0 ms];
% distance to CG
s.l = [0 0 X1; ...
       0 0 0; ...
       0 0 Z1];

s.In = [0 0 I1xx; ...
        0 0 0; ...
        0 0 0; ...
        0 0 0; ...
        0 0 I1yy; ...
        0 0 0; ...
        0 0 0; ...
        0 0 0; ...
        0 0 I1zz];

s.frc = [0 0 frcx1; ...
         0 0 frcy1; ...
         0 0 frcz1];

s.trq = [0 0 trqx1; ...
         0 0 trqy1; ...
         0 0 trqz1];

C1 = cos(q(1));
S1 = sin(q(1));
C2 = cos(q(2));
S2 = sin(q(2));
C3 = cos(q(3));
S3 = sin(q(3));

% == Block_0_1_0_0_0_1 ==

% Forward Kinematics

AlF21 = -(s.g(2)*C1+s.g(3)*S1);
AlF31 = s.g(2)*S1-s.g(3)*C1;
OM12 = qd(1)*C2;
OM32 = qd(1)*S2;
```

```

OpF12 = -qd(1)*qd(2)*S2;
OpF32 = qd(1)*qd(2)*C2;
BS32 = OM12*OM32;
AlF12 = -(s.g(1)*C2+AlF31*S2);
AlF32 = -(s.g(1)*S2-AlF31*C2);
OM13 = qd(2)*S3+OM12*C3;
OM23 = qd(2)*C3-OM12*S3;
OM33 = qd(3)+OM32;
OpF13 = -(qd(3)*OM12*S3-C3*(OpF12+qd(2)*qd(3)));
OpF23 = -(qd(3)*OM12*C3+S3*(OpF12+qd(2)*qd(3)));
BS23 = OM13*OM23;
BS33 = OM13*OM33;
BS63 = OM23*OM33;
OpM13_1 = C2*C3;
OpM23_1 = -C2*S3;

% = = Block_0_2_0_1_0_1 = =

% Backward Dynamics

FA13 = -(s.frc(1,3)-s.m(3)*(s.l(2,3)*(BS23-
OpF32)+s.l(3,3)*(BS33+OpF23)+AlF12*C3+AlF21*S3-
s.l(1,3)*(OM23*OM23+OM33*OM33)));
FA23 = -(s.frc(2,3)+s.m(3)*(s.l(2,3)*(OM13*OM13+OM33*OM33)-
s.l(3,3)*(BS63-OpF13)+AlF12*S3-AlF21*C3-s.l(1,3)*(BS23+OpF32)));
FA33 = -(s.frc(3,3)-s.m(3)*(AlF32+s.l(2,3)*(BS63+OpF13)-
s.l(3,3)*(OM13*OM13+OM23*OM23)+s.l(1,3)*(BS33-OpF23)));
CF13 = -(s.trq(1,3)-s.In(1,3)*OpF13-s.In(2,3)*OpF23-s.In(3,3)*OpF32-
s.l(2,3)*FA33+s.l(3,3)*FA23-
OM23*(s.In(3,3)*OM13+s.In(6,3)*OM23+s.In(9,3)*...
OM33)+OM33*(s.In(2,3)*OM13+s.In(5,3)*OM23+s.In(6,3)*OM33));
CF23 = -(s.trq(2,3)-s.In(2,3)*OpF13-s.In(5,3)*OpF23-s.In(6,3)*OpF32-
s.l(3,3)*FA13+FA33*s.l(1,3)+OM13*(s.In(3,3)*OM13+s.In(6,3)*OM23+s.In(9,3)
*...
OM33)-OM33*(s.In(1,3)*OM13+s.In(2,3)*OM23+s.In(3,3)*OM33));
CF33 = -(s.trq(3,3)-s.In(3,3)*OpF13-s.In(6,3)*OpF23-
s.In(9,3)*OpF32+s.l(2,3)*FA13-FA23*s.l(1,3)-
OM13*(s.In(2,3)*OM13+s.In(5,3)*OM23+s.In(6,3)*...
OM33)+OM23*(s.In(1,3)*OM13+s.In(2,3)*OM23+s.In(3,3)*OM33));
FB13_1 = -s.m(3)*(s.l(2,3)*S2-s.l(3,3)*OpM23_1);
FB23_1 = -s.m(3)*(s.l(3,3)*OpM13_1-s.l(1,3)*S2);
FB33_1 = s.m(3)*(s.l(2,3)*OpM13_1-OpM23_1*s.l(1,3));
CM13_1 =
s.In(1,3)*OpM13_1+s.In(2,3)*OpM23_1+s.In(3,3)*S2+s.l(2,3)*FB33_1-
s.l(3,3)*FB23_1;
CM23_1 =
s.In(2,3)*OpM13_1+s.In(5,3)*OpM23_1+s.In(6,3)*S2+s.l(3,3)*FB13_1-
FB33_1*s.l(1,3);
CM33_1 = s.In(3,3)*OpM13_1+s.In(6,3)*OpM23_1+s.In(9,3)*S2-
s.l(2,3)*FB13_1+FB23_1*s.l(1,3);
FB13_2 = s.l(3,3)*s.m(3)*C3;
FB23_2 = -s.l(3,3)*s.m(3)*S3;
FB33_2 = s.m(3)*(s.l(2,3)*S3-s.l(1,3)*C3);
CM33_2 = s.In(3,3)*S3+s.In(6,3)*C3-s.l(2,3)*FB13_2+FB23_2*s.l(1,3);

```

```

CM33_3 = s.In(9,3)+s.l(2,3)*s.l(2,3)*s.m(3)+s.m(3)*s.l(1,3)*s.l(1,3);
FA12_ = -(s.frc(1,2)-s.m(2)*(AlF12-
s.l(1,2)*(qd(2)*qd(2)+OM32*OM32)+s.l(3,2)*BS32));
FA22 = -(s.frc(2,2)-s.m(2)*(AlF21-
qd(1)*qd(1)*s.l(2,2)+s.l(1,2)*(OpF32+qd(2)*OM12)-s.l(3,2)*(OpF12-
qd(2)*OM32)));
FA32 = -(s.frc(3,2)-s.m(2)*(AlF32+s.l(1,2)*BS32-
s.l(3,2)*(qd(2)*qd(2)+OM12*OM12)));
CF22 = -(s.trq(2,2)-s.In(6,2)*OpF32+s.l(1,2)*FA32-s.l(3,2)*FA12-CF13*S3-
CF23*C3-OM32*(qd(2)*s.In(2,2)+s.In(1,2)*OM12+s.In(3,2)*OM32-s.In(9,2)*...
OM12));
FB12_1 = -s.l(2,2)*s.m(2)*S2;
FB22_1 = s.m(2)*(s.l(1,2)*S2-s.l(3,2)*C2);
FB32_1 = s.l(2,2)*s.m(2)*C2;
CM22_1 = s.In(6,2)*S2-
s.l(1,2)*FB32_1+s.l(3,2)*FB12_1+CM13_1*S3+CM23_1*C3;
CM22_2 =
s.In(5,2)+s.l(1,2)*s.l(1,2)*s.m(2)+s.l(3,2)*s.l(3,2)*s.m(2)+C3*(s.In(2,3)
*S3+s.In(5,3)*C3+s.l(3,3)*FB13_2-FB33_2*s.l(1,3))+S3*(...
s.In(1,3)*S3+s.In(2,3)*C3+s.l(2,3)*FB33_2-s.l(3,3)*FB23_2);
CF11 = -(s.trq(1,1)+s.l(2,1)*(s.frc(3,1)-s.m(1)*(AlF31-
qd(1)*qd(1)*s.l(3,1)))-s.l(3,1)*(s.frc(2,1)-s.m(1)*(AlF21-
qd(1)*qd(1)*s.l(2,1)))+C2*(...
s.trq(1,2)-s.In(1,2)*OpF12-s.In(3,2)*OpF32-s.l(2,2)*FA32+s.l(3,2)*FA22-
CF13*C3+CF23*S3+OM32*(qd(2)*(s.In(5,2)-
s.In(9,2))+s.In(6,2)*OM32))+S2*(...
s.trq(3,2)-CF33+qd(2)*(qd(2)*s.In(2,2)+s.In(1,2)*OM12+s.In(3,2)*OM32)-
s.In(9,2)*OpF32-s.l(1,2)*FA22+s.l(2,2)*FA12-
OM12*(qd(2)*s.In(5,2)+s.In(6,2)*...
OM32)));
CM11_1 =
s.In(1,1)+s.l(2,1)*s.l(2,1)*s.m(1)+s.l(3,1)*s.l(3,1)*s.m(1)+C2*(s.In(1,2)
*C2+s.In(3,2)*S2+s.l(2,2)*FB32_1-s.l(3,2)*FB22_1+CM13_1*C3-...
CM23_1*S3)+S2*(CM33_1+s.In(9,2)*S2+s.l(1,2)*FB22_1-s.l(2,2)*FB12_1);

% == Block_0_3_0_0_0_0 ==

% Symbolic Outputs

M(1,1) = CM11_1;
M(1,2) = CM22_1;
M(1,3) = CM33_1;
M(2,1) = CM22_1;
M(2,2) = CM22_2;
M(2,3) = CM33_2;
M(3,1) = CM33_1;
M(3,2) = CM33_2;
M(3,3) = CM33_3;
c(1) = CF11;
c(2) = CF22;
c(3) = CF33;

iM = M\(-(c.));

```

## Modified Elbow Symbolic Function

```
function [iM] = dirdyna_elbow (q, qd, ...
    frcx1, frcy1, frcz1, ...
    trqx1, trqy1, trqz1)

global I2xx I2yy I2zz;
global X2 Z2;
global modeldata;

s.g = [0; 0; -9.8];
% mass of static model
ms = modeldata.m2+modeldata.m3; %kg
s.m = [0 ms];
% distance to CG
s.l = [0 X2; ...
       0 0; ...
       0 Z2];

s.In = [0 I2xx; ...
        0 0; ...
        0 0; ...
        0 0; ...
        0 I2yy; ...
        0 0; ...
        0 0; ...
        0 0; ...
        0 I2zz];

s.frc = [0 frcx1; ...
         0 frcy1; ...
         0 frcz1];

s.trq = [0 trqx1; ...
         0 trqy1; ...
         0 trqz1];

% Trigonometric Variables

C1 = cos(q(1));
S1 = sin(q(1));
C2 = cos(q(2));
S2 = sin(q(2));

% == Block_0_1_0_0_0_1 ==

% Forward Kinematics

AlF21 = -(s.g(2)*C1+s.g(3)*S1);
AlF31 = s.g(2)*S1-s.g(3)*C1;
OM12 = qd(1)*C2;
OM32 = qd(1)*S2;
```

```

OpF12 = -qd(1)*qd(2)*S2;
OpF32 = qd(1)*qd(2)*C2;
BS32 = OM12*OM32;

% == Block_0_2_0_1_0_1 ==

% Backward Dynamics

FA12 = -(s.frc(1,2)+s.m(2)*(s.g(1)*C2-
s.l(3,2)*BS32+AlF31*S2+s.l(1,2)*(qd(2)*qd(2)+OM32*OM32)));
FA22 = -(s.frc(2,2)-s.m(2)*(AlF21-qd(1)*qd(1)*s.l(2,2)-s.l(3,2)*(OpF12-
qd(2)*OM32)+s.l(1,2)*(OpF32+qd(2)*OM12)));
FA32 = -(s.frc(3,2)+s.m(2)*(s.g(1)*S2+s.l(3,2)*(qd(2)*qd(2)+OM12*OM12)-
AlF31*C2-BS32*s.l(1,2)));
CF22 = -(s.trq(2,2)-s.In(2,2)*OpF12-s.In(6,2)*OpF32-
s.l(3,2)*FA12+FA32*s.l(1,2)+OM12*(qd(2)*s.In(6,2)+s.In(3,2)*OM12+s.In(9,2)
)*OM32)-OM32*(...
qd(2)*s.In(2,2)+s.In(1,2)*OM12+s.In(3,2)*OM32));
FB12_1 = -s.l(2,2)*s.m(2)*S2;
FB22_1 = -s.m(2)*(s.l(3,2)*C2-s.l(1,2)*S2);
FB32_1 = s.l(2,2)*s.m(2)*C2;
CM22_1 = s.In(2,2)*C2+s.In(6,2)*S2+s.l(3,2)*FB12_1-FB32_1*s.l(1,2);
CM22_2 = s.In(5,2)+s.l(3,2)*s.l(3,2)*s.m(2)+s.m(2)*s.l(1,2)*s.l(1,2);
CF11 = -(s.trq(1,1)+s.l(2,1)*(s.frc(3,1)-s.m(1)*(AlF31-
qd(1)*qd(1)*s.l(3,1)))-s.l(3,1)*(s.frc(2,1)-s.m(1)*(AlF21-
qd(1)*qd(1)*s.l(2,1)))+C2*(...
s.trq(1,2)-qd(2)*(qd(2)*s.In(6,2)+s.In(3,2)*OM12+s.In(9,2)*OM32)-
s.In(1,2)*OpF12-s.In(3,2)*OpF32-
s.l(2,2)*FA32+s.l(3,2)*FA22+OM32*(qd(2)*s.In(5,2)+...

s.In(2,2)*OM12+s.In(6,2)*OM32))+S2*(s.trq(3,2)+qd(2)*(qd(2)*s.In(2,2)+s.I
n(1,2)*OM12+s.In(3,2)*OM32)-s.In(3,2)*OpF12-
s.In(9,2)*OpF32+s.l(2,2)*FA12-...
FA22*s.l(1,2)-OM12*(qd(2)*s.In(5,2)+s.In(2,2)*OM12+s.In(6,2)*OM32)));
CM11_1 =
s.In(1,1)+s.l(2,1)*s.l(2,1)*s.m(1)+s.l(3,1)*s.l(3,1)*s.m(1)+C2*(s.In(1,2)
*C2+s.In(3,2)*S2+s.l(2,2)*FB32_1-s.l(3,2)*FB22_1)+S2*(...
s.In(3,2)*C2+s.In(9,2)*S2-s.l(2,2)*FB12_1+FB22_1*s.l(1,2));

% == Block_0_3_0_0_0_0 ==

% Symbolic Outputs

M(1,1) = CM11_1;
M(1,2) = CM22_1;
M(2,1) = CM22_1;
M(2,2) = CM22_2;
c(1) = CF11;
c(2) = CF22;

% ===== END Task 0 =====

iM = M\(-(c.));

```



## Modified Wrist Symbolic Function

```
function [iM] = dirdyna_wrist (q, qd, ...
    frcx1, frcy1, frcz1, ...
    trqx1, trqy1, trqz1)

global I3xx I3yy I3zz;
global X3 Z3;
global modeldata;

s.g = [0; 0; -9.8];
% mass of static model
ms = modeldata.m3; %kg
s.m = [0 ms];
% distance to CG
s.l = [0 X3; ...
       0 0; ...
       0 Z3];

s.In = [0 I3xx; ...
        0 0; ...
        0 0; ...
        0 0; ...
        0 I3yy; ...
        0 0; ...
        0 0; ...
        0 0; ...
        0 I3zz];

s.frc = [0 frcx1; ...
         0 frcy1; ...
         0 frcz1];

s.trq = [0 trqx1; ...
         0 trqy1; ...
         0 trqz1];

C1 = cos(q(1));
S1 = sin(q(1));
C2 = cos(q(2));
S2 = sin(q(2));

% == Block_0_1_0_0_0_1 ==

% Forward Kinematics

AlF11 = -(s.g(1)*C1-s.g(3)*S1);
AlF31 = -(s.g(1)*S1+s.g(3)*C1);
OM12 = qd(1)*S2;
OM22 = qd(1)*C2;
OpF12 = qd(1)*qd(2)*C2;
OpF22 = -qd(1)*qd(2)*S2;
BS22 = OM12*OM22;
```

```
% == Block_0_2_0_1_0_1 ==
```

```
% Backward Dynamics
```

```
FA12 = -(s.frc(1,2)+s.m(2)*(s.g(2)*S2+s.l(1,2)*(qd(2)*qd(2)+OM22*OM22)-  
s.l(2,2)*BS22-AlF11*C2));  
FA22 = -(s.frc(2,2)+s.m(2)*(s.g(2)*C2-  
s.l(1,2)*BS22+s.l(2,2)*(qd(2)*qd(2)+OM12*OM12)+AlF11*S2));  
FA32 = -(s.frc(3,2)-s.m(2)*(AlF31-qd(1)*qd(1)*s.l(3,2)-s.l(1,2)*(OpF22-  
qd(2)*OM12)+s.l(2,2)*(OpF12+qd(2)*OM22)));  
CF32 = -(s.trq(3,2)-s.l(1,2)*FA22+s.l(2,2)*FA12-  
OM12*(qd(2)*s.In(6,2)+s.In(5,2)*OM22)+OM22*(qd(2)*s.In(3,2)+s.In(1,2)*OM1  
2+s.In(2,2)*OM22));  
FB12_1 = s.l(3,2)*s.m(2)*C2;  
FB22_1 = -s.l(3,2)*s.m(2)*S2;  
FB32_1 = -s.m(2)*(s.l(1,2)*C2-s.l(2,2)*S2);  
CM32_1 = s.l(1,2)*FB22_1-s.l(2,2)*FB12_1;  
CM32_2 = s.In(9,2)+s.l(1,2)*s.l(1,2)*s.m(2)+s.l(2,2)*s.l(2,2)*s.m(2);  
CF21 = -(s.trq(2,1)-s.l(1,1)*(s.frc(3,1)-s.m(1)*(AlF31-  
qd(1)*qd(1)*s.l(3,1)))+s.l(3,1)*(s.frc(1,1)-s.m(1)*(AlF11-  
qd(1)*qd(1)*s.l(1,1)))+C2*(...  
s.trq(2,2)-qd(2)*(qd(2)*s.In(3,2)+s.In(1,2)*OM12+s.In(2,2)*OM22-  
s.In(9,2)*OM12)-s.In(5,2)*OpF22+s.l(1,2)*FA32-  
s.l(3,2)*FA12)+S2*(s.trq(1,2)+qd(2)*(...  
qd(2)*s.In(6,2)+s.In(5,2)*OM22-s.In(9,2)*OM22)-s.In(1,2)*OpF12-  
s.In(2,2)*OpF22-s.l(2,2)*FA32+s.l(3,2)*FA22));  
CM21_1 =  
s.In(5,1)+s.l(1,1)*s.l(1,1)*s.m(1)+s.l(3,1)*s.l(3,1)*s.m(1)+C2*(s.In(5,2)  
*C2-s.l(1,2)*FB32_1+s.l(3,2)*FB12_1)+S2*(s.In(1,2)*S2+...  
s.In(2,2)*C2+s.l(2,2)*FB32_1-s.l(3,2)*FB22_1);
```

```
% == Block_0_3_0_0_0_0 ==
```

```
% Symbolic Outputs
```

```
M(1,1) = CM21_1;  
M(1,2) = CM32_1;  
M(2,1) = CM32_1;  
M(2,2) = CM32_2;  
c(1) = CF21;  
c(2) = CF32;
```

```
% ===== END Task 0 =====
```

```
iM = M\(-(c.));
```

## Appendix E

# DAE Functions for the Shoulder, Elbow and Wrist Complex

## Shoulder DAE Function

```
% OCP DAE function
function [dae] = ocpDaeShoulder(sol)

global dataMus;
global f1System f2System;
global L1 hx q1 q2 q3 q5;

t = sol.time;
x = sol.state;
u = sol.control;

f1n = ppval(f1System, t);
tf1x = 0.001*L1*cos(q2).*f1n;
tf1y = 0*t;
tf1z = (sin(q2)*0.001*L1+0.001*hx).*f1n;

f2n = ppval(f2System, t);
tf2x = sin(q3).*cos(q2).^2*0.001*L1.*f2n*cos(q5).*cos(q1)+sin(q2)*...
    0.001*L1.*f2n*cos(q5).*(-cos(q3).*sin(q1)+sin(q3).*sin(q2).*cos(q1));
tf2y = -
sin(q2)*0.001*L1.*f2n*cos(q5).*(sin(q3).*sin(q1)+cos(q3).*sin(q2).*...
    cos(q1))-cos(q3).*cos(q2).^2*0.001*L1.*f2n*cos(q5).*cos(q1);
tf2z = cos(q3).*cos(q2)*0.001*L1.*f2n*cos(q5).*(-
cos(q3).*sin(q1)+sin(q3).*...
    sin(q2).*cos(q1))-
sin(q3).*cos(q2)*0.001*L1.*f2n*cos(q5).*(sin(q3).*...
    sin(q1)+cos(q3).*sin(q2).*cos(q1));

x1dot = x(:,4);
x2dot = x(:,5);
x3dot = x(:,6);

x4dot = zeros(size(t, 1), 1);
x5dot = zeros(size(t, 1), 1);
x6dot = zeros(size(t, 1), 1);

for i = 1:size(t, 1)
    %% Muscle 1
    armx = 0.001*dataMus{1}.rxfunc(x(i,1), x(i,2), x(i,3));
    army = 0.001*dataMus{1}.ryfunc(x(i,1), x(i,2), x(i,3));
    armz = 0.001*dataMus{1}.rzfunc(x(i,1), x(i,2), x(i,3));
    musx = dataMus{1}.mxfunc(x(i,1), x(i,2), x(i,3));
    musy = dataMus{1}.myfunc(x(i,1), x(i,2), x(i,3));
    musz = dataMus{1}.mzfunc(x(i,1), x(i,2), x(i,3));
    F = 1500*u(i,1)*sin(5*t(i));
    tau1 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 2
    armx = 0.001*dataMus{2}.rxfunc(x(i,1), x(i,2), x(i,3));
    army = 0.001*dataMus{2}.ryfunc(x(i,1), x(i,2), x(i,3));
    armz = 0.001*dataMus{2}.rzfunc(x(i,1), x(i,2), x(i,3));
```

```

musx = dataMus{2}.mxfunc(x(i,1), x(i,2), x(i,3));
musy = dataMus{2}.myfunc(x(i,1), x(i,2), x(i,3));
musz = dataMus{2}.mzfunc(x(i,1), x(i,2), x(i,3));
F = 1500*u(i,2)*sin(5*t(i));
tau2 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
%% Muscle 3
armx = 0.001*dataMus{3}.rxfunc(x(i,1), x(i,2), x(i,3));
army = 0.001*dataMus{3}.ryfunc(x(i,1), x(i,2), x(i,3));
armz = 0.001*dataMus{3}.rzfunc(x(i,1), x(i,2), x(i,3));
musx = dataMus{3}.mxfunc(x(i,1), x(i,2), x(i,3));
musy = dataMus{3}.myfunc(x(i,1), x(i,2), x(i,3));
musz = dataMus{3}.mzfunc(x(i,1), x(i,2), x(i,3));
F = 1500*u(i,3)*sin(5*t(i));
tau3 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);

t1x = tau1(1)+tau2(1)+tau3(1)-tf1x(i)-tf2x(i);
t1y = tau1(2)+tau2(2)+tau3(2)-tf1y(i)-tf2y(i);
t1z = tau1(3)+tau2(3)+tau3(3)-tf1z(i)-tf2z(i);

f1x = 0;
f1y = 0;
f1z = 0;

iM = dirdyna_shoulder([x(i,1) x(i,2) x(i,3)], ...
    [x(i,4) x(i,5) x(i,6)], ...
    f1x,f1y,f1z, ...
    t1x,t1y,t1z);

x4dot(i) = iM(1);
x5dot(i) = iM(2);
x6dot(i) = iM(3);

end

dae = [x1dot x2dot x3dot x4dot x5dot x6dot];

end

```

## Elbow DAE Function

```
% OCP DAE function
function [dae] = ocpDaeElbow(sol)

global dataMus;
global f1System f2System;
global zo ax;
global q2;

t = sol.time;
x = sol.state;
u = sol.control;

f1n = ppval(f1System, t);
tf1x = 0.001*zo*f1n;
tf1y = 0*t;
tf1z = 0*t;

f2n = ppval(f2System, t);
tf2x = 0*t;
tf2y = -0.001*ax*f2n;
tf2z = 0*t;

x1dot = x(:,3);
x2dot = x(:,4);

x3dot = zeros(size(t, 1), 1);
x4dot = zeros(size(t, 1), 1);

for i = 1:size(t, 1)
    % Muscle 4
    armx = 0.001*dataMus{4}.rxfunc(x(i,1), x(i,2)-q2);
    army = 0.001*dataMus{4}.ryfunc(x(i,1), x(i,2)-q2);
    armz = 0.001*dataMus{4}.rzfunc(x(i,1), x(i,2)-q2);
    musx = dataMus{4}.mxfunc(x(i,1), x(i,2)-q2);
    musy = dataMus{4}.myfunc(x(i,1), x(i,2)-q2);
    musz = dataMus{4}.mzfunc(x(i,1), x(i,2)-q2);
    F = 500*u(i,1)*sin(5*t(i));
    tau4 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 5
    armx = 0.001*dataMus{5}.rxfunc(x(i,1), x(i,2)-q2);
    army = 0.001*dataMus{5}.ryfunc(x(i,1), x(i,2)-q2);
    armz = 0.001*dataMus{5}.rzfunc(x(i,1), x(i,2)-q2);
    musx = dataMus{5}.mxfunc(x(i,1), x(i,2)-q2);
    musy = dataMus{5}.myfunc(x(i,1), x(i,2)-q2);
    musz = dataMus{5}.mzfunc(x(i,1), x(i,2)-q2);
    F = 500*u(i,2)*sin(5*t(i));
    tau5 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 6
    armx = 0.001*dataMus{6}.rxfunc(x(i,1), x(i,2)-q2);
    army = 0.001*dataMus{6}.ryfunc(x(i,1), x(i,2)-q2);
```

```

armz = 0.001*dataMus{6}.rzfunc(x(i,1), x(i,2)-q2);
musx = dataMus{6}.mxfunc(x(i,1), x(i,2)-q2);
musy = dataMus{6}.myfunc(x(i,1), x(i,2)-q2);
musz = dataMus{6}.mzfunc(x(i,1), x(i,2)-q2);
F = 500*u(i,3)*sin(5*t(i));
tau6 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
%% Muscle 7
armx = 0.001*dataMus{7}.rxfunc(x(i,1), x(i,2)-q2);
army = 0.001*dataMus{7}.ryfunc(x(i,1), x(i,2)-q2);
armz = 0.001*dataMus{7}.rzfunc(x(i,1), x(i,2)-q2);
musx = dataMus{7}.mxfunc(x(i,1), x(i,2)-q2);
musy = dataMus{7}.myfunc(x(i,1), x(i,2)-q2);
musz = dataMus{7}.mzfunc(x(i,1), x(i,2)-q2);
F = 500*u(i,4)*sin(5*t(i));
tau7 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);

t2x = tau4(1)+tau5(1)+tau6(1)+tau7(1)-tf1x(i)-tf2x(i);
t2y = tau4(2)+tau5(2)+tau6(2)+tau7(2)-tf1y(i)-tf2y(i);
t2z = tau4(3)+tau5(3)+tau6(3)+tau7(3)-tf1z(i)-tf2z(i);

f1x = 0;
f1y = 0;
f1z = 0;

iM = dirdyna_elbow([x(i,1) x(i,2)], ...
    [x(i,3) x(i,4)], ...
    f1x,f1y,f1z, ...
    t2x,t2y,t2z);

x3dot(i) = iM(1);
x4dot(i) = iM(2);

end

dae = [x1dot x2dot x3dot x4dot];

end

```

## Wrist DAE Function

```
% OCP DAE function
function [dae] = ocpDaeWrist(sol)

global dataMus;
global flSystem;
global hx L2 q2 q5;

t = sol.time;
x = sol.state;
u = sol.control;

fln = ppval(flSystem, t);
tflx = 0*t;
tfly = 0*t;
tflz = (0.001*hx-0.001*L2)*fln;

x1dot = x(:,3);
x2dot = x(:,4);

x3dot = zeros(size(t, 1), 1);
x4dot = zeros(size(t, 1), 1);

for i = 1:size(t, 1)
    %% Muscle 8
    armx = 0.001*dataMus{8}.rxfunc(x(i,1), x(i,2)-q2-q5);
    army = 0.001*dataMus{8}.ryfunc(x(i,1), x(i,2)-q2-q5);
    armz = 0.001*dataMus{8}.rzfunc(x(i,1), x(i,2)-q2-q5);
    musx = dataMus{8}.mxfunc(x(i,1), x(i,2)-q2-q5);
    musy = dataMus{8}.myfunc(x(i,1), x(i,2)-q2-q5);
    musz = dataMus{8}.mzfunc(x(i,1), x(i,2)-q2-q5);
    F = 500*u(i,1)*sin(5*t(i));
    tau8 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 9
    armx = 0.001*dataMus{9}.rxfunc(x(i,1), x(i,2)-q2-q5);
    army = 0.001*dataMus{9}.ryfunc(x(i,1), x(i,2)-q2-q5);
    armz = 0.001*dataMus{9}.rzfunc(x(i,1), x(i,2)-q2-q5);
    musx = dataMus{9}.mxfunc(x(i,1), x(i,2)-q2-q5);
    musy = dataMus{9}.myfunc(x(i,1), x(i,2)-q2-q5);
    musz = dataMus{9}.mzfunc(x(i,1), x(i,2)-q2-q5);
    F = 500*u(i,2)*sin(5*t(i));
    tau9 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 10
    armx = 0.001*dataMus{10}.rxfunc(x(i,1), x(i,2)-q2-q5);
    army = 0.001*dataMus{10}.ryfunc(x(i,1), x(i,2)-q2-q5);
    armz = 0.001*dataMus{10}.rzfunc(x(i,1), x(i,2)-q2-q5);
    musx = dataMus{10}.mxfunc(x(i,1), x(i,2)-q2-q5);
    musy = dataMus{10}.myfunc(x(i,1), x(i,2)-q2-q5);
    musz = dataMus{10}.mzfunc(x(i,1), x(i,2)-q2-q5);
    F = 500*u(i,3)*sin(5*t(i));
    tau10 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);
    %% Muscle 11
```



```

armx = 0.001*dataMus{11}.rxfunc(x(i,1), x(i,2)-q2-q5);
army = 0.001*dataMus{11}.ryfunc(x(i,1), x(i,2)-q2-q5);
armz = 0.001*dataMus{11}.rzfunc(x(i,1), x(i,2)-q2-q5);
musx = dataMus{11}.mxfunc(x(i,1), x(i,2)-q2-q5);
musy = dataMus{11}.myfunc(x(i,1), x(i,2)-q2-q5);
musz = dataMus{11}.mzfunc(x(i,1), x(i,2)-q2-q5);
F = 500*u(i,4)*sin(5*t(i));
tau11 = cross([armx'; army'; armz'], [F*musx'; F*musy'; F*musz']);

t1x = tau8(1)+tau9(1)+tau10(1)+tau11(1)-tf1x(i);
t1y = tau8(2)+tau9(2)+tau10(2)+tau11(2)-tf1y(i);
t1z = tau8(3)+tau9(3)+tau10(3)+tau11(3)-tf1z(i);

f1x = 0;
f1y = 0;
f1z = 0;

iM = dirdyna_wrist([x(i,1) x(i,2)], ...
    [x(i,3) x(i,4)], ...
    f1x,f1y,f1z, ...
    t1x,t1y,t1z);

x3dot(i) = iM(1);
x4dot(i) = iM(2);

end

dae = [x1dot x2dot x3dot x4dot];

end

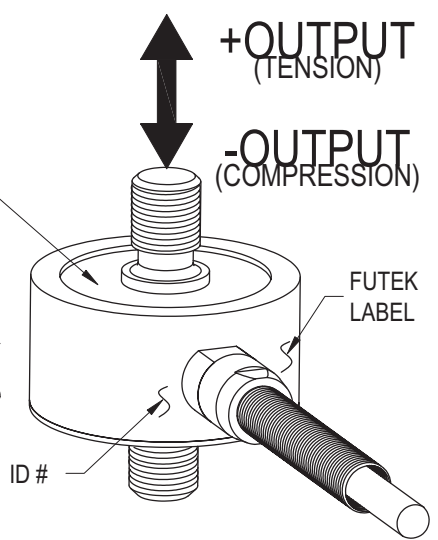
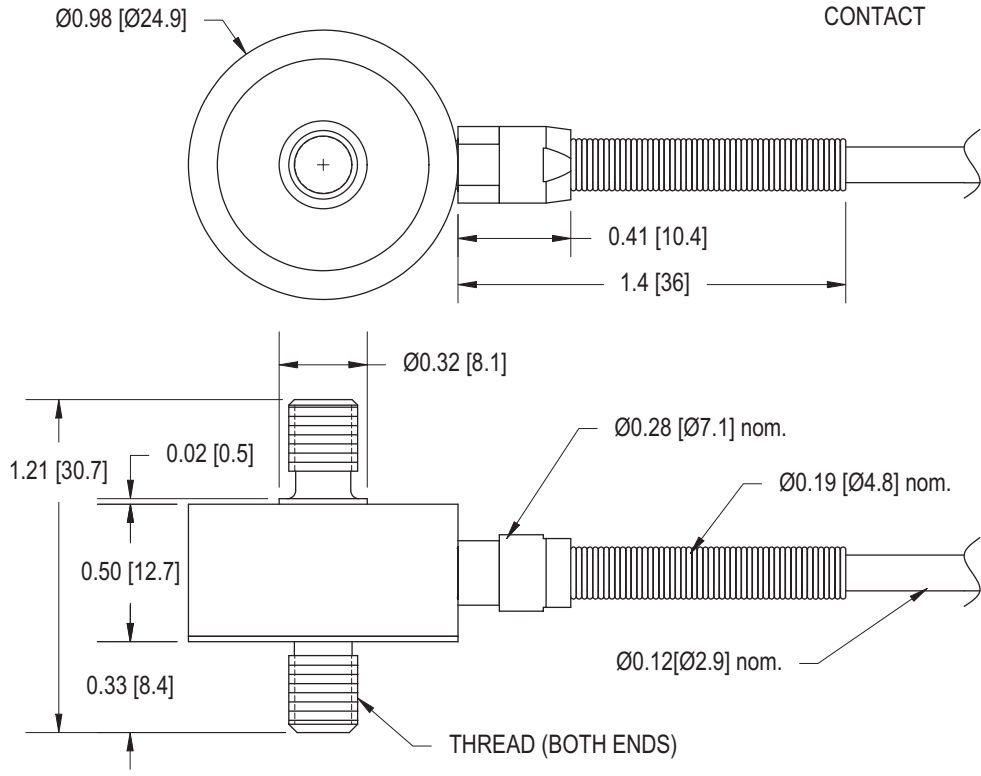
```

## Appendix F

# Datasheet of the Integrated Force-Gauges

# FUTEK MODEL LCM300 (L1650) *Tension and Compression Load Cell (Miniature/Inline Threaded)* (Improved Design)

<b>Drawing Number: FI1059-A</b>			
<b>INCH [mm]</b>		<b>R.O.= Rated Output</b>	
<b>WIRING CODE (WC1)</b>			
+Excitation	-Excitation	+Signal	-Signal
RED	BLACK	GREEN	WHITE
Shield			
FLOATING			

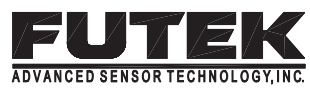


STK #	CAPACITIES		THREAD
	lb	N	
FSH02629	25	111	1/4-28-2A
FSH02698			M6X1
FSH02630	50	223	1/4-28-2A
FSH02699			M6X1
FSH02631	100	445	1/4-28-2A
FSH02700			M6X1
FSH02632	250	1112	1/4-28-2A
FSH02701			M6X1
FSH02633	500	2224	1/4-28-2A
FSH02702			M6X1
FSH02634	1000	4448	1/4-28-2A
FSH02703			M6X1

### SPECIFICATIONS:

<b>RATED OUTPUT</b>	2 mV/V nom.
<b>SAFE OVERLOAD</b>	150% of R.O.
<b>ZERO BALANCE</b>	±3% of R.O.
<b>EXCITATION (VDC OR VAC)</b>	15 MAX
<b>BRIDGE RESISTANCE</b>	700 Ω nom.
<b>NONLINEARITY</b>	±0.5% of R.O.
<b>HYSTERESIS</b>	±0.5% of R.O.
<b>NONREPEATABILITY</b>	±0.1% of R.O.
<b>TEMP. SHIFT ZERO</b>	±0.005% of R.O./°F [0.01% of R.O./°C]
<b>TEMP. SHIFT SPAN</b>	±0.02% of LOAD/°F [0.036% of LOAD/°C]
<b>COMPENSATED TEMP.</b>	60 to 160°F [15 to 72°C]
<b>OPERATING TEMP.</b>	-45 to 200°F [-42 to 93°C]
<b>WEIGHT</b>	2oz. [57g] (1.2oz [33g]- 25 lb Capacity)
<b>MATERIAL (FLEXURE)</b>	17-4PH S.S. (Aluminum- 25 lb Capacity)
<b>DEFLECTION</b>	0.001 to 0.002 [0.03 to 0.05] nom.
<b>CABLE: #28 AWG, 4 Conductor, Spiral Shielded PVC Cable 10 ft [3 m] Long</b>	
<b>ACCESSORIES AND RELATED INSTRUMENTS AVAILABLE</b>	
<b>CALIBRATION (STD)</b>	5 pt TENSION; 100 K Ω SHUNT CAL. VALUE
<b>CALIBRATION (AVAILABLE)</b>	COMPRESSION
<b>CALIBRATION TEST EXCITATION</b>	10 VDC

STOCK #'S IN THE TABLE HAS REPLACED: 25lb-FSH00665, 50lb-FSH00666, 100lb-FSH00667, 250lb-FSH00668, 500lb-FSH00669, 1Klb-FSH00670



This drawing is submitted solely for the information and exclusive use of the original addressee. It is not to be divulged in whole or in part, by any firm or individual without written permission from FUTEK

10 THOMAS  
IRVINE, CA 92618 USA  
1-800-23-FUTEK (38835)

INTERNET:  
<http://www.futek.com>

## Appendix G

# Datasheets of the Data-Acquisition System



**Technical Sales**  
Australia  
1 800 300 800  
info.australia@ni.com

## NI USB-6210

### 16-Bit, 250 kS/s M Series Multifunction DAQ, Bus-Powered

- 16 analog inputs (16-bit, 250 kS/s)
- 4 digital inputs; 4 digital outputs; two 32-bit counters
- Bus-powered USB for high mobility; built-in signal connectivity
- NI signal streaming for sustained high-speed data streams over USB
- Compatibility with LabVIEW, LabWindows™/CVI, and Measurement Studio for Visual Studio .NET
- NI-DAQmx driver software and LabVIEW SignalExpress LE interactive data-logging software



## Overview

---

The NI USB-6210 is a bus-powered M Series multifunction data acquisition (DAQ) module for USB that is optimized for superior accuracy at fast sampling rates. It offers 16 analog inputs; a 250 kS/s single-channel sampling rate; four digital input lines; four digital output lines; four programmable input ranges ( $\pm 0.2$  to  $\pm 10$  V) per channel; digital triggering; and two counter/timers.

The USB-6210 is designed specifically for mobile or space-constrained applications. Plug-and-play installation minimizes configuration and setup time, while direct screw-terminal connectivity keeps costs down and simplifies signal connections. This product does not require external power.

This module also features the new NI signal streaming technology, which gives you DMA-like bidirectional high-speed streaming of data across USB. For more information about NI signal streaming, view the Resources tab.

Bus-powered M Series devices for USB are available in OEM versions. Check the Resources tab or use the left navigation to get pricing and technical information.

### Driver Software

NI-DAQmx driver and measurement services software provides easy-to-use configuration and programming interfaces with features such as the DAQ Assistant to help reduce development time. Browse the information in the Resources tab to learn more about driver software or download a driver. M Series devices are not compatible with the Traditional NI-DAQ (Legacy) driver.

### Application Software

Every M Series DAQ device includes a copy of NI LabVIEW SignalExpress LE data-logging software, so you can quickly acquire, analyze, and present data without programming. In addition to LabVIEW SignalExpress, M Series DAQ devices are compatible with the following versions (or later) of NI application software: LabVIEW 7.1, LabWindows/CVI 7.x, or Measurement Studio 7.x. M Series DAQ devices are also compatible with Visual Studio .NET, C/C++, and Visual Basic 6.0.

The mark LabWindows is used under a license from Microsoft Corporation. Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

# Specifications

---

## Specifications Documents

- Specifications
- Data Sheet

## Specifications Summary

### General

Product Name	USB-6210
Product Family	Multifunction Data Acquisition
Form Factor	USB
Operating System/Target	Windows , Linux , Mac OS
DAQ Product Family	M Series
Measurement Type	Voltage
RoHS Compliant	Yes

### Analog Input

Channels	16 , 8
Single-Ended Channels	16
Differential Channels	8
Resolution	16 bits
Sample Rate	250 kS/s
Max Voltage	10 V
Maximum Voltage Range	-10 V , 10 V
Maximum Voltage Range Accuracy	2.69 mV
Maximum Voltage Range Sensitivity	91.6 $\mu$ V
Minimum Voltage Range	-200 mV , 200 mV
Minimum Voltage Range Accuracy	0.088 mV
Minimum Voltage Range Sensitivity	4.8 $\mu$ V
Number of Ranges	4
Simultaneous Sampling	No
On-Board Memory	4095 samples

### Analog Output

Channels	0
----------	---

### Digital I/O

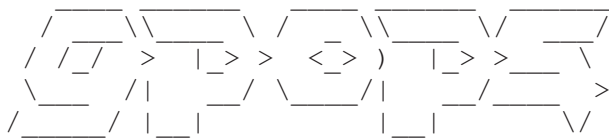
<b>Bidirectional Channels</b>	0
<b>Input-Only Channels</b>	4
<b>Output-Only Channels</b>	4
<b>Number of Channels</b>	0 , 4 , 4
<b>Timing</b>	Software
<b>Logic Levels</b>	TTL
<b>Input Current Flow</b>	Sinking
<b>Output Current Flow</b>	Sourcing
<b>Programmable Input Filters</b>	No
<b>Supports Programmable Power-Up States?</b>	Yes
<b>Current Drive Single</b>	16 mA
<b>Current Drive All</b>	50 mA
<b>Watchdog Timer</b>	No
<b>Supports Handshaking I/O?</b>	No
<b>Supports Pattern I/O?</b>	No
<b>Maximum Input Range</b>	0 V , 5.25 V
<b>Maximum Output Range</b>	0 V , 3.8 V
<b>Counter/Timers</b>	
<b>Counters</b>	2
<b>Buffered Operations</b>	Yes
<b>Debouncing/Glitch Removal</b>	Yes
<b>GPS Synchronization</b>	No
<b>Maximum Range</b>	0 V , 5.25 V
<b>Max Source Frequency</b>	80 MHz
<b>Pulse Generation</b>	Yes
<b>Resolution</b>	32 bits
<b>Timebase Stability</b>	50 ppm
<b>Logic Levels</b>	TTL
<b>Physical Specifications</b>	
<b>Length</b>	16.9 cm
<b>Width</b>	9.4 cm

## Appendix H

# OCSP Convergence Statistics



**Experiment #1 OCSF Stats**



-----  
 GPOPS Version 2.2 beta: A MATLAB Implementation of the Gauss Pseudospectral Method  
 Copyright (c) 2008 Anil V. Rao, David Benson, Geoffrey T. Huntington,  
 Christopher L. Darby, Michael Patterson, and Camila Francolin  
 -----

-----  
 -----  
 Downloading, using, copying, or modifying the GPOPS code constitutes an agreement to  
 ALL of  
 the terms of the GPOPS license. Please see the file LICENSE given in the home  
 directory of  
 the GPOPS distribution or see the summary file printed when running an example.  
 -----  
 -----

-----  
 Summary of Problem Written to File: Shoulder OCP.txt  
 -----

Using Default Sparsity  
 Automatic Scaling Turned Off  
 Objective Gradient Being Estimated via Finite Differencing  
 Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	60	Linear constraints	7
Nonlinear variables	104	Linear variables	0
Jacobian variables	92	Objective variables	104
Total constraints	67	Total variables	104

The user has defined 0 out of 764 first derivatives

	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	-4.1E+00	27	3.0E+04					
	200	-3.2E+07	28	2.6E+04		2.6716911E+08			
Major Minors		Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	236		1	2.9E+05	7.0E-01	1.3042813E+11			r i c
Major Minors		Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	236		1	2.9E+05	7.0E-01	1.3042813E+11			r i c
1	7	1.1E-01	2	2.1E+05	2.4E+00	1.1603757E+11			n r l i c
2	50	1.2E-01	4	1.5E+05	1.1E-01	1.0263806E+11	1		s M i c
	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	-8.0E-06	35	1.7E+04					

SNOPTA EXIT 10 -- the problem appears to be infeasible  
 SNOPTA INFO 15 -- infeasible linear constraints in QP subproblem

Problem name  
 No. of iterations 396 Objective value 9.5478550546E-02  
 No. of infeasibilities 32 Sum of infeas 1.5646036559E+04  
 Elastic weight 1.0E+04 Scaled Merit 9.9870648839E+06  
 No. of major iterations 3 Linear objective 0.0000000000E+00

Penalty parameter	0.000E+00	Nonlinear objective	9.5478550546E-02
No. of calls to funobj	1373	No. of calls to funcon	1373
Calls with modes 1,2 (known g)	6	Calls with modes 1,2 (known g)	6
Calls for forward differencing	104	Calls for forward differencing	104
Calls for central differencing	1248	Calls for central differencing	1248
No. of degenerate steps	14	Percentage	3.54
Max x	31 3.1E+00	Max pi	51 3.6E-07
Max Primal infeas	141 1.0E+05	Max Dual infeas	14 1.0E+00
Nonlinear constraint violn	7.9E+05		

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	30.02 seconds
Time for solution output	.00 seconds
Time for constraint functions	32.72 seconds
Time for objective function	.00 seconds

-----  
BEGIN: Computation of Endpoint Controls in Phase 1  
-----

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	3	Linear variables	0
Jacobian variables	0	Objective variables	3
Total constraints	1	Total variables	3

The user has defined 0 out of 3 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	1.4133240E+11		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	1.4133240E+11		r	c

SNOPTA EXIT 0 -- finished successfully  
SNOPTA INFO 1 -- optimality conditions satisfied

Problem name

No. of iterations	0	Objective value	1.4133240453E+11
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	1.4133240453E+11
No. of calls to funobj	10	No. of calls to funcon	10
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	3	Calls for forward differencing	3
Calls for central differencing	6	Calls for central differencing	6
No. of degenerate steps	0	Percentage	.00
Max x	2 7.5E-04	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	0 0.0E+00

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.03 seconds
Time for solution output	.00 seconds
Time for constraint functions	.08 seconds
Time for objective function	.00 seconds

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	3	Linear variables	0
Jacobian variables	0	Objective variables	3
Total constraints	1	Total variables	3

The user has defined 0 out of 3 first derivatives

Major	Minors	Step	nObj	Feasible	Optimal	Objective	nS	
0	2		1		2.0E-02	-1.4743659E-02	1	r
1	1	9.0E-01	2		2.2E-03	-7.2971877E-02		n rl
2	0	1.0E+00	3		(0.0E+00)	-8.1257577E-02		s c
2	0	1.0E+00	3		(0.0E+00)	-8.1257577E-02		s c

SNOPTA EXIT 0 -- finished successfully  
 SNOPTA INFO 1 -- optimality conditions satisfied

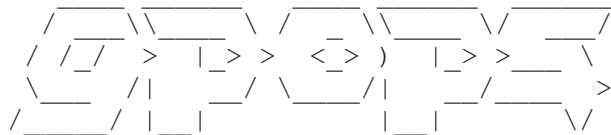
Problem name

No. of iterations	3	Objective value	-8.1257576582E-02
No. of major iterations	2	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	-8.1257576582E-02
No. of calls to funobj	22	No. of calls to funcon	22
Calls with modes 1,2 (known g)	3	Calls with modes 1,2 (known g)	3
Calls for forward differencing	9	Calls for forward differencing	9
Calls for central differencing	6	Calls for central differencing	6
No. of degenerate steps	0	Percentage	.00
Max x	1 1.0E+00	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	0 0.0E+00

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.08 seconds
Time for solution output	.00 seconds
Time for constraint functions	.11 seconds
Time for objective function	.00 seconds

-----  
 END: Computation of Endpoint Controls in Phase 1  
 -----



-----  
 GPOPS Version 2.2 beta: A MATLAB Implementation of the Gauss Pseudospectral Method  
 Copyright (c) 2008 Anil V. Rao, David Benson, Geoffrey T. Huntington,  
 Christopher L. Darby, Michael Patterson, and Camila Francolin  
 -----

-----  
 Downloading, using, copying, or modifying the GPOPS code constitutes an agreement to  
 ALL of

the terms of the GPOPS license. Please see the file LICENSE given in the home directory of the GPOPS distribution or see the summary file printed when running an example.

-----  
 Summary of Problem Written to File: Elbow OCP.txt  
 -----

Using Default Sparsity  
 Automatic Scaling Turned Off  
 Objective Gradient Being Estimated via Finite Differencing  
 Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	40	Linear constraints	5
Nonlinear variables	90	Linear variables	0
Jacobian variables	82	Objective variables	90
Total constraints	45	Total variables	90

The user has defined 0 out of 490 first derivatives

	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	-4.0E-01	9	9.1E+01					
Major	Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	191		1	9.5E+00	1.3E-01	2.3296796E+06			r i
1	3	3.1E-01	2	6.7E+00	9.8E-02	1.9285613E+06	1		n rli
2	9	4.5E-01	4	5.4E+00	6.3E-02	1.5144136E+06			sM li
3	1	1.0E+00	6	4.6E+00	1.4E-02	9.9365107E+05	1		M i
4	1	1.0E+00	7	4.6E+00	2.7E-04	9.9251036E+05	1		i
5	1	1.0E+00	8	4.6E+00	1.8E-05	9.9251006E+05	1		i
6	4	1.0E+00	9	4.6E+00	(2.3E-09)	9.9251005E+11	1		i c
6	5	1.0E+00	9	4.6E+00	(9.6E-09)	9.9251005E+11	1		i c

SNOPTA EXIT 10 -- the problem appears to be infeasible

SNOPTA INFO 13 -- nonlinear infeasibilities minimized

Problem name

No. of iterations	211	Objective value	8.6600783855E-03
No. of infeasibilities	14	Sum of infeas	9.9250996133E+01
Elastic weight	1.0E+10	Scaled Merit	9.9250996133E+01
No. of major iterations	6	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	8.6600783855E-03
No. of calls to funobj	1009	No. of calls to funcon	1009
Calls with modes 1,2 (known g)	9	Calls with modes 1,2 (known g)	9
Calls for forward differencing	810	Calls for forward differencing	810
Calls for central differencing	180	Calls for central differencing	180
No. of superbasics	1	No. of basic nonlinears	31
No. of degenerate steps	19	Percentage	9.00
Max x	33 3.1E+00	Max pi	42 1.5E+11
Max Primal infeas	130 2.4E+01	Max Dual infeas	3 5.9E+03
Nonlinear constraint violn	2.4E+01		

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	25.58 seconds
Time for solution output	.01 seconds

Time for constraint functions 28.23 seconds  
 Time for objective function .00 seconds

-----  
 BEGIN: Computation of Endpoint Controls in Phase 1  
 -----

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4
Total constraints	1	Total variables	4

The user has defined 0 out of 4 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	3.4295840E+12		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	4	1		1.8E+00	3.4295840E+12		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	4	1		1.8E+00	3.4295840E+12		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	4	1		1.8E+00	3.4295840E+12		r	c

SNOPTA EXIT 40 -- terminated after numerical difficulties

SNOPTA INFO 41 -- current point cannot be improved

Problem name

No. of iterations	4	Objective value	3.4295840437E+12
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	3.4295840437E+12
No. of calls to funobj	22	No. of calls to funcon	22
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	4	Calls for forward differencing	4
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	1 0.0E+00	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	1 3.6E+00

Solution printed on file 9

Time for MPS input .00 seconds  
 Time for solving problem .08 seconds  
 Time for solution output .00 seconds  
 Time for constraint functions .13 seconds  
 Time for objective function .00 seconds

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4
Total constraints	1	Total variables	4

The user has defined 0 out of 4 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	2	1		7.0E+08	2.0428754E+12		r	



Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	40	Linear constraints	5
Nonlinear variables	90	Linear variables	0
Jacobian variables	82	Objective variables	90
Total constraints	45	Total variables	90

The user has defined 0 out of 490 first derivatives

	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	4.5E+00	15	3.7E+02					
	200	1.7E+00	19	8.3E+02					
	300	6.4E+04	16	9.2E+02			1.1069978E+07		
Major Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty		
0	321	1	1.2E+02	1.1E+01	2.6630116E+07				r i
1	7	1.7E-01	2	8.9E+01	1.0E+01	3			n r l i
2	24	2.1E-01	3	7.1E+01	4.2E+00	9			s l i
3	15	1.0E+00	4	3.9E+01	3.0E+01	9			i
4	16	1.0E+00	6	3.9E+01	1.7E+00	9			m i
5	18	1.0E+00	7	3.7E+01	1.1E+00	14			i
6	6	1.0E+00	8	3.8E+01	1.8E-01	13			i
7	13	1.0E+00	10	3.9E+01	5.2E+00	11			m i
8	5	1.0E+00	12	4.0E+01	1.3E-01	13			m i
9	4	1.0E+00	14	4.0E+01	4.7E-02	12			m i
Major Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty		
10	4	1.0E+00	16	4.0E+01	4.9E-02	11			m i
11	2	1.0E+00	18	3.9E+01	1.2E-01	10			m i
12	1	1.0E+00	20	3.9E+01	1.1E-01	10			n i
13	1	1.0E+00	22	3.9E+01	1.2E-02	10			n R i
14	1	1.0E+00	24	3.9E+01	1.2E-02	10			n R i
15	1	1.0E+00	26	3.9E+01	1.6E-02	9			n r i
16	1	1.0E+00	28	3.8E+01	(1.9E-11)	9			sm i c
16	2	1.0E+00	28	3.8E+01	(1.7E-11)	9			sm i c
16	3	1.0E+00	28	3.8E+01	(1.2E-11)	9			smr i c
17	1	4.7E-04	29	3.8E+01	(7.6E-08)	9	1.5E-06		s i c
18	1	1.4E-02	30	3.8E+01	(9.0E-09)	9	4.2E-06		i c
18	2	1.0E+00	30	3.8E+01	(2.5E-11)	9			r i c

SNOPTA EXIT 40 -- terminated after numerical difficulties

SNOPTA INFO 41 -- current point cannot be improved

Problem name

No. of iterations	445	Objective value	3.6600919396E-01
No. of infeasibilities	19	Sum of infeas	9.2121073000E+02
Elastic weight	1.2E+04	Scaled Merit	9.2121076041E+02
No. of major iterations	18	Linear objective	0.0000000000E+00
Penalty parameter	4.676E-06	Nonlinear objective	3.6600919396E-01
No. of calls to funobj	3143	No. of calls to funcon	3143
Calls with modes 1,2 (known g)	30	Calls with modes 1,2 (known g)	30
Calls for forward differencing	2520	Calls for forward differencing	2520
Calls for central differencing	540	Calls for central differencing	540
No. of superbasics	9	No. of basic nonlinears	25
No. of degenerate steps	17	Percentage	3.82
Max x	28 3.1E+00	Max pi	42 1.8E+04
Max Primal infeas	126 3.2E+02	Max Dual infeas	9 9.9E-02
Nonlinear constraint violn	3.2E+02		

Solution printed on file 9

```

Time for MPS input          .00 seconds
Time for solving problem    84.27 seconds
Time for solution output    .00 seconds
Time for constraint functions 86.87 seconds
Time for objective function .00 seconds

```

-----  
BEGIN: Computation of Endpoint Controls in Phase 1  
-----

```

Nonlinear constraints      0      Linear constraints      1
Nonlinear variables       4      Linear variables       0
Jacobian variables        0      Objective variables    4
Total constraints          1      Total variables        4

```

The user has defined 0 out of 4 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	4.6124463E+07		r	c
0	4	1		2.8E-05	4.6124463E+07	4	r	c
0	4	1		2.8E-05	4.6124463E+07	4	r	c
0	4	1		2.8E-05	4.6124463E+07	4	r	c

SNOPTA EXIT 40 -- terminated after numerical difficulties

SNOPTA INFO 41 -- current point cannot be improved

Problem name

No. of iterations	4	Objective value	4.6124463326E+07
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	4.6124463326E+07
No. of calls to funobj	22	No. of calls to funcon	22
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	4	Calls for forward differencing	4
Calls for central differencing	8	Calls for central differencing	8
No. of superbasics	4	No. of basic nonlinears	0
No. of degenerate steps	0	Percentage	.00
Max x	2 5.3E-06	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	1 5.6E-05

Solution printed on file 9

```

Time for MPS input          .00 seconds
Time for solving problem    .08 seconds
Time for solution output    .00 seconds
Time for constraint functions .13 seconds
Time for objective function .00 seconds

```

```

Nonlinear constraints      0      Linear constraints      1
Nonlinear variables       4      Linear variables       0
Jacobian variables        0      Objective variables    4
Total constraints          1      Total variables        4

```



The user has defined 0 out of 4 first derivatives

Major	Minors	Step	nObj	Feasible	Optimal	Objective	nS			
0	4		1		2.5E+05	2.0069900E+05				r
1	0	1.0E+00	2		(0.0E+00)	-1.0002744E+06		n	r	c
1	0	1.0E+00	2		(0.0E+00)	-1.0002744E+06		n	r	c

SNOPTA EXIT 0 -- finished successfully  
SNOPTA INFO 1 -- optimality conditions satisfied

Problem name

No. of iterations	4	Objective value	-1.0002743832E+06
No. of major iterations	1	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	-1.0002743832E+06
No. of calls to funobj	21	No. of calls to funcon	21
Calls with modes 1,2 (known g)	2	Calls with modes 1,2 (known g)	2
Calls for forward differencing	8	Calls for forward differencing	8
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	3 5.0E-13	Max pi	1 1.0E+00
Max Primal infeas	3 5.0E-13	Max Dual infeas	0 0.0E+00

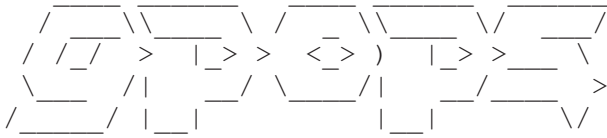
Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.06 seconds
Time for solution output	.00 seconds
Time for constraint functions	.11 seconds
Time for objective function	.00 seconds

-----  
END: Computation of Endpoint Controls in Phase 1  
-----

Total Solution Time (min): 2.5151

## Experiment #2 OCSF Stats



-----  
 GPOPS Version 2.2 beta: A MATLAB Implementation of the Gauss Pseudospectral Method  
 Copyright (c) 2008 Anil V. Rao, David Benson, Geoffrey T. Huntington,  
 Christopher L. Darby, Michael Patterson, and Camila Francolin  
 -----

-----  
 -----  
 Downloading, using, copying, or modifying the GPOPS code constitutes an agreement to  
 ALL of  
 the terms of the GPOPS license. Please see the file LICENSE given in the home  
 directory of  
 the GPOPS distribution or see the summary file printed when running an example.  
 -----  
 -----

-----  
 Summary of Problem Written to File: Shoulder OCP.txt  
 -----

Using Default Sparsity  
 Automatic Scaling Turned Off  
 Objective Gradient Being Estimated via Finite Differencing  
 Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	60	Linear constraints	7
Nonlinear variables	104	Linear variables	0
Jacobian variables	92	Objective variables	104
Total constraints	67	Total variables	104

The user has defined 0 out of 764 first derivatives

Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
100	-1.9E+01	26	4.2E+04					
200	-5.1E+04	30	3.4E+04		3.5071387E+08			
Major Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	239	1	3.1E+05	1.3E-01	1.4575098E+11			r i c
Major Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	239	1	3.1E+05	1.3E-01	1.4575098E+11			r i c
1	8	9.7E-02	2	2.2E+05	1.9E-01	1.3158754E+11		n r l i c
2	79	1.1E-01	4	1.7E+05	6.4E-02	1.1744605E+11	31	sM l i c

SNOPTA EXIT 10 -- the problem appears to be infeasible

SNOPTA INFO 15 -- infeasible linear constraints in QP subproblem

Problem name

No. of iterations	422	Objective value	1.1742636400E-01
No. of infeasibilities	29	Sum of infeas	2.5398053174E+04
Elastic weight	1.0E+04	Scaled Merit	1.1330018289E+07
No. of major iterations	3	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	1.1742636400E-01
No. of calls to funobj	1365	No. of calls to funcon	1365

Calls with modes 1,2 (known g)	6	Calls with modes 1,2 (known g)	6
Calls for forward differencing	104	Calls for forward differencing	104
Calls for central differencing	1248	Calls for central differencing	1248
No. of degenerate steps	11	Percentage	2.61
Max x	16 1.7E+00	Max pi	41 4.1E-07
Max Primal infeas	144 1.5E+05	Max Dual infeas	14 1.0E+00
Nonlinear constraint violn	4.2E+05		

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	37.39 seconds
Time for solution output	.00 seconds
Time for constraint functions	40.27 seconds
Time for objective function	.00 seconds

-----  
BEGIN: Computation of Endpoint Controls in Phase 1  
-----

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	3	Linear variables	0
Jacobian variables	0	Objective variables	3
Total constraints	1	Total variables	3

The user has defined 0 out of 3 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0 0		1		(0.0E+00)	1.0482169E+11		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0 3		1		5.7E-02	1.0482169E+11	3	r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0 3		1		5.7E-02	1.0482169E+11	3	r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0 3		1		5.7E-02	1.0482169E+11	3	r	c

SNOPTA EXIT 40 -- terminated after numerical difficulties

SNOPTA INFO 41 -- current point cannot be improved

Problem name

No. of iterations	3	Objective value	1.0482169475E+11
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	1.0482169475E+11
No. of calls to funobj	19	No. of calls to funcon	19
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	3	Calls for forward differencing	3
Calls for central differencing	6	Calls for central differencing	6
No. of superbasics	3	No. of basic nonlinears	0
No. of degenerate steps	0	Percentage	.00
Max x	1 1.0E+00	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	2 1.1E-01

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.08 seconds

```

Time for solution output          .00 seconds
Time for constraint functions      .13 seconds
Time for objective function       .00 seconds

```

```

Nonlinear constraints      0      Linear constraints      1
Nonlinear variables       3      Linear variables      0
Jacobian variables        0      Objective variables   3
Total constraints          1      Total variables       3

```

The user has defined 0 out of 3 first derivatives

Major	Minors	Step	nObj	Feasible	Optimal	Objective	nS
0	3		1		8.5E-03	4.5760406E-03	1 r
1	1	6.9E-01	2		2.7E-03	-1.8243365E-02	n rl
2	0	1.0E+00	3		(0.0E+00)	-3.0754645E-02	s c
2	0	1.0E+00	3		(0.0E+00)	-3.0754645E-02	s c

```

SNOPTA EXIT  0 -- finished successfully
SNOPTA INFO  1 -- optimality conditions satisfied

```

```

Problem name
No. of iterations          4      Objective value      -3.0754645174E-02
No. of major iterations    2      Linear objective     0.0000000000E+00
Penalty parameter         0.000E+00      Nonlinear objective  -3.0754645174E-02
No. of calls to funobj     22      No. of calls to funcon  22
Calls with modes 1,2 (known g)  3      Calls with modes 1,2 (known g)  3
Calls for forward differencing  9      Calls for forward differencing  9
Calls for central differencing  6      Calls for central differencing  6
No. of degenerate steps    0      Percentage           .00
Max x                     1 1.0E+00      Max pi                1 1.0E+00
Max Primal infeas        0 0.0E+00      Max Dual infeas       0 0.0E+00

```

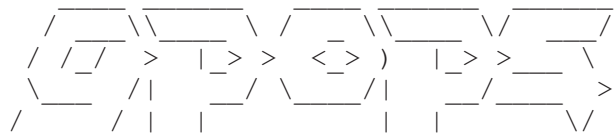
Solution printed on file 9

```

Time for MPS input          .00 seconds
Time for solving problem    .08 seconds
Time for solution output    .00 seconds
Time for constraint functions .11 seconds
Time for objective function .00 seconds

```

-----  
END: Computation of Endpoint Controls in Phase 1  
-----



-----  
GPOPS Version 2.2 beta: A MATLAB Implementation of the Gauss Pseudospectral Method  
Copyright (c) 2008 Anil V. Rao, David Benson, Geoffrey T. Huntington,  
Christopher L. Darby, Michael Patterson, and Camila Francolin  
-----

-----  
 -----  
 Downloading, using, copying, or modifying the GPOPS code constitutes an agreement to ALL of the terms of the GPOPS license. Please see the file LICENSE given in the home directory of the GPOPS distribution or see the summary file printed when running an example.  
 -----  
 -----

-----  
 Summary of Problem Written to File: Elbow OCP.txt  
 -----

Using Default Sparsity  
 Automatic Scaling Turned Off  
 Objective Gradient Being Estimated via Finite Differencing  
 Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	40	Linear constraints	5
Nonlinear variables	90	Linear variables	0
Jacobian variables	82	Objective variables	90
Total constraints	45	Total variables	90

The user has defined 0 out of 490 first derivatives

	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	4.9E-01	10	1.0E+02					
Major Minors	Step		nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	199		1	9.7E+00	1.3E-01	2.4697136E+06			r i
1	12	2.9E-01	2	6.5E+00	9.5E-02	2.0595922E+06	2		n rli
2	6	4.2E-01	4	5.6E+00	1.5E-01	1.6322518E+06	4		sM li
3	8	1.0E+00	6	5.5E+00	2.0E-02	1.0341106E+06	2		M i
4	3	1.0E+00	7	5.4E+00	7.5E-03	1.0316304E+06	2		i
5	1	1.0E+00	8	5.5E+00	4.0E-04	1.0314529E+06	2		i
6	1	1.0E+00	9	5.5E+00	5.6E-04	1.0314521E+06	2		i
7	1	1.0E+00	10	5.4E+00	6.1E-04	1.0314516E+06	2		i
8	2	1.0E+00	11	5.1E+00	4.5E-03	1.0314435E+06	1		i
9	1	1.0E+00	12	4.5E+00	4.5E-03	1.0314190E+06	1		i
Major Minors	Step		nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
10	1	1.0E+00	13	4.5E+00	3.9E-03	1.0314104E+06	1		i
11	1	1.0E+00	14	4.5E+00	3.9E-03	1.0313953E+06	1		i
12	1	1.0E+00	15	4.5E+00	(3.7E-07)	1.0313394E+06	1		R i c
12	2	1.0E+00	15	4.5E+00	(3.7E-07)	1.0313394E+06	1		R i c
13	4	1.0E+00	16	4.5E+00	(2.2E-08)	1.0313394E+12		1.1E+04	s i c

SNOPTA EXIT 10 -- the problem appears to be infeasible

SNOPTA INFO 13 -- nonlinear infeasibilities minimized

Problem name

No. of iterations	242	Objective value	8.2137751061E-03
No. of infeasibilities	12	Sum of infeas	1.0313392650E+02
Elastic weight	1.0E+10	Scaled Merit	1.0313392650E+02
No. of major iterations	13	Linear objective	0.0000000000E+00
Penalty parameter	1.064E+04	Nonlinear objective	8.2137751061E-03
No. of calls to funobj	1748	No. of calls to funcon	1748
Calls with modes 1,2 (known g)	16	Calls with modes 1,2 (known g)	16
Calls for forward differencing	1350	Calls for forward differencing	1350
Calls for central differencing	360	Calls for central differencing	360

No. of degenerate steps	18	Percentage	7.44
Max x	33 3.1E+00	Max pi	42 1.3E+11
Max Primal infeas	130 2.3E+01	Max Dual infeas	3 9.4E+03
Nonlinear constraint violn	2.3E+01		

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	49.86 seconds
Time for solution output	.00 seconds
Time for constraint functions	52.97 seconds
Time for objective function	.00 seconds

-----  
BEGIN: Computation of Endpoint Controls in Phase 1  
-----

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4
Total constraints	1	Total variables	4

The user has defined 0 out of 4 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	7.9746283E+11		r	c
Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	7.9746283E+11		r	c

SNOPTA EXIT 0 -- finished successfully

SNOPTA INFO 1 -- optimality conditions satisfied

Problem name

No. of iterations	0	Objective value	7.9746283478E+11
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	7.9746283478E+11
No. of calls to funobj	13	No. of calls to funcon	13
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	4	Calls for forward differencing	4
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	1 0.0E+00	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	0 0.0E+00

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.05 seconds
Time for solution output	.00 seconds
Time for constraint functions	.06 seconds
Time for objective function	.00 seconds

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4

Total constraints 1 Total variables 4

The user has defined 0 out of 4 first derivatives

Major	Minors	Step	nObj	Feasible	Optimal	Objective	nS
0	1		1		2.0E+07	1.4127632E+12	r
1	0	1.0E+00	2		(0.0E+00)	1.4127231E+12	n r c
1	0	1.0E+00	2	1	(0.0E+00)	1.4127231E+12	n r c

SNOPTA EXIT 0 -- finished successfully  
SNOPTA INFO 1 -- optimality conditions satisfied

Problem name

No. of iterations	1	Objective value	1.4127231499E+12
No. of major iterations	1	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	1.4127231499E+12
No. of calls to funobj	20	No. of calls to funcon	20
Calls with modes 1,2 (known g)	2	Calls with modes 1,2 (known g)	2
Calls for forward differencing	8	Calls for forward differencing	8
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	1 0.0E+00	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	0 0.0E+00

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.08 seconds
Time for solution output	.00 seconds
Time for constraint functions	.08 seconds
Time for objective function	.00 seconds

-----  
END: Computation of Endpoint Controls in Phase 1  
-----

-----  
GPOPS Version 2.2 beta: A MATLAB Implementation of the Gauss Pseudospectral Method  
Copyright (c) 2008 Anil V. Rao, David Benson, Geoffrey T. Huntington,  
Christopher L. Darby, Michael Patterson, and Camila Francolin  
-----

-----  
-----  
Downloading, using, copying, or modifying the GPOPS code constitutes an agreement to  
ALL of  
the terms of the GPOPS license. Please see the file LICENSE given in the home  
directory of  
the GPOPS distribution or see the summary file printed when running an example.  
-----  
-----

-----  
 Summary of Problem Written to File: Wrist OCP.txt  
 -----

Using Default Sparsity  
 Automatic Scaling Turned Off  
 Objective Gradient Being Estimated via Finite Differencing  
 Constraint Jacobian Being Estimated via Finite Differencing

Nonlinear constraints	40	Linear constraints	5
Nonlinear variables	90	Linear variables	0
Jacobian variables	82	Objective variables	90
Total constraints	45	Total variables	90

The user has defined 0 out of 490 first derivatives

	Minor	LPmult	nInf	SumInf	rgNorm	Elastic	LPobj		
	100	-2.3E-01	23	6.6E+02					
	200	-8.4E-02	20	1.2E+03					
	300	-1.8E+04	20	1.3E+03			1.5219062E+07		
Major	Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
0	309		1	1.3E+02	8.7E+00	2.8523340E+07			r i
1	15	1.7E-01	2	9.6E+01	8.8E+00	2.6312820E+07	3		n r l i
2	56	2.0E-01	3	8.5E+01	2.7E+00	2.4128904E+07	10		s l i
3	32	1.0E+00	5	4.6E+01	1.5E+01	1.5527623E+07	10		M i
4	12	1.0E+00	6	4.6E+01	3.7E+01	1.5434033E+07	11		i
5	5	1.0E+00	8	4.7E+01	1.1E+01	1.5380913E+07	11		M i
6	3	1.0E+00	9	4.9E+01	3.4E+00	1.5337504E+07	11		i
7	1	1.0E+00	11	4.9E+01	5.1E+00	1.5334463E+07	11		n i
8	2	1.0E+00	13	4.9E+01	5.9E+00	1.5332878E+07	10		n R i
9	2	1.0E+00	15	4.9E+01	1.0E+01	1.5330068E+07	9		n R i
Major	Minors	Step	nCon	Feasible	Optimal	MeritFunction	nS	Penalty	
10	10	1.0E+00	17	4.9E+01	1.1E+00	1.5325810E+07	4		n r i
11	18	1.0E+00	19	4.7E+01	2.6E-01	1.5182326E+07	3		sm i
12	2	1.0E+00	20	4.4E+01	2.0E-02	1.5144300E+07	4		i
13	1	1.0E+00	22	4.4E+01	7.5E-05	1.5144161E+07	4		n i c
13	2	1.0E+00	22	4.4E+01	7.5E-05	1.5144161E+07	4		n i c
14	4	1.0E+00	23	4.4E+01	(8.2E-13)	1.5144160E+13	4	3.5E+04	n R i c

SNOPTA EXIT 10 -- the problem appears to be infeasible

SNOPTA INFO 13 -- nonlinear infeasibilities minimized

Problem name

No. of iterations	473	Objective value	3.1487947561E-01
No. of infeasibilities	22	Sum of infeas	1.2654586349E+03
Elastic weight	1.2E+10	Scaled Merit	1.2654586349E+03
No. of major iterations	14	Linear objective	0.0000000000E+00
Penalty parameter	3.463E+04	Nonlinear objective	3.1487947561E-01
No. of calls to funobj	2399	No. of calls to funcon	2399
Calls with modes 1,2 (known g)	23	Calls with modes 1,2 (known g)	23
Calls for forward differencing	1980	Calls for forward differencing	1980
Calls for central differencing	360	Calls for central differencing	360
No. of superbasics	4	No. of basic nonlinears	22
No. of degenerate steps	18	Percentage	3.81
Max x	22 3.1E+00	Max pi	41 4.1E+10
Max Primal infeas	126 3.6E+02	Max Dual infeas	9 9.5E-02
Nonlinear constraint violn	3.6E+02		



Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	82.52 seconds
Time for solution output	.00 seconds
Time for constraint functions	85.19 seconds
Time for objective function	.00 seconds

-----  
BEGIN: Computation of Endpoint Controls in Phase 1  
-----

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4
Total constraints	1	Total variables	4

The user has defined 0 out of 4 first derivatives

Major Minors	Step	nObj	Feasible	Optimal	Objective	nS		
0	0	1		(0.0E+00)	-8.7547071E+12		r	c
0	4	1		7.3E+00	-8.7547071E+12		r	c
0	4	1		7.3E+00	-8.7547071E+12		r	c
0	4	1		7.3E+00	-8.7547071E+12		r	c

SNOPTA EXIT 40 -- terminated after numerical difficulties

SNOPTA INFO 41 -- current point cannot be improved

Problem name

No. of iterations	4	Objective value	-8.7547070655E+12
No. of major iterations	0	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	-8.7547070655E+12
No. of calls to funobj	22	No. of calls to funcon	22
Calls with modes 1,2 (known g)	1	Calls with modes 1,2 (known g)	1
Calls for forward differencing	4	Calls for forward differencing	4
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	4 1.9E-05	Max pi	1 1.0E+00
Max Primal infeas	0 0.0E+00	Max Dual infeas	1 1.5E+01

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.08 seconds
Time for solution output	.00 seconds
Time for constraint functions	.11 seconds
Time for objective function	.00 seconds

Nonlinear constraints	0	Linear constraints	1
Nonlinear variables	4	Linear variables	0
Jacobian variables	0	Objective variables	4
Total constraints	1	Total variables	4

The user has defined 0 out of 4 first derivatives

Major	Minors	Step	nObj	Feasible	Optimal	Objective	nS
0	3		1		8.9E+11	-5.3202644E+10	r
1	0	8.0E-01	2		1.8E+11	-2.7842657E+12	n r l
2	0	1.0E+00	3		(0.0E+00)	-3.4853059E+12	n r c
2	0	1.0E+00	3		(0.0E+00)	-3.4853059E+12	n r c

SNOPTA EXIT 0 -- finished successfully  
SNOPTA INFO 1 -- optimality conditions satisfied

Problem name

No. of iterations	3	Objective value	-3.4853058866E+12
No. of major iterations	2	Linear objective	0.0000000000E+00
Penalty parameter	0.000E+00	Nonlinear objective	-3.4853058866E+12
No. of calls to funobj	29	No. of calls to funcon	29
Calls with modes 1,2 (known g)	3	Calls with modes 1,2 (known g)	3
Calls for forward differencing	12	Calls for forward differencing	12
Calls for central differencing	8	Calls for central differencing	8
No. of degenerate steps	0	Percentage	.00
Max x	4 1.0E+00	Max pi	1 1.0E+00
Max Primal infeas	4 6.0E-13	Max Dual infeas	0 0.0E+00

Solution printed on file 9

Time for MPS input	.00 seconds
Time for solving problem	.09 seconds
Time for solution output	.00 seconds
Time for constraint functions	.14 seconds
Time for objective function	.00 seconds

-----  
END: Computation of Endpoint Controls in Phase 1  
-----

Total Solution Time (min): 3.0332