

THE CATHOLIC UNIVERSITY OF AMERICA

Exploiting Interlimb Coupling to Investigate Upper-Extremity Bimanual Loading in  
Robot-Assisted Neuro-Rehabilitation on Subjects with Stroke

A DISSERTATION

Submitted to the Faculty of the

Department of Electrical Engineering and Computer Sciences

School of Engineering

Of The Catholic University of America

In Partial Fulfillment of the Requirements

For the Degree

Doctor of Philosophy

By

Hoi Ba Nguyen

Washington, D.C.

2014

# Exploiting Interlimb Coupling to Investigate Upper-Extremity Bimanual Loading in Robot-Assisted Neuro-Rehabilitation on Subjects with Stroke

Hoi Ba Nguyen

Director: Peter S Lum, Ph.D.

Impaired arm function is one of the most common outcomes for the estimated 700,000 stroke survivors in the US each year. The arm contralateral to the damaged hemisphere is often weak, while the opposite arm is usually intact functionally. Arm weakness in muscles contralateral to the stroke is caused predominantly by the inability to activate the corticospinal pathways needed to activate agonist muscles. Training methods are needed to stimulate these pathways to promote neurorehabilitation. One approach that has received considerable attention is the use of bilateral training. It is known that representations of specific muscles are present in the contralateral motor cortex, and that homologous regions of motor cortex in the two hemispheres are connected through the corpus callosum. Therefore, bilateral activity may result in enhanced activation of the cortical representations of weak muscles through activation of callosal pathways from the undamaged to the damaged hemisphere.

In this dissertation, we performed experiments to investigate the optimal bilateral training parameters. Loading of the non-paretic limb during bilateral symmetric movements may enhance the interlimb coupling effect by increasing activity in the pathway from the undamaged

hemisphere to the damaged cortex. However, prior studies in stroke survivors have found contradictory results. Different loading profiles and experimental conditions may explain these contradictory findings. To resolve this controversy, we have performed the first study that used robotic methods to systematically evaluate the effects of different loading profiles and amplitudes applied to the non paretic limb during bilateral symmetric elbow extension.

In order for the robot to accurately load the non-paretic limb, we faced the technical hurdle of compensating for the intrinsic dynamics of the robot (MIT-MANUS). First, we developed and compared two inertia compensation algorithms for the robot. One of the methods used a novel algorithm for digital differentiation of the encoder signals from the robot. This new method reduced the robot intrinsic impedance up to 64%, and tangential force anisotropy was reduced by 74%. We then developed robot algorithms to provide three different loading profiles: inertial, constant and spring resistance. Second, we performed an experiment with post-stroke survivors that examined the effects on the kinematics of paretic limb elbow extension from loading of the non-paretic limb. We found that all of the bilateral movements were slower than unilateral movements, probably because of the constraint placed on the subjects to move the two arms in synchrony (verbal instruction). Increasing load level in the non-paretic limb improved speed, acceleration, EMG in the paretic limb within resistive loading under spring and constant force patterns. The constant loading appears to be most effective of the bimanual conditions, and inertial loading was the least effective. Increasing inertial loading actually decreased the speed of the paretic limb.

In the previous study, the two limbs were coupled by the instruction to move the two limbs in synchrony. Data from this study suggested that paretic limb speed was highest in trials where the two limbs were highly synchronized. In a follow on study in healthy controls, we studied different methods to enforce greater synchrony between the two limbs. An EMG coherence method was used to assess the degree of interlimb coupling as a function of different coupling methods. To simplify the experiment, we studied isometric elbow extension and flexion torque generation instead of movement. During bilateral elbow torque generation, three types of coupling methods were tested: low difficulty visual coupling, high difficulty visual coupling, and haptic coupling through a mechanical apparatus. For elbow extension, 8-51 Hz coherence was higher in the haptic coupling condition compared to the two visual coupling conditions. The coherence was largest with agonist muscle pairs during the ramp phase of torque generation. No difference in coherence was seen across test conditions for elbow flexors. To our knowledge, this was the first study to use EMG coherence to evaluate bilateral coupling.

In conclusion, we discuss the implications of these studies, and make suggestions for future work. The novel aspects of this work relate to the development of an active inertia compensation algorithm for the MIT-MANUS robot, first systematic evaluation of bilateral interlimb coupling in stroke survivors using robotic loading of the non-paretic limb, and first study to use EMG coherence to evaluate the degree of interlimb coupling as function of different coupling methods.



This dissertation by Hoi Ba Nguyen fulfills the dissertation requirement for the doctoral degree in electrical engineering approved by Peter S Lum, Ph.D., as Director, Michelle L Harris-Love, Ph.D., Charles C Nguyen, D.Sc., and Sang Wook Lee, Ph.D., as readers.

---

Peter S Lum, Ph.D., Director

---

Michelle L Harris-Love, Ph.D., Reader

---

Charles C Nguyen, D.Sc., Reader

---

Sang Wook Lee, Ph.D., Reader

# Contents

<b>Figures</b>	<b>vi</b>
<b>Tables</b>	<b>ix</b>
<b>Acknowledgements</b>	<b>x</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
Interlimb Coupling	1
InMotion2 Robot	3
Dynamic Compensation Algorithms	4
EMG-EMG Coherence	6
Problem Statements and Contributed Works	9
<b>Chapter 2 InMotion2 Compensation Algorithm</b>	<b>12</b>
Introduction	12
Materials and Methodology	17
<i>InMotion2 Dynamics</i>	<i>17</i>
<i>Software Algorithms</i>	<i>18</i>
<i>Experimental Setup and Tasks</i>	<i>23</i>
<i>Statistical Analysis</i>	<i>25</i>
Results	27
Discussions	32

Conclusions	35
<b>Chapter 3 Interlimb Coupling with Bimanual Loading on Subjects with Stroke</b>	<b>37</b>
Introduction	37
Methodology	41
<i>Participants</i>	<i>41</i>
<i>Procedure</i>	<i>44</i>
<i>Data Acquisition and Analysis</i>	<i>47</i>
Results	55
Discussions	75
Conclusions	78
<b>Chapter 4 EMG Coherence in Bilateral Tasks with Visual Feedback on Healthy</b>	<b>81</b>
Introduction	81
Methodology	85
<i>Procedure and Apparatus</i>	<i>85</i>
<i>Data Acquisition and Analysis</i>	<i>91</i>
Results	100
Discussions	114
Conclusions	121
<b>Chapter 5 Conclusions</b>	<b>122</b>
<b>Appendix</b>	<b>125</b>

<i>The Experimental Desk and Chair</i>	125
<i>MicroFET2 Muscle Tester</i>	125
<i>Resistive Loading Test</i>	127
<i>Delsys Bagnoli EMG Unit</i>	129
<i>FMA Scoresheet</i>	130
<i>Outlier Quartiles</i>	132
<i>Cross Correlation Demonstration</i>	132
<i>Fourier Transform</i>	133
<i>Calibration of Tekscan Force Sensors</i>	135
<i>The Coherence Apparatus Mechanical Parts</i>	137
<i>Noraxon EMG Unit and Accessories</i>	138
<i>National Instruments Data Acquisition</i>	138
<i>Motor Unit</i>	139
<i>The Coherence GUI Code</i>	140
<b>Bibliography</b>	<b>213</b>

## Figures

2.1	Top-down drawing of InMotion2 robot with locations of targets	14
2.2	Comparison of time lag between BD and QF	19
2.3	Block diagram of the robot arm control program	22
2.4	Ellipses representing tangential peak-to-peak interaction forces	30
2.5	Force profile along the direction of motion for 4 targets at 100 cm/s	32
3.1	InMotion2 robot is used for either left or right non-paretic limb loading	40
3.2	Reaching task apparatus	44
3.3	Typical pathways of the bilateral movements of two hands	49
3.4	Triceps RMS EMG graphs	51
3.5	Typical cross correlation (xCorr) graphs	53
3.6	Temporal onset and complete synchrony, spatial onset synchrony	54
3.7	Bar plot of peak velocity and acceleration of the whole group of 9 subjects	58
3.8	One low movement subject in bar & profile plots of velocity and acceleration	60
3.9	Time to Peak (T2P) and peak triceps RMS EMG (P-EMG) graphs	61
3.10	Peak biceps RMS EMG	62
3.11	Cross correlation (xCorr) time lag and movement time (MT) graphs	63
3.12	Scatter charts of cross correlation lag of 10 and 12 subjects	65
3.13	Scatter charts of cross correlation lag, classified by groups of conditions	66
3.14	Scatter charts of onset temporal synchrony of 10 and 12 subjects	67
3.15	Scatter charts of onset temporal synchrony, classified by groups of conditions	69
3.16	Plot of peak velocity and peak-to-peak acceleration of one distinct subject	69
3.17	Peak triceps P-EMG, NMU, and xCorr lag of one distinct subject	70

3.18	Velocity at the time the target is reached	71
3.19	Plot of peak velocity and acceleration of the group of 2 healthy subjects	72
3.20	Plot of peak velocity and acceleration of the group of 2 removed stroke subjects	73
3.21	Low (2 subjects) versus high speed group (7 subjects)	79
4.1	The coherence study: task apparatus of elbow extension/flexion	86
4.2	GUI's data collection of a successful trial in task condition 1 or 4	95
4.3	GUI's data collection of a successful trial in task condition 2 or 5	96
4.4	GUI's data collection of a successful trial in task condition 3 or 6	97
4.5	Plot of raw EMG and RMS EMG of five trials in a task condition	99
4.6	Total coherence on ramp phase of the agonist/antagonist muscle pairs	100
4.7	Total coherence on stay phase of the agonist/antagonist muscle pairs	101
4.8	Coherence on agonist ramp for extension for each of the four frequency bands	102
4.9	Total coherence on ramp phase of agonist-antagonist	105
4.10	Force variation on ramp and stay phases	107
4.11	RMS EMG of ramp biceps and triceps	109
4.12	RMS EMG of stay biceps and triceps	110
4.13	Coherence plot versus frequency of the task condition 1 in one subject	111
4.14	Coherence plot of the task conditions 2 and 3 in one subject	112
4.15	Coherence plot of the task conditions 4 and 5 in one subject	113
4.16	Coherence plot of the task condition 6 in one subject	114
4.17	Presentation of EMG in coherence	115
A.1	The table and chair with straps which were used for all three studies	125
A.2	Hand-held force sensor microFET2	126

A.3	The ATI force sensor	127
A.4	The Bagnoli-16 EMG unit	129
A.5	Boxplot with with quartiles and an interquartile range	132
A.6	Cross correlation demonstration	133
A.7	Fourier transform	134
A.8	Flexiforce A401	136
A.9	Mechanical parts of the coherence apparatus	137
A.10	Myosystem 1400A	138
A.11	National Instruments DAQ USB 6210	138
A.12	Motor unit	139
A.13	Properties of motor units	139
A.14	Slow, fast fatigue-resistant, and fast fatigable motor units	140

## Tables

2.1	Kinematics	26
2.2	Peak-to-Peak Interaction Forces	28
2.3	Anisotropy in peak-to-peak forces (maximum force minus minimum force)	29
3.1	List of participants	43
3.2	Statistical analysis p-value outcome for the whole group of 9 subjects	56
3.3	Statistical analysis for two groups of subjects divided by higher and lower FMA	74
4.1	Task conditions in random order and unique for each subject	91
4.2	Statistics on combined alpha-beta-gamma band	103
4.3	Statistics on alpha and beta bands separately	104
A.1	ATI calibration	128
A.2	FMA scoresheet	130
A.3	FlexiForce calibration	137



## **Acknowledgements**

I would like to express great thanks to my mentor Dr. Peter Lum for years of thoughtful and valuable advices during the course of my dissertation, as well as his kindness. This inspiration will accompany me for years to come when I continue working on the products to produce them with the highest quality. Many thanks go to Dr. Michelle Harris-Love with years mentoring on the neuroscience, weekly discussions, and her inspired study on interlimb coupling. I admired her optimistic style everyday as well as her teamwork spirit in keeping people working together. I enjoyed the initial six months working with my former advisor Dr. Tobias Nef before he left for a career change in Switzerland. I really miss the exciting moments when we worked together. Thanks also go to the individuals for providing me with financial supports. Many thanks go to Dr. Charles Nguyen for introducing me to the field of robotics which opened door for me to be in touch with the above professors. I really appreciate his kindness in providing my wife and me with a graduate assistantship that enables us to join CUA at the same time. Many thanks also go to Dr. Sang Wook Lee for his kind help and instructions on the coherence study. It was very fortunate to be able to work with him since his experience in the relevant subjects helped me tremendously.

Many thanks go to Evan Chan, iian Black, Justin Carter, Peggy Bruce, Maureen Whitford, Wendy Fuller, Diane Nichols, Mary Kate Cullinane, Rahsaan Holley, Erika Tinoco for providing me with academic, experimental, administrative supports and for their friendship. Final thanks go to about other forty people that I can't name here, for their participation in

various experiments. Last but not least, my adored wife and my handsome son, my parents and my siblings deserve my deepest gratitude for their love, encouragement and understanding.

# **Chapter 1**

## **Introduction**

### **1.1 Interlimb Coupling**

Interlimb coupling, a basic category of neuromotor system organization, concerns how movements are coordinated across limbs through nervous system (Kelso et al., 1979), can be exploited to improve impaired limb movement performance during bilateral limb tasks. The motor system shows a strong tendency toward symmetry and synchronicity between the paretic and non-paretic limb movements when bimanual limb tasks are executed (Goodman et al., 1983; Kelso et al., 1979; Serrien et al., 1999). Therefore, when poststroke hemiparesis patients perform bilateral upper extremity movements, the overflow (interhemispheric sharing of normal movement commands from the undamaged hemisphere) of movement characteristics from the non-paretic limb to the paretic arm could be activated to improve the paretic limb performance (Mudie et al., 2000; Lewis et al., 2004).

Prior studies of upper extremity interlimb coupling in participants with poststroke hemiparesis have had contradictory findings. Several studies (Rice et al., 2001; Lewis et al., 2004; Chang et al., 2006; Messier et al., 2006; Kilbreath et al., 2006) showed that interlimb coupling after stroke results in no improvement in paretic arm motor performance and disrupted non-paretic arm motor performance. One study (Ustinova et al., 2006) showed that interlimb

coupling was impaired and unstable for rhythmical tasks. However, four studies have shown that some subjects with hemiparesis show improvement in paretic arm performance during bilateral tasks compared with unilateral tasks (Cunningham et al., 2002; Rice et al., 2004; Harris-Love et al., 2005; Waller et al., 2006).

With discrete single-joint reaching tasks with arm support, bilateral reaching tasks were reported to facilitate movement of the paretic arm in half of the subjects they tested (Cunningham et al., 2002). Interlimb coupling conducted on stroke subjects with box-reaching task at fastest possible speed showed a mutual beneficial effect early in the movement (increased paretic and non-paretic peak acceleration) and then, later in the movement, the non-paretic limb affected the paretic arm positively, resulting in a higher paretic peak velocity while there was unchanged non-paretic peak velocity (Harris-Love et al., 2005). Therefore, interlimb coupling effects can be used to produce an immediate improvement in paretic arm reaching performance. In another study, bilateral upper limb movements improved paretic movement time and movement speed compared to sequential movement (Waller et al., 2006).

Studies on loading of the non-paretic limb have also yielded mixed results. Adding inertia to the non-paretic arm during bilateral reaching task brought benefits to paretic arm performance (Cunningham et al., 2002) in terms of movement smoothness. In contrast, non-paretic arm loading showed no positive effect in coffee mug moving tasks (Chang et al., 2006). Loading on the non-paretic arm during bilateral reaching did not result in further improvement in paretic arm performance (Harris-Love et al., 2005).

## 1.2 InMotion2 Robot

In our study, a robot is used to precisely control the loading of the non-paretic arm. InMotion2 arm robot is applied for rehabilitation and motor learning, with the key feature of low end-point impedance, made possible by using direct drive motors to apply force at the end effector. In numerous applications, the end point force generated by the robot is controlled open loop and depends only on the Jacobian. In these cases, low intrinsic impedance is critical for both rehabilitation and motor learning applications to fulfill accuracy in the perturbed forces.

InMotion2 is the commercial version of the MIT-MANUS arm robot for rehabilitation following neurological injuries (Krebs et al., 1998; Lo et al., 2010). The robot is a two-degree-of-freedom robot that assists planar pointing movements of the shoulder and elbow. A key feature is low intrinsic end-point impedance, made possible by direct drive direct-current motors at the base of the device that drive a linkage mechanism that can apply force at the end effector in any direction within the horizontal plane. For rehabilitation, active control is impedance based, whereby the robot minimally interferes with normal movement and applies assistance only when needed to complete tasks. The robot can also be used to quantify motor impairments in patient populations (Bosecker et al., 2010; Finley et al., 2009).

However, these applications are hindered by the fact that even low-impedance robots can alter the neural control strategies employed during natural movements outside of the robot. Campolo and colleagues showed that in a wrist pointing task, subjects solve the redundancy problem by using intrinsic or “neural” constraints that restrict wrist rotations to subject-specific

2-D surfaces within the wrist's 3-D configuration space (Campolo et al., 2009). When a hand performs the same task attached to a low-impedance robot, the 2-D surfaces are perturbed by the non-zero impedance of the robot, leading to surfaces that were remarkably consistent from trial to trial and between subjects. Importantly, if the robot impedance is reduced with a force control algorithm, subject-specific 2-D surfaces reappeared (Tagliamonte et al., 2011). In terms of rehabilitation, the robot impedance results in an artificial haptic environment during robotic training, which may inhibit recovery of efficient movement strategies and limit performance gains outside of the robot. The InMotion2 and similarly designed robots are also extensively used in motor adaptation studies whereby the robot applies novel force fields to the arm, and over repeated movement trials, one can study the sensorimotor processes associated with implicit adaptation to the novel environment (Hwang et al., 2006; Huang et al., 2010; Schabowsky et al., 2007 and 2008; Scheidt et al., 2007). In these cases, the end point force applied by the robot is often controlled open loop and based solely on the Jacobian that relates end point force to the motor torques. Thus, low intrinsic impedance is critical to achieve accuracy in the applied forces. However, most studies ignore the magnitude and anisotropy of the robot intrinsic impedance.

### **1.3 Dynamic Compensation Algorithms**

Two methods for digital differentiation of the robot encoders were developed as part of our dynamic compensation algorithms, including a classic backward derivative and a novel method involving use of a least squares polynomial fit of recent data at each time step.

The backward divided differentiation approach has been commonly used, i.e., difference equations (Oppenheim and Schafer, 1975), digital differentiator design (Mitra and Kaiser, 1993), Newton polynomial interpolation (Vaseghi, 2000), and discrete first and second derivative (Elali, 2004). The backward differentiation method can have a large error due to its simple approximation (Chapra and Canale, 2005). Noise is also amplified by the process. Errors and noise in calculating the velocity will yield much larger problems in acceleration estimation. To reduce noise, a filter is commonly used (Palazzolo et al., 2007; Schabowsky et al., 2007). However, the filter creates delay for real-time velocity and especially real-time acceleration. This delay exposed drawbacks of backward derivative when we moved the robot at high speed, around 100 cm/s or above. The central differentiation method has much better accuracy (Chapra and Canale, 2005) to be applied for our problem because it uses the previous value and future value to calculate the mid-point derivative, however central differentiation cannot be implemented in real-time. Our aim was to find a good method to best estimate the future values of velocity and acceleration, in real-time. Therefore, we proposed an improved method of this real-time derivative computation based on the method of least-squares regression.

Real-time sophisticated least-squares regression techniques have been used in many different areas of study. Real-time tracking of the location and dynamic motion of a mobile user using constrained least-squares estimation has been shown to have improved accuracy (Wang and Ching, 2006). A nonlinear version of recursive least-squares algorithm was explored to construct mean squared error solutions to nonlinear least-squares problems that are frequently encountered in signal processing applications (Engel et al., 2004). Regression in real-time

processing of neurophysiological signals was used to implement unrealizable ideal filters (Krieger et al., 1996). Partial least-squares regression was applied to achieve a new medical imaging technique for predictive cardiac motion modeling and correction (Ablitt et al., 2004). Partial least-squares regression was exploited to predict the position of the diaphragm to improve the performance of a real-time couch-based motion compensation system during radiotherapy (Qiu et al., 2007). Complex linear regression of the transmission coefficient was explored to achieve highly accurate and real-time determination of resonant characteristics (Inoue et al., 2004). The global motion parameters were estimated using fast least trimmed square regression and hierarchical processing in order to achieve a robust approach to the detection and tracking of small targets with low contrast (Wang et al., 2005). A spectrum estimation for random samples recorded for real-time applications was solved based upon the Fourier series and the method of least-squares (Kar et al., 1981).

#### **1.4 EMG-EMG Coherence**

Coherence determines the degree of correlation between two signals in the frequency domain (Rosenberg et al. 1989). The presence of EMG-EMG coherence in bilateral muscles is evidence for interlimb coupling of muscles through cortical, subcortical or spinal mechanisms. This coherence was found between bilateral arm muscles during fatiguing contractions, i.e. elbow flexor or extensor against upward/downward loading of a steel bar (Boonstra et al 2007). The coherence was highest in the range of 8-12 Hz and increased with fatigue, and was clearer for extension than flexion. In an isometric contraction task of abducting the thumb against a



manipulandum, the analysis found that EMG-EMG coherence is maximal in the range of 1-12 Hz (Farmer et al 2007). While the coherence is high in the range of 16-32 Hz, this depends on cortical drive to motoneurons and is coherent with cortical oscillations at around 20 Hz. The coherence, especially at around 20 Hz, is small for 4-9 year-old children, but gets larger in subjects at ages 12-14 years to adults. EMG coherence is stronger in extrinsic than intrinsic muscle pairs for a series of 3-digit (thumb, index and middle fingers) grasp tasks in 8 subjects. In addition, the coherence is not affected by force level (Poston et al 2010).

Kilner et al., 1999 showed peak EMG coherence at the range of 15-30 Hz for a 'hold' precision grip task at about two Newtons. While during the ramp task (hold-ramp-hold precision grip), the coherence at the same range is significantly reduced during the ramp movement, but significantly increased in the second hold period, relative to the initial hold. Note that they use rectified-EMG that may affect the results.

Another study also used rectified-EMG. Bilateral EMG coherence was found in the frequency band of 7-13 Hz for about one second in the phase of increasing force of both hands in a thumb-index-fingers grip task to a stable force production (Boonstra et al 2009). However, the EMG synchronization is diminished or absent during the stable force phase of grip. Target force was one Newton and task duration was 4.5 seconds in which the ramp was about one second. Thirteen healthy participants were recruited. EMGs on the first dorsal interosseous (FDI) muscle and flexor pollicis brevis (FPB) muscle of both hands were recorded.

Though this is not directly related to pure EMG coherence analysis, in an isometric contraction task on six healthy subjects, 15-33 Hz cortical-muscle (MEG-EMG) coherence was

found for upper and lower limb muscles (Salenius et al., 1997). In an isometric contraction task on 6 healthy subjects, 15-33 Hz cortical-muscle (MEG-EMG) within-limb coherence was found for upper and lower limb muscles. Note that they used rectified-EMG. Electrocorticogram - electromyogram coherence during isometric contraction of wrist extensor is observed between primary motor cortex and EMG at 12-18Hz in all 8 subjects (Ohara et al, 2000). Six subjects performed isometric contraction with four muscles (Gross et al, 2000) and cortico-muscular coherence was found around 20 Hz. The phase lag in this data supported the hypothesis that this 20Hz coherence is due to the primary motor cortex driving the motoneuron pool. The study showed strongest cortico-muscular coherence in isometric contraction in range of 14-40Hz as well as particular role of beta band (they define as 13-24 Hz) in movement control.

To decide a movement task is either objective (apply force to move over a distance) or isometric (apply certain force without movement). Though contradictory, the above review implied that more studies mentioned that isometric movement may be a better choice to produce higher coherence. In addition, slower movement is likely better in producing coherence (Evans et al., 2003; McAuley et al., 1999). With index finger in-phase movements, bilateral coherence was most exhibited within the transition from a force ramp to stable phase of contractions (Evans et al., 2003). In contrast, during a grip task, the cortical muscle coherence was significantly smaller when the task was performed under an isometric condition compared with a compliant condition in which subjects moved the levers against a spring-like load (Kilner et al., 2000). Furthermore, larger displacements during the compliant condition of the grip task produced higher coherence, and so isometric condition gave least coherence. EMG coherence was found between leg

muscles of the walking task or standing even though it is not strong as within leg (Boonstra et al., 2008; Halliday et al., 2003). Note that Boonstra used rectified-EMG in this study.

Coherence may be enhanced in tasks with visual feedback. EMG coherence was improved after visuo-motor (ankle dorsi-platarflexion movement) training session in 8 out of 11 subjects at range of 15-35 Hz (Perez et al., 2006). Similarly, EEG-EMG coherence around the same range was significantly increased in 9 out of 11 subjects after the training, specific for trained muscles. The coherence remained unchanged for untrained muscles. In this study, we investigated the effects of visual and haptic feedback methods on coherence between muscles in the two limbs during bilateral symmetric isometric force generation.

## **1.5 Problem Statements and Contributed Works**

First, most studies neglect the magnitude and anisotropy of intrinsic impedance of the robot. This could be problematic in cases where data from left and right arms are pooled together or compared, and when movements in different directions are compared. We first sought to eliminate these effects, in order to implement accurate loading profiles. We must quantify the intrinsic impedance of the InMotion2, and develop a compensation algorithm to reduce the impedance felt by the subject during use of the robot. The dynamic equations of motion of the robot were derived and a feedforward compensation scheme was implemented whereby the algorithm commands the robot motors to generate torques real-time to compensate for inertial and velocity-dependent forces that would normally be felt by the user during dynamic movements. Successful implementation is heavily dependent on the accuracy of real-time

calculation of velocities and accelerations of the robot links. Two methods for digital differentiation of the robot encoders were tested, including a novel method involving use of a least squares polynomial fit of recent data at each time step. Performance of the algorithms were tested by measuring the robot-user interaction forces during fast reaching movements. After successful development and testing, we included our algorithms of intrinsic inertia compensation into our loading paradigm for the interlimb coupling experiments to produce more precise force perturbations.

Second, we thought the inertial loading might have limited effects if movements are at slow or constant velocities. Therefore, other types of loading profiles should be investigated. We systematically studied the effects of different loading profiles and amplitudes using a robotic device attached to the non-paretic limb during bilateral symmetric elbow movements. The hypotheses of the interlimb coupling study were to investigate the influence of novel loading for reaching movements in chronic stroke. With a novel set-up of tasks, conditions, apparatus, measurement and analysis methods, we would (1) investigate if bilateral reaching would result in improved paretic arm performance, compared with unilateral reaching; (2) explore if during the bilateral reaching task, paretic arm reaching performance would be improved by resistive loading on the non-paretic arm using a robotic device, and; (3) discover if during bilateral reaching task, the type and amplitude of the loading profile applied to the non-paretic limb has an effect on paretic limb performance.

Third, since the stroke interlimb coupling data also suggested that the degree of synchrony between the two arms affects the strength of the coupling effect. As a follow study,

EMG coherence was employed to explore effects of tighter visual and haptic constraints to seek more synchrony between two limbs. Coherence on subjects with stroke is weaker than on the healthy adults. We made a decision to move on with a totally new setup of coherence, and on healthy subjects only. The coherence study on healthy adults is with hypotheses regarding (1) Haptic feedback tasks will have higher coherence than visual feedback tasks ; (2) Visual feedback tasks that strictly force synchrony between limbs will have higher coherence than visual feedback tasks that require less strict synchrony; (3) Elbow extensors will have similar coherence patterns as elbow flexors.

The novel contributions of these studies relate to the development of an active inertia compensation algorithm for the InMotion2 robot, first systemic evaluation of bilateral interlimb coupling in stroke survivors using robotic loading of the non-paretic limb, and first study to use EMG coherence to evaluate the degree of interlimb coupling as function of different coupling methods.

## **Chapter 2**

### **InMotion2 Compensation Algorithm**

#### **2.1 Introduction**

The InMotion2 and similarly designed robots, are commonly used for rehabilitation of neurological injuries and motor adaptation studies. These robots are used to simulate haptic environments; however, anisotropy in end-point impedance due to the intrinsic robot dynamics can compromise these experiments. The goal of this chapter was to decrease the magnitude and anisotropy of the robot impedance using a dynamic compensation algorithm that reduces the forces normally felt by the user during rapid movements.

The assumption of negligible inertia would be accurate only for certain operating conditions, and this is the case for some types of simple mechanical systems (Doebelin et al., 1998). However, when we want to develop more accurate environment models, we will include this inertia for some parts for mechanical systems. For example, the moving part of the damper should have inertia. Similarly, the mass of the damper cylinder should be counted. Furthermore, suitable model choice and online estimation of the mechanical impedance during the contact of a robotic system with an unknown environment can improve the interaction performance between robotic devices and unknown environments (Diolaiti et al., 2005) of either stiff or soft material.

For human-robot cooperative task, (Tsumugiwa et al., 2004) says that impedance characteristics of the robot and its compliance have significant impact on system stability. By including the compliance of the robot in the simulation, there is a match between simulation and experimental results in stability evaluation. Compensation for the friction, gravity and inertia were mentioned in (Dodds and Glover, 1995; Goto et al., 2007) to achieve precise control or new control method with external forces.

The InMotion2 (Interactive Motion Technologies Inc., Watertown MA, USA) is the commercial version of the MIT-MANUS arm robot for rehabilitation following neurological injuries (Krebs et al., 1998; Lo et al., 2010). The InMotion2 is a 2-DOF robot that assists planar pointing movements of the shoulder and elbow (Figure 2.1). A key feature is low intrinsic end-point impedance, made possible by direct drive DC motors at the base of the device that drive a linkage mechanism that can apply force at the end effector in any direction within the horizontal plane. For rehabilitation, active control is impedance based, whereby the robot minimally interferes with normal movement and applies assistance only when needed to complete tasks. The InMotion2 can also be used to quantify motor impairments in patient populations (Bosecker et al., 2010; Finley et al., 2009).

However, these applications are hindered by the fact that even low-impedance robots can alter the neural control strategies employed during natural movements outside of the robot. Campolo and colleagues showed that in a wrist pointing task, subjects solve the redundancy problem by using intrinsic or “neural” constraints that restrict wrist rotations to subject-specific 2-D surfaces within the wrist’s 3-D configuration space (Campolo et al., 2009). When a hand

performs the same task attached to a low-impedance robot, the 2-D surfaces are perturbed by the non-zero impedance of the robot, leading to surfaces that were remarkably consistent from trial to trial and between subjects. Importantly, if the robot impedance is reduced with a force

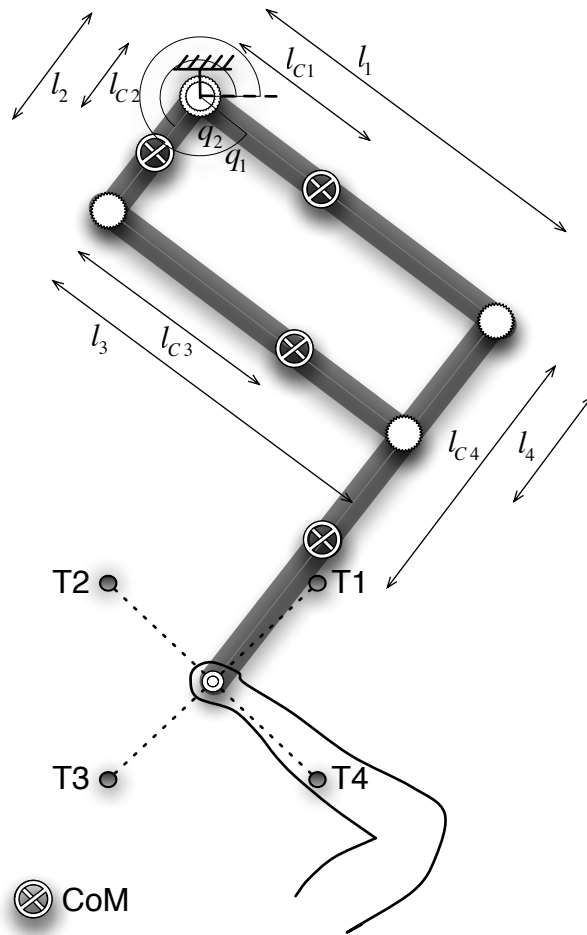


Figure 2.1. Top-down drawing of InMotion2 robot with locations of targets. The mass (kg), moment of inertia ( $\text{kg}\cdot\text{m}^2$ ), proximal link to center of mass distance (m), and length (m) of links 1 to 4 are:  $m_1=1.3936$ ,  $m_2=0.8143$ ,  $m_3=0.7138$ ,  $m_4=1.5394$ ;  $I_1=0.03346$ ,  $I_2=0.00465$ ,  $I_3=0.01777$ ,  $I_4=0.05770$ ;  $l_{c1}=0.1021$ ,  $l_{c2}=0.0728$ ,  $l_{c3}=0.2032$ ,  $l_{c4}=0.2461$ ;  $l_1=0.4063$ ,  $l_2=0.1523$ ,  $l_3=0.4064$ ,  $l_4=0.1524$ .



control algorithm, subject-specific 2-D surfaces reappeared (Tagliamonte et al., 2011). In terms of rehabilitation, the robot impedance results in an artificial haptic environment during robotic training, which may inhibit recovery of efficient movement strategies and limit performance gains outside of the robot.

The InMotion2 and other similarly designed robots are also extensively used in motor adaptation studies whereby the robot applies novel force fields to the arm, and over repeated movement trials, one can study the sensorimotor processes associated with implicit adaptation to the novel environment (Hwang et al., 2006; Huang et al., 2010; Schabowsky et al., 2007; Schabowsky et al., 2008; Scheidt et al., 2007). In these applications, the end point force applied by the robot is often controlled open loop and based solely on the Jacobian that relates end point force to the motor torques. In these cases, low intrinsic impedance is critical to achieve accuracy in the applied forces. However, most studies ignore the magnitude and anisotropy of the intrinsic impedance of the robot. This could be problematic in cases where data from left and right arms are pooled together or compared, and when movements in different directions are compared.

In this study we quantified the intrinsic impedance of the InMotion2, and developed a compensation algorithm to reduce the impedance felt by the subject during use of the robot. The dynamic equations of motion of the robot were derived and a feedforward compensation scheme was implemented whereby the algorithm commands the robot motors to generate torques real-time to compensate for inertial and velocity-dependent forces that would normally be felt by the user during dynamic movements. Successful implementation is heavily dependent on the accuracy of real-time calculation of velocities and accelerations of the robot links. Two methods

for digital differentiation of the robot encoders were tested, including a novel method involving use of a least squares polynomial fit of recent data at each time step. Performance of the algorithms were tested by measuring the robot-user interaction forces during fast reaching movements.

The backward divided differentiation has been using long time ago, i.e., difference equations (Oppenheim and Schafer, 1975), digital differentiator design (Mitra and Kaiser, 1993), Newton polynomial interpolation (Vaseghi, 2000), and discrete first and second derivative (Elali, 2004). The backward differentiation method has its large error due to its simple approximation (Chapra and Canale, 2005). A large error in calculating the velocity will yield much larger error in acceleration. To reduce this error, a filter was used (Palazzolo et al., 2007; Schabowsky et al., 2007). However, the filter created delay for real-time velocity and especially real-time acceleration. This delay exposed drawbacks of backward derivative when we moved the robot at high speed, around 100 cm/s. The central differentiation method has much better accuracy (Chapra and Canale, 2005) to be applied for our problem because it uses the previous value and future value to calculate the mid-point derivative. Our aim was to find a good method to best estimate the future value of displacement and velocity, in real-time. Therefore, we proposed an improved method of this real-time derivative computation based on the method of least-squares regression.

Studies on real-time sophisticated least-squares regression technique were in multi majors. A real-time tracking of the location and dynamic motion of a mobile user using constrained least-squares estimation algorithm had shown the improved accuracy (Wang and

Ching, 2006). Nonlinear version of recursive least-squares algorithm was explored to construct mean squared error solutions to nonlinear least-squares problems that are frequently encountered in signal processing applications (Engel et al., 2004). Regression in real-time processing of neurophysiological signals was used to obtain equivalent application of unrealizable ideal filters (Krieger et al., 1996). Partial least-squares regression was applied to achieve a new medical imaging technique for predictive cardiac motion modeling and correction (Ablitt et al., 2004). Partial least-squares regression was exploited to predict the position of the diaphragm to improve the performance of a real-time couch-based motion compensation system during radiotherapy (Qiu et al., 2007). Complex linear regression of the transmission coefficient was explored to achieve highly accurate and real-time determination of resonant characteristics (Inoue et al., 2004). The global motion parameters were estimated using fast least trimmed square regression and hierarchical processing in order to achieve a robust approach to the detection and tracking of small targets with low contrast (Wang et al., 2005). Or, back to many years ago, a spectrum estimation for randomly samples record for real-time applications was solved based upon the Fourier series and the method of least-squares (Kar et al., 1981).

## **2.2 Materials and Methodology**

### ***2.2.1 InMotion2 Dynamics***

The Euler-Lagrange approach was used to derive the inverse dynamics equations for the InMotion2 (Figure 2.1).

$$\begin{aligned}
\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} &= \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & I_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} (s_1 c_2 - c_1 s_2)(m_3 l_{c3} + m_4 l_{c4} l_1) \dot{q}_2 \dot{q}_1 \\ (c_1 s_2 - s_1 c_2)(m_3 l_{c3} + m_4 l_{c4} l_1) \dot{q}_1 \dot{q}_2 \end{bmatrix} \\
I_{11} &= m_1 l_{c1}^2 + m_3 l_{c3}^2 + m_4 l_1^2 + I_1 + I_3 \\
I_{22} &= m_2 l_{c2}^2 + m_3 l_2^2 + m_4 l_{c4}^2 + I_2 + I_4 \\
I_{12} = I_{21} &= (s_1 s_2 + c_1 c_2)(m_3 l_2 l_{c3} + m_4 l_{c4} l_1)
\end{aligned} \tag{2.1}$$

$m_i$ ,  $I_i$ ,  $l_{ci}$ ,  $l_i$  are the mass, moment of inertia, distance from proximal link to center of mass, and length of link  $i$  ( $i=1-4$ ), respectively. These mechanical properties were available from the manufacturer. We also confirmed the mass and center of mass location of each link after disassembling the robot linkage. Moment of inertia can also be estimated using the pendulum method (Dowling et al., 2006).  $\tau_k$  and  $q_k$  are the torques and angular displacements of the two motors ( $k=1-2$ ).  $s_k = \sin(q_k)$  and  $c_k = \cos(q_k)$ . The subject moves the robot handle and consequently changes the angular displacements, velocities and accelerations of the robot links. At each time step, the motor torques were calculated using equation 2.1 and commanded at the motors to compensate for the dynamics of the robot.

### 2.2.2 Software Algorithms

Control software for the robot was written in Matlab Real Time Workshop and XPC-Target (Mathworks Inc, Natick MA, USA). Two approaches for calculating the real-time angular velocities and accelerations were coded. A commonly used backward difference method was implemented. For velocity, the difference between the previous and current angles was divided by the sample period and filtered with a second-order Butterworth filter (cutoff frequency = 31 Hz). The cut-off frequency was empirically chosen by reducing it until chatter during robot

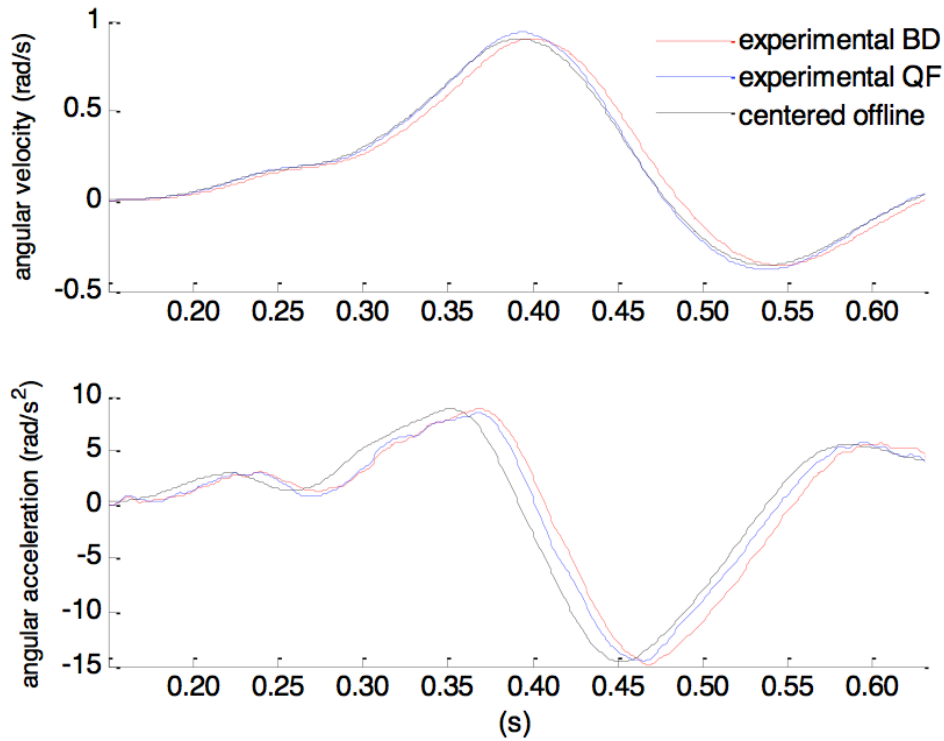


Figure 2.2. Comparison of time lag between backward derivative and improved derivative for a typical reaching movement on MIT-MANUS, at one of the two robot joints. Both are from the same experimental position data. The blue dotted lines show velocity and acceleration of improved derivative. The red solid lines show velocity and acceleration of backward derivative. The black dashed lines show the true offline approximation (central divided derivative was used to compute offline velocity and offline acceleration without time lag). Time counter is understood as the time count of reaching movement progress, based on a fixed step of 0.001 seconds.

operation was not perceptible. A similar procedure on the velocity profile was used to calculate accelerations.

A second approach was developed based on estimating the quadratic curve that best fits the most recent portion of angle data, and calculating the current derivative directly from the quadratic equation. The effects of noise are minimized by the fitting procedure, eliminating the need for any further filtering of the signal. The equations that define the best quadratic curve were derived using standard least squares methods. Briefly, we fit a portion of the angle profile to the following equation:

$$u = a_0 + a_1 t + a_2 t^2 \quad (2.2)$$

$a_0$ ,  $a_1$ , and  $a_2$  are the quadratic coefficients, and  $u$  is the angle profile. For  $n$  time points, the least squares criteria requires minimizing the following quantity:

$$\begin{aligned} \sum_{i=1}^n e_i^2 &= \sum_{i=1}^n (u_{i,measured} - u_{i,model})^2 \\ &= \sum_{i=1}^n (u_i - a_0 - a_1 t_i - a_2 t_i^2)^2 \end{aligned} \quad (2.3)$$

Setting the partial derivatives with respect to  $a_0$ ,  $a_1$  and  $a_2$  to zero, yields the following three algebraic equations:

$$\begin{aligned} (n)a_0 + \left(\sum_{i=1}^n t_i\right)a_1 + \left(\sum_{i=1}^n t_i^2\right)a_2 &= \sum_{i=1}^n u_i \\ \left(\sum_{i=1}^n t_i\right)a_0 + \left(\sum_{i=1}^n t_i^2\right)a_1 + \left(\sum_{i=1}^n t_i^3\right)a_2 &= \sum_{i=1}^n t_i u_i \\ \left(\sum_{i=1}^n t_i^2\right)a_0 + \left(\sum_{i=1}^n t_i^3\right)a_1 + \left(\sum_{i=1}^n t_i^4\right)a_2 &= \sum_{i=1}^n t_i^2 u_i \end{aligned} \quad (2.4)$$

Solving these three equations simultaneously for  $a_0$ ,  $a_1$  and  $a_2$  defines the best fit quadratic curve. If the data is well fit to a quadratic over short time periods, the method might outperform the backward difference method by reducing the delay caused by the filtering needed in this method.

We examined the performance of the real-time quadratic fit method by comparing it to an offline calculation of derivative that used a central difference algorithm followed by forward and backward low pass Butterworth filtering. This offline method should theoretically have no delay relative to the actual derivative. As the number of sample points ( $n$ ) used in the quadratic fit method increased, the delay between the quadratic fit and offline derivatives became larger. This increasing delay was caused by the fact that as  $n$  increases, data from time points further in the past can affect the current derivative estimate. As  $n$  was decreased, delay decreased, but noise in the estimated derivative increased, eventually leading to first chatter and then instability in the robot. We empirically determined that 35 and 55 ms were the minimum periods that could be used for velocity and acceleration calculations without introducing chatter or instabilities in the control.

To show the intrinsic reason why this improved derivative could bring benefits to the robot performance, the experimental position of the robot handle was used to calculate offline true velocity and offline true acceleration. Numerical central divided derivative was used for the offline approximation, followed by a special filter to produce no time lag. These offline terms were compared with the experimental velocity and acceleration (Figure 2.2). In addition, by this comparison, we came up with thirty-five points for velocity regression and fifty-five points for

acceleration regression to meet both criteria, a best match between theory and experiment and stability of movements. By moving the robot handle in multi-direction at multi-speed and either smooth or pulled; the first criteria was tested by comparison between experimental velocity and acceleration and theoretic ones; and the second criteria was checked by observing the handle stability with the emphasis on safety for patient, to guarantee that the movement is not vibrated or strongly pulled.

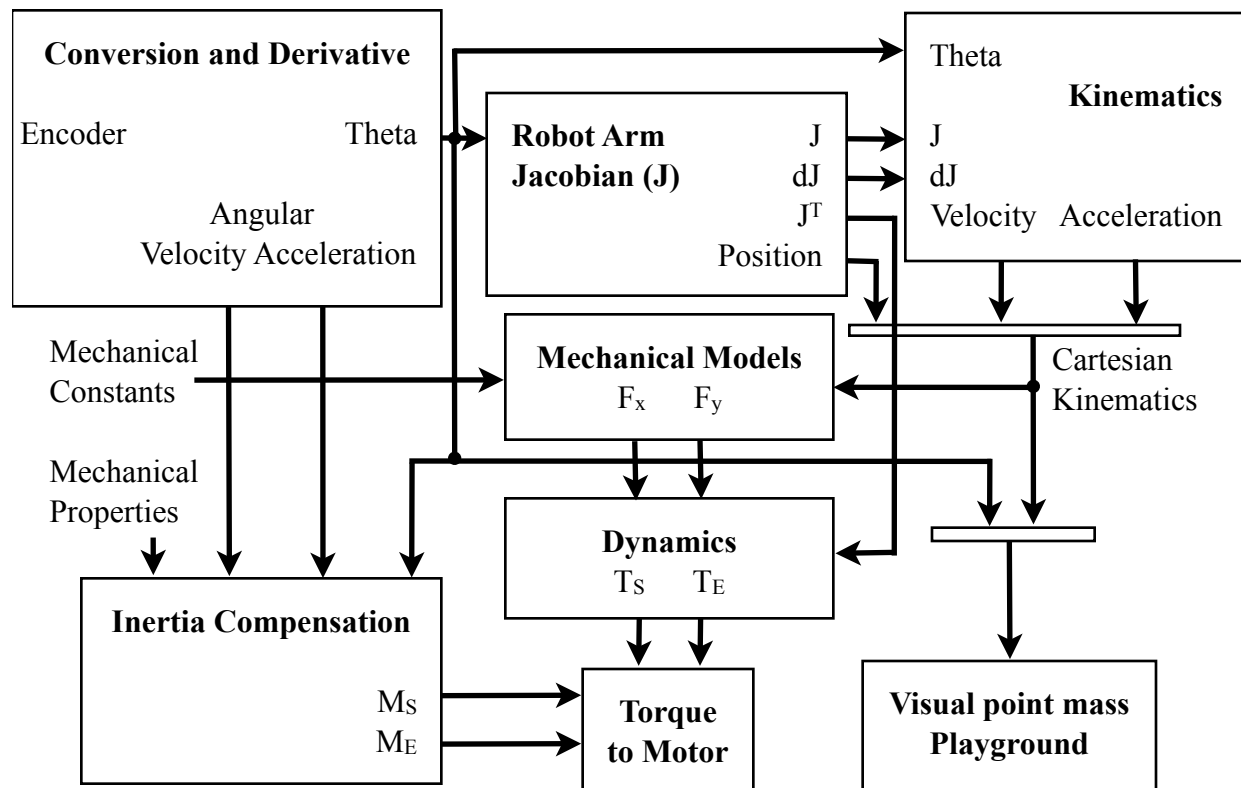


Figure 2.3. Block diagram of the robot arm control program including the developed inertia compensation with improved real-time derivative.



With backward derivative (BD) shown in red solid lines in Figure 2.2, the experimental velocity and acceleration are in time lag compared to the true offline during a typical ballistic reaching movement. In contrast, with quadratic fit derivative (QF) shown in blue solid lines, the experimental velocity is the same as the true offline. No time lag occurs in QF experimental velocity. The QF experimental acceleration is little delayed, in compared with the true offline. Note that the figure shows only angular velocity and acceleration along one joint of the robot, while the one in another joint is similar. Therefore, with backward derivative, velocity and acceleration have larger time lag as shown in the figure. While, with improved derivative, we have better shorter time lag, and this leads us to improved reaching movement performance.

The InMotion2 dynamics for inertia compensation and two methods of real-time digital differentiation have successfully implemented in parallel with other control functions of the robot (Figure 2.3).

### ***2.2.3 Experimental Setup and Tasks***

We tested the algorithms on the tasks most commonly used with this robot. Six healthy subjects (age=28±4 years) performed consecutive “center out” reaching movements under three test conditions: compensation with backward derivative (CBD), compensation with quadratic fit (CQF), and no compensation (NC). In the NC condition, zero torque is commanded to the robot motors throughout. All subjects provided informed consent to participate and all protocols were approved by the MedStar Health Research Institute Human Subjects IRB. Subjects were seated at the table and straps limited torso movement. The subjects grabbed the robot end effector and the forearm was strapped to a support trough. The order of the test conditions was randomized

across subjects. Not only the early phase (from start to the peak velocity) was analyzed, but the full phase (from start to settle-down) of “center out” reaching movements was explored.

Handle position and the desired targets were displayed on a screen in real-time as colored circles with diameters of 0.5 cm. Four targets (T1, T2, T3, T4) were located 15 cm radial to a start position that was oriented along the subject’s midline and located at the center of the robot’s workspace (Figure 2.1). The subject performed rapid reaching movements from the center position to the radial targets. Feedback was provided after each trial to encourage the subject to move at the targeted peak tangential velocity. Upon complete of each trial, the target circle changes its color: white signaled that the reaching movement fell within the desired peak velocity range; green and red signaled that movements were too slow or fast, respectively. Thirty familiarization trials were performed prior to beginning testing. Under each test condition, movements were performed to each target until at least 5 movements were performed at the desired peak velocities of 50 cm/s and 100 cm/s (acceptable speed window was 5 cm/s and 10 cm/s centered at the desired velocity, respectively). A force transducer (Gamma, ATI Industrial Automation, Inc., Apex, NC) mounted below the robot handle measured the interaction forces between the robot and subject. Force in the direction of the target was analyzed separately from force orthogonal to movement direction.

To explore the effects of higher speeds and movements further away from the center of the robot’s workspace, *another 6 healthy subjects* (age=24±4 years) performed a series of movements to targets 25 cm from the center start point at peak velocities of 150 cm/s. The algorithm may degrade as  $q_1$ - $q_2$  approaches 0 or 180 degrees (see Figure 2.1). Our movement

targets encompassed a  $q_1$ - $q_2$  range from 60 to 153 degrees. Physical stops limit  $q_1$ - $q_2$  between 29 and 162 deg. Considering overshoot, the actual  $q_1$ - $q_2$  angle often came very close to the max angle of 162 deg. We were unable to test closer to the min  $q_1$ - $q_2$  angle due to limited human arm range of motion.

In rapid movements, the inertia of the handle and sensor results in a non-zero force signal in the sensor that is unrelated to the interaction forces between the robot and subject. To eliminate these effects, additional trials were performed where the subject grabbed the robot below the force transducer (which is mounted beneath the handle) and performed 10 movements to each radial target at the 3 speeds. In this case, the force sensor signals were only due to the inertia of the sensor and handle. The sensor signals were ensemble averaged across these 10 trials and used to bias all other force data collected when the handle was held properly. Peak-to-peak tangential forces in these bias trials increased from 1.1 N at the 50 cm/s speed up to 4.7 N at 150 cm/s.

#### ***2.2.4 Statistical Analysis***

Several metrics were calculated, including peak-to-peak interaction forces, peak velocity, peak acceleration, peak deceleration, peak overshoot and peak lateral displacement from a straight line between start and end points. Interaction forces were separated into a tangential component that was directed along the direction of the desired movement and a normal component. For each test condition and movement direction, metrics were first averaged across repeated trials. For each metric, a repeated measures ANOVA was performed using within-subject factors of movement direction (T1,T2,T3,T4) and test condition (NC, CBD, CQF). A

Table 2.1. Kinematics

<b>metric</b>	<b>NC</b>	<b>CBD</b>	<b>CQF</b>
<b>peak velocity (cm/s)</b>			
50 cm/s	50.7±0.4	50.5±0.4	50.0±0.2
100 cm/s	99.5±1.0	99.3±0.8	99.3±1.2
150 cm/s	151.0±0.5	150.5±0.7	150.6±0.6
<b>peak acceleration (cm/s<sup>2</sup>)</b>			
50 cm/s	287±17	289±17	274±17
100 cm/s	919±76	940±75	886±78
150 cm/s	1404±44	1330±41	1317±59
<b>peak deceleration (cm/s<sup>2</sup>)</b>			
50 cm/s	-252±11	-259±6	-244±6
100 cm/s	-882±59	-927±57	-872±69
150 cm/s	-1315±59	-1518±45*	-1320±50#
<b>peak lateral deviation (cm)</b>			
50 cm/s	0.6±0.1	0.6±0.1	0.6±0.1
100 cm/s	0.9±0.1	0.8±0.1	0.8±0.1
150 cm/s	1.7±0.3	2.0±0.3	2.0±0.4
<b>peak overshoot (cm)</b>			
50 cm/s	1.1±0.2	0.6±0.1*	0.6±0.1*
100 cm/s	2.3±0.7	1.8±0.6*	1.5±0.5*
150 cm/s	1.7±0.3	1.8±0.4	1.3±0.3

NC (no compensation)

CBD (compensation with backward difference derivative)

CQF (compensation with quadratic fit derivative)

\* - significantly different from NC,  $p < 0.05$

# - significant difference between CQF and CBD,  $p < 0.05$

significant effect of test condition was further investigated with a series of paired t-tests (NC vs. CQF, NC vs. CBD, CBD vs. CQF). A Bonferoni correction was applied on these t-tests.

## 2.3 Results

The movement kinematics were similar across the three test conditions (Table 2.1). Repeated measures ANOVA found that test condition was not a significant factor in peak velocity ( $p>0.2$ ), peak acceleration ( $p>0.08$ ) and peak lateral deviation ( $p>0.3$ ). Peak deceleration was not different across conditions at 50 and 100 cm/s ( $p>0.3$ ), but test condition was a significant factor at 150 cm/s ( $p=0.011$ ). We further investigated this effect by first averaging across movement directions and then comparing conditions with paired t-tests. At 150 cm/s, deceleration was slightly higher in the CBD condition compared to the other conditions ( $p<0.03$ ). Test condition was also a significant factor in peak overshoot at both 50 cm/s ( $p=0.005$ ) and 100 cm/s ( $p=0.033$ ). Paired t-tests found this was due to significantly larger overshoot in the NC condition compared to both compensation conditions ( $p<0.04$ ).

Table 2.2 shows the peak-to-peak forces tangential and normal to the movement direction. At all speeds, ANOVA reported that forces varied significantly across test condition and movement direction ( $p<0.001$ ). Subsequent investigation into these effects with t-tests found the tangential peak-to-peak forces in the compensation conditions (CQF, CBD) were statistically smaller than in the NC condition in all target directions and speeds ( $p<0.001$ ). CQF tangential forces were smaller than CBD forces in the T1 and T3 directions at all speeds ( $p<0.004$ ). Additionally CQF forces were smaller than CBD forces in T2 and T4 directions at 150 cm/s

Table 2.2. Peak-to-Peak Interaction Forces

Target	Tangential (N)			Normal (N)		
	NC	CBD	CQF	NC	CBD	CQF
<b>peak velocity of 50 cm/s</b>						
T1	13.7±0.4	6.8±0.2*	5.3±0.2* #	5.2±0.1	3.9±0.1*	3.8±0.1*
T2	7.2±0.2	5.0±0.1*	5.1±0.1*	3.3±0.2	2.8±0.1	2.9±0.1
T3	11.9±0.4	5.0±0.1*	3.8±0.1* #	4.7±0.2	4.0±0.1*	3.8±0.1*
T4	7.0±0.1	5.4±0.1*	5.3±0.1*	4.7±0.2	3.0±0.1*	2.8±0.1* #
<b>peak velocity of 100 cm/s</b>						
T1	45.8±1.6	22.8±0.7*	17.1±0.7* #	11.0±0.7	6.8±0.4*	6.2±0.3*
T2	19.4±0.7	12.9±0.9*	12.0±0.7*	5.9±0.6	5.2±0.6	5.1±0.8
T3	44.0±2.4	21.0±0.9*	15.8±0.5* #	8.0±0.6	5.2±0.3*	4.8±0.3* #
T4	15.8±0.6	10.6±0.9*	9.9±0.2*	15.3±1.6	10.9±1.0*	8.4±0.9* #
<b>peak velocity of 150 cm/s</b>						
T1	68.9±1.9	31.8±0.4*	24.4±0.7* #	21.8±0.8	12.5±0.5*	11.3±0.4*
T2	29.6±1.8	17.7±0.5*	15.2±0.5* #	11.4±0.7	8.9±0.5*	6.7±0.5*
T3	69.5±2.4	35.7±0.8*	26.7±0.8* #	15.3±1.0	12.0±1.6	8.0±0.7* #
T4	19.1±0.3	12.3±0.6*	11.2±0.5* #	26.2±1.5	15.1±0.6*	13.8±0.3*

NC (no compensation)

CBD (compensation with backward difference derivative)

CQF (compensation with quadratic fit derivative)

\* - significantly different from NC,  $p < 0.05$

# - significant difference between CQF and CBD,  $p < 0.05$

( $p < 0.05$ ). Tangential forces were generally much larger in the T1 and T3 directions compared with the T2 and T4 directions. In these high impedance directions, tangential forces were reduced 64% by CQF (averaged across T1 and T3 directions, and then across the three speeds).

Normal forces in CQF and CBD conditions were statistically smaller than NC in the majority of cases, and there were several instances of CQF normal forces being statistically smaller than CBD forces (Table 2.2). In the T4 direction, normal forces were unusually large compared to tangential forces, especially at 150 cm/s where normal forces were larger in magnitude than tangential forces.

To quantify anisotropy, we examined the peak-to-peak forces in the 4 directions and calculated the difference between the max and min value (Table 2.3). In the tangential forces, compensation algorithms produced significant reductions in anisotropy at all speeds ( $p < 0.001$ ). CQF anisotropy was smaller than CBD anisotropy at 100 and 150 cm/s ( $p < 0.015$ ). Averaged

Table 2.3. Anisotropy in peak-to-peak forces (maximum force minus minimum force)

Speed (cm/s)	Tangential (N)			Normal (N)		
	NC	CBD	CQF	NC	CBD	CQF
50	6.8±0.4	1.9±0.2*	1.7±0.1*	1.9±0.2	1.2±0.1*	1.2±0.1*
100	31.1±1.6	12.2±0.9*	7.3±0.7* #	9.6±1.1	6.2±0.7*	4.2±0.7*
150	52.7±2.0	23.4±0.8*	15.6±1.0* #	15.1±1.5	7.8±0.6*	7.4±0.4*

NC (no compensation)

CBD (compensation with backward difference derivative)

CQF (compensation with quadratic fit derivative)

\* - significantly different from NC,  $p < 0.05$

# - significant difference between CQF and CBD,  $p < 0.05$

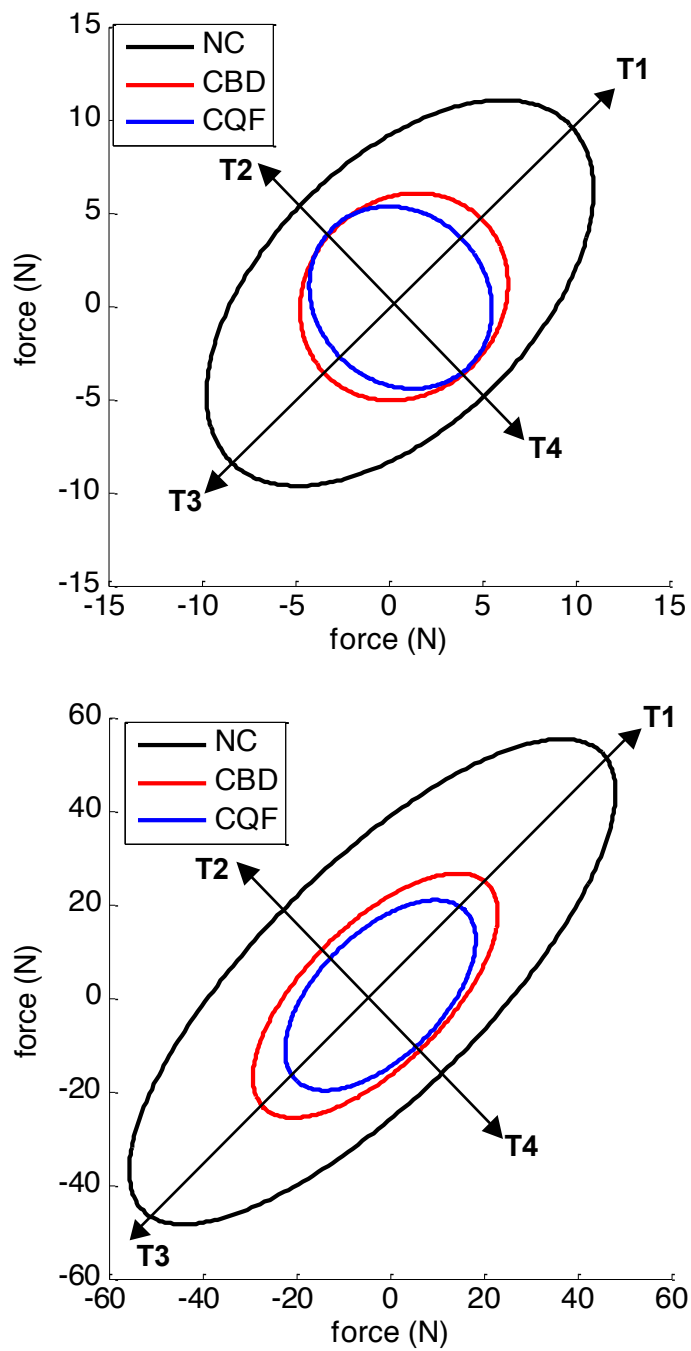


Figure 2.4. Ellipses representing tangential peak-to-peak interaction forces in each movement direction at 50 cm/s (top panel) and 150 cm/s (bottom panel). Data is the average of all subjects.



across speeds, anisotropy reductions were 74% for CQF and 62% for CBD. The anisotropy reductions are illustrated in Figure 2.4, where tangential forces in the 4 directions were fit to ellipses. In the normal forces, compensation also significantly reduced anisotropy at all speeds ( $p < 0.04$ ).

We investigated the source of the impedance anisotropy. In the NC condition at 150 cm/s, impedance in the T1-T3 directions was *3 times* larger than in T2-T4 directions (Table 2.2). Using equation 2.1, we simulated the handle forces needed to make these movements using minimum jerk end point trajectories (Flash and Hogan 1985). The simulated tangential forces showed the same anisotropy we observed experimentally. Examination of the algebraic relationship between tangential force and joint torque revealed that as a first approximation, tangential force in T1 directed movements was mostly related to  $\tau_1$ , while tangential force in T4 directed movements was related to  $\tau_2$ . We then compared the peak-to-peak magnitudes of several simulation variables.  $\tau_1$  in the T1 directed movements was *3 times* larger than  $\tau_2$  in T4 directed movements. Further examination of the terms that contributed to these torques showed that this 3-fold difference was due to  $I_{11}\ddot{q}_1$  in T1 movements being approximately *3 times* larger than  $I_{22}\ddot{q}_2$  in T4 movements. This difference was driven in part by  $\ddot{q}_1$  being *1.35 times* larger than  $\ddot{q}_2$ . However the major cause of the endpoint anisotropy was the imbalance in the diagonal terms in the inertia matrix;  $I_{11}$  was *2.1 times* larger than  $I_{22}$ .

Full phase force profiles of reaching movements are shown in Figure 2.5 for one of the three ranges of speed. In addition, the impedance anisotropy critical ratio at highest speed 150 cm/s, calculated as the max divided by the min value, was reduced from 3.64 in NC to 2.90 in CBD and to 2.38 in CQF.

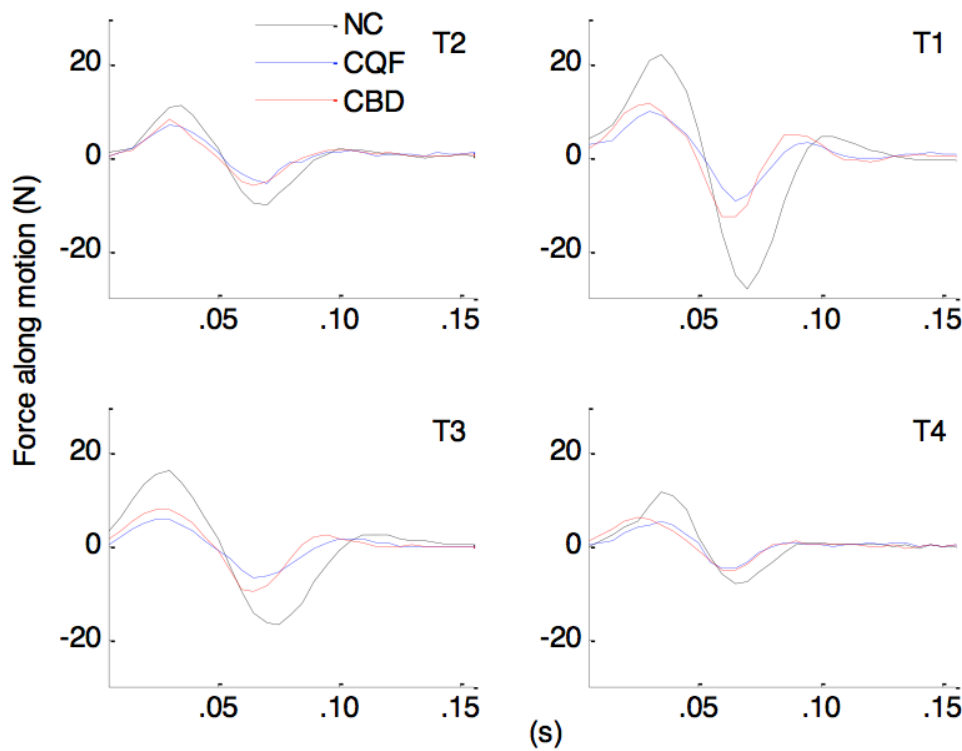


Figure 2.5. Force profile along the direction of motion for 4 targets at 100 cm/s.

## 2.4 Discussions

Both CQF and CBD clearly reduced interaction forces and anisotropy compared to the NC condition. CQF outperformed CBD in terms of reducing tangential forces in the high-impedance directions (T1,T3). Normal forces were also reduced by the compensation

algorithms, with CQF outperforming CBD in several cases. Both compensation algorithms substantially reduced tangential force anisotropy, with the CQF mode outperforming CBD at the 100 and 150 cm/s speeds. We tested movements throughout the workspace and at speeds that approached the maximum the tested subjects could produce in the robot for 15-25 cm movements. No instabilities were observed. However we did notice an instability when the robot end effector was in physical contact with a solid surface.

Peak overshoot was larger in the NC condition than the compensation conditions, which is consistent with a larger effective inertia without compensation. However, the kinematics were very consistent across the three test conditions; statistical analysis found no differences between test conditions in peak velocity or peak acceleration (Table 2.1). Therefore, its unlikely kinematic differences played a major factor in the large reductions in tangential interaction forces with the compensation algorithms. The normal forces were larger than expected. These forces were caused by slightly misdirected movements and the dynamics of the robot linkage. However, the amount of lateral deviation did not vary across conditions (Table 2.1), so the reductions in normal forces by the compensation algorithms (Table 2.2) were also not due to different kinematics. The large differences in interaction forces (Table 2.2), coupled with very similar kinematics across conditions, suggests subjects adapted to the different forces applied by the robot across conditions to satisfy the task goal of reaching the target with the desired peak velocity.

Previous studies have shown that even low impedance robots perturb the natural movement patterns subjects would normally produce in a zero-impedance robot (Campolo et al.,

2009; Tagliamonte et al., 2011). Despite this fact, the majority of motor adaptation studies using robots similar to the InMotion2 do not employ active algorithms that reduce impedance.

Furthermore, our data highlights that without active compensation, the impedance is highly anisotropic. Without compensation, the anisotropy in tangential force increased from 6.8 N at 50 cm/s to 52.7 N at 150 cm/s. Normal force anisotropy was also substantial; 1.9 N at 50 cm/s and increasing to 15.1 N at 150 cm/s. Anisotropy was significantly reduced by the compensation algorithms at all speeds.

CQF has similarities to a method employed previously to decrease interaction forces in the InMotion2 robot (Tagliamonte et al., 2009). The main difference between these two approaches is the method used for compensating for inertial forces. CQF calculates inertial terms directly, making use of real-time acceleration estimates from the quadratic fit derivative algorithm, while the Tagliamonte method uses a force feedback control term derived from force sensor measurements. CQF has the advantage of not requiring a force sensor, which can introduce noise as well as error in estimates of interaction forces with rapid movements (we measured force sensor values as large as 4.7 N when interaction forces were zero). On the other hand, the Tagliamonte method is robust to changes or errors in estimates of the mechanical properties of the robot and is not susceptible to errors in real-time calculation of accelerations. It is difficult to compare performance of the two algorithms, as Taglimonte et al. only tested 1 subject and did not evaluate their algorithm systematically across different movement speeds and directions. Nevertheless, they reported a 54% reduction in peak interaction force, which is

comparable to the 64% reduction in tangential peak-to-peak forces we observed in the high impedance directions.

## 2.5 Conclusions

We tested the inertia compensation algorithm with two different methods for real-time calculation of derivatives, a novel CQF and the commonly used CBD. Twelve subjects performed a series of point-to-point movements under three conditions (CQF, CBD, NC), in different directions at peak speeds of 50, 100 and 150 cm/s. With NC, tangential peak-to-peak forces were as large as 69 N in certain directions at the 150 cm/s speed. Both CQF and CBD significantly reduced tangential forces in all directions and speeds. CQF outperformed CBD in the directions with highest intrinsic impedance, reducing tangential forces by 64% in these directions. Compensation also significantly reduced forces normal to the movement direction, with CQF again outperforming CBD in several cases. Anisotropy was assessed by the range of tangential peak-to-peak forces across movement directions. In NC, anisotropy was as high as 52.7 N at the 150 cm/s speed, but an average anisotropy reduction of 74% was achieved with CQF. The CQF method can significantly reduce impedance and anisotropy in this class of robot. We have successfully implemented CQF in parallel with force field environments such those used commonly in motor adaptation experiments (de Xivry et al., 2012). No modifications to the compensation or the force field algorithms were necessary when combining them and no instabilities were introduced.

Once the technical hurdle of compensating for the intrinsic dynamics of the robot was solved, we could move to the next chapter of systemic evaluation of bilateral interlimb coupling in stroke survivors using precise robotic loading of the non-paretic limb.

## **Chapter 3**

### **Interlimb Coupling with Bimanual Loading on Subjects with Stroke**

#### **3.1 Introduction**

It has been suggested that interlimb coupling effects could be exploited to enhance affected arm motor performance, particularly if additional loading is applied to the non-paretic arm. Previous studies have shown that bilateral symmetric movements can improve performance of the paretic limb. However, there are conflicting results as to whether loading the non-paretic limb further improves paretic limb performance. To address this question and to investigate the effects of different types of loading conditions, unilateral and bilateral movements, including a variety of different non-paretic limb loading conditions, were performed by individuals with chronic stroke. The reaching task was carried out by having the participant's forearms placed on two arm supports which can be rotated around the elbow joint. Beginning with elbows flexed and forearms pronated, participants were required to reach out in the horizontal plane to tap a target bar with a range of about 40-degree elbow extension. In unilateral movements, either paretic (P) or non-paretic (NP) arm does reaching. In unloaded bilateral movements, both arms move to reach the target without loading on NP arm. In loaded bilateral movements, NP arm is perturbed with an external force by a robot. Three force patterns (inertia, spring, and constant) were applied at two levels of loading (10% and 20% of maximum voluntary contraction). In total, nine

movement conditions were employed: unilateral paretic and non-paretic; bilateral unloaded; bilateral inertia, spring and constant each at two levels of 10 and 20. Kinematic signals of the hands were recorded while EMG signals of triceps and biceps were collected. Standard clinical tests were done prior to the movement experiments.

Interlimb coupling, a fundamental aspect of neuromotor system organization, concerns how movements are coordinated across limbs (Kelso et al., 1979), can be exploited to improve paretic limb movement performance during bilateral limb tasks. The motor system shows a strong tendency toward symmetry and synchronicity between the paretic and non-paretic limb movements when bimanual limb tasks are executed (Goodman et al., 1983; Kelso et al., 1979; Serrien et al., 1999). Therefore, when poststroke hemiparesis patients perform bilateral upper extremity movements, the overflow (interhemispheric sharing of normal movement commands from the undamaged hemisphere) of movement characteristics from the non-paretic limb to the paretic arm could be activated to improve the paretic limb performance (Mudie et al., 2000; Lewis et al., 2004).

Prior studies of upper extremity interlimb coupling in participants with poststroke hemiparesis have had contradictory findings. Several studies (Rice et al., 2001; Lewis et al., 2004; Chang et al., 2006; Messier et al., 2006; Kilbreath et al., 2006) showed that interlimb coupling after stroke results in no improvement in paretic arm motor performance and disrupted non-paretic arm motor performance. One study (Ustinova et al., 2006) showed that interlimb coupling was impaired and unstable for rhythmical tasks. However, four studies have shown that some subjects with hemiparesis show improvement in paretic arm performance during bilateral



tasks compared with unilateral tasks (Cunningham et al., 2002; Rice et al., 2004; Harris-Love et al., 2005; Waller et al., 2006).

With six stroke subjects under constraints of discrete single-joint reaching tasks with arm support, bilateral reaching tasks were reported to facilitate movement of the paretic arm in three subjects (Cunningham et al., 2002). Interlimb coupling conducted on thirty-two stroke subjects with box-reaching task at fastest possible speed was retained, mutual beneficial early in the movement (increased paretic and non-paretic peak acceleration) and then, later in the movement, the non-paretic will affect the paretic arm more than vice versa, resulting in higher paretic peak velocity while unchanged non-paretic peak velocity (Harris-Love et al., 2005); interlimb coupling effects can be used to produce an immediate improvement in paretic arm reaching performance. Bilateral upper limb movements of sixteen ischemic stroke subjects improved paretic movement time and movement speed compared to sequential movement (Waller et al., 2006).

Inertial adding to the non-paretic arm during bilateral reaching task brought benefits to paretic arm performance in 5 of 6 subjects studied (Cunningham et al., 2002) in terms of movement smoothness. In contrast, non-paretic arm loading showed no positive effect in coffee mug moving tasks of 20 stroke subjects (Chang et al., 2006). Loading on the non-paretic arm during bilateral reaching did not result in further improvement in paretic arm performance (Harris-Love et al., 2005). We observed that inertial loading might have limitation. One possible explanation could be that non-paretic loading increases the inertia, but not the resistance against the movement, and therefore has no influence when moving at constant or slow velocity.

In our study, a robotic device is used to precisely control the loading of the non-paretic arm. InMotion2 arm robot is applied for rehabilitation and motor learning, with key feature of low end-point impedance, made possible by using direct drive motors to apply force at the end effector. In numerous applications, the end point force generated by the robot is controlled open loop and depended only on the Jacobian. In these cases, low intrinsic impedance is critical for both rehabilitation and motor learning applications to fulfill accuracy in the perturbed forces. Most studies neglect the magnitude and anisotropy of intrinsic impedance of the robot but we

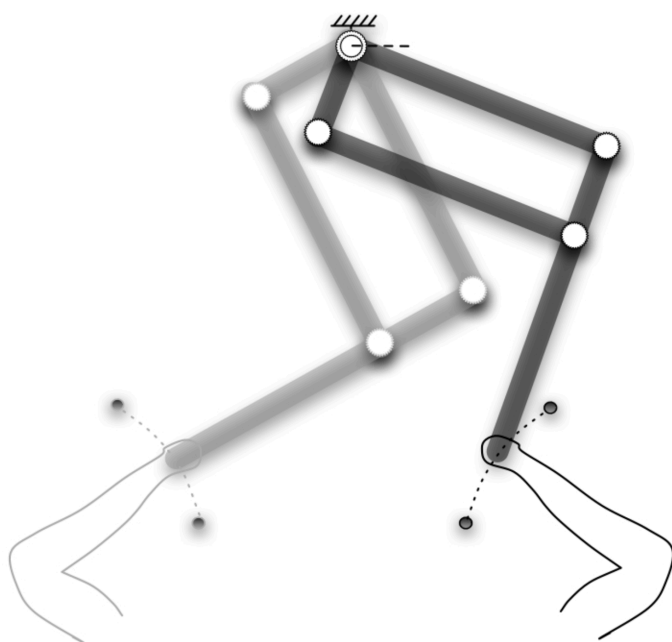


Figure 3.1. InMotion2 robot is used for either left or right non-paretic limb loading. Grasping the robot handle and move on identical paths, subjects with left and right non-paretic hand receive the force perturbation by the robot differently without inertial compensation algorithm.

saw this issue must be solved (Nguyen and Lum, 2013). We included our developed algorithm of intrinsic inertia compensation into our loading paradigm to produce more precise force perturbation (Figure 3.1).

The hypotheses of the study chapter were to investigate the influence of novel loading for reaching movements in chronic stroke. We further investigated the parameters that influence the interlimb coupling. With a novel set-up of tasks, conditions, apparatus, measurement and analysis methods, we would (1) verify that the interlimb coupling effects is retained after stroke (2) verify that bilateral reaching task would result in improved paretic arm performance, compared with unilateral reaching task (3) explore that during bilateral reaching task, paretic arm reaching performance would be improved by resistive loading on the non-paretic arm (4) discover that during bilateral reaching task, the overflow from non-paretic arm to paretic arm would be clearly evident during the resistive loading conditions, and (5) explore the potential effects on the paretic limb performance by inertial loading on non-paretic limb using a robotic device.

## **3.2 Methodology**

### ***3.2.1 Participants***

A target of ten stroke subjects was selected for this study with, age ranged from 18 to 70 years, first-ever ischemic or hemorrhagic stroke (if possible, to exclude hemorrhagic strokes in order to avoid confounding factors due to different origin), occurred at least 6 months prior to the study, having at least 45 degrees of elbow flexion and 100 degrees of elbow extension, both left

and right hemisphere involvement and the subject signs a written informed consent. Fugl-Meyer Assessment (FMA) of motor recovery after stroke for upper extremities will be recorded prior to the test by a standard clinical test.

Subjects will be excluded from this study, if they fulfill one of the following criteria: 1) excessive spasticity of the affected arm (mAS is greater than 3), 2) any serious medical or psychiatric illness, 3) women known to be pregnant or lactating, 4) inability to communicate effectively with the examiner such that the validity of the patient's data could be compromised, 5) serious cognitive deficits (defined as equivalent to a Mini-mental state exam score of 24 or less). Subjects will not be excluded on the basis of race, ethnicity or sex.

We started with 19 subjects, then 6 individuals were withdrawn from the study by the investigators because they had little or no stroke-related motor impairment of the paretic arm, as measured by the FMA (score of 55 or higher) or by comparison with the non-paretic arm and healthy subjects. In these individuals, it would be nearly impossible to determine if the different experimental conditions produced any improvement in paretic arm motor behavior. These individuals were therefore excluded from further testing.

Hence, there were 13 subjects going through full data collection (Table 3.1). Each subject received a distinct random order of nine task conditions. These conditions of the task will be specified in the next part. Separately to these participants with stroke, we had two healthy subjects both with right limb dominant joined the study in which the order of task conditions was also random and contrasted from all executed orders so far.

Comparison of kinematic result between healthy subjects with those eleven, we decided to remove two stroke subjects (marked as \* in Table 3.1) from further analysis because their velocity and acceleration are equivalent to those from healthy participants. We then withdrew one more subject due to its poor performance of movement (marked as \*\* in Table 3.1). This subject moved too slow that we could not set a criteria to extract the time at which they started to move and when they actually stopped. Likewise, analysis including this subject will not help us able to detect the onset/target reaching time of movement for any other subjects. Eventually, there were 10 subjects embraced into analysis towards the study outcome, with FMA  $34 \pm 16$  (range within 15-53), six left and four right neuro-hemiparesis deficit.

Table 3.1. List of participants; \* removed from final analysis, \*\* could not be analyzed. Task condition order is random and unique for each subject. FMA is Fugl-Meyer Assessment score for upper extremity of motor recovery after stroke. Side of hemisphere tells the weak paretic side.

Subject	FMA	Side of Hemisphere	Random Task Condition Order									
			UniP	UniNP	Bil	Inertia 10	Inertia 20	Spring 10	Spring 20	Const 10	Const 20	
S01	50	R	5	7	8	3	6	1	2	4	9	
S02	47	L	9	2	1	6	8	3	7	4	5	
S03	37	L	1	6	8	4	2	9	5	7	3	
S04	52	L	4	3	7	9	8	1	5	6	2	
S05	53	L	7	2	8	5	6	9	4	1	3	
S06	15	R	9	1	7	3	8	2	5	4	6	
S07	24	R	7	2	1	8	3	4	5	6	9	
S08	22	L	1	8	3	2	9	5	4	7	6	
S09	22	R	3	4	5	1	9	8	6	7	2	
S10	16	L	8	3	4	9	1	2	7	5	6	
* S11	40	R	6	3	7	8	5	1	2	4	9	
* S12	40	R	1	5	2	9	4	3	7	6	8	
** S13	14	L	4	7	5	3	9	1	2	8	6	

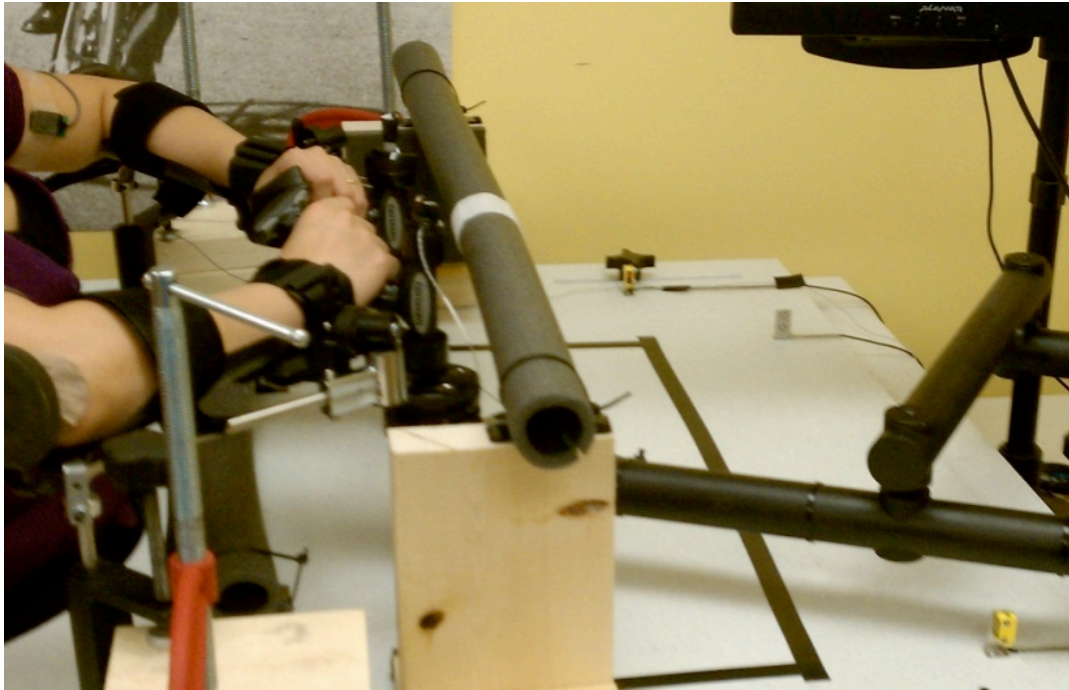


Figure 3.2. Reaching task apparatus (InMotion2 robot is loading the right arm).

### ***3.2.2 Procedure***

The participants were seated on an adjustable-height chair, trapped by a seat-belt to avoid their body leaning forward during the task, at a table one meter width with elbow being kept to be not moving but can be rotated about its joint. Both limbs are assisted by the light forearm supports and restricted on a horizontal plane, while two hands grasp identical ergonomic handles. The front of the body is aligned with elbow fixed point. The elbows are flexed at 45 degrees at starting position. Paretic, Non-Paretic (UniP, UniNP) unilateral, P-free-NP-free bilateral (Bil), P-free-NP-inertial (Inertia) bimanual, P-free-NP-spring-resistive (Spring) bimanual, P-free-NP-constant-resistive (Const) bimanual reaching task conditions are to perform a discrete, 40 degree elbow extension to move their hand(s) to reach a fixed soft rectangular bar at destination (Figure

3.2). Therefore, only single joint or proximal muscles are involved in this reaching task. In all bilateral conditions, both hands are required to reach the bar at the same time and not too hard. In bilateral resistive reaching with loading (Spring and Constant), the InMotion2 robot is used to add a resistive load to non-paretic limb. While in Inertia loading condition, a virtual mass (using the above robot) will be attached to the non-paretic hand.

With spring loading, the force is against the direction of the movement and is position-dependent. The farther the forearm moves, the larger the force it receives. Therefore, in the spring loading condition, the highest movement resistance occurs towards the end of the movement. With constant loading, the force is held constant at all times, and is applied in a direction that is against movement. With inertia loading, the applied force is acceleration-dependent, making the task similar to having the subject holding and moving a weight. Inertial loading causes the resistance against NP arm movement to be varied, such that the highest force against movement occurs at the early part of the task. In total, nine movement conditions were employed: unilateral paretic and non-paretic; bilateral unloaded; bilateral inertia, spring and constant each at two levels of 10 and 20.

Reaching task was familiarized by 5 trials at self-selected comfortable speed, and then 10 trials for each condition are executed at fastest possible speed. A visual and audible signal was used to denote start and stop of reaching. We instructed the subject orally and visually during familiarization and keep it repeated at beginning of each reaching condition. Sensing markers attached at the hand were used with tracking system to record kinematic data while EMGs of triceps and biceps were collected. Once familiarization was done, non-paretic limb's maximal

voluntary force (MVC) was recorded prior to the test by a handy load-cell. Two levels of 10% and 20% of MVC were specified within bilateral conditions with loading, abbreviated as Inertia 10, Inertia 20, Spring 10, Spring 20, Const 10 and Const 20.

In Spring, a resistive force profile was used with the interaction force equal the product of  $K$  and  $x$ . In which  $x$  is the position of the hand counted from start position, while  $K$  is assigned accordingly with the two loading levels such that the product of  $K$  and maximum  $x$  is equal 10% or 20% of MVC of the non-paretic arm. The  $x$  reaches its maximum when the hand hits the target bar. In Inertia, a mass will be assigned to restrain the applied force which is the product of mass and peak acceleration equal either 10% or 20% of MVC. The peak acceleration is achieved during familiarization trials of bilateral no-load. Another resistive loading being used is Constant, where the applied force was always held constant during the movement, at two levels of 10% and 20% of MVC. The three equations have minus sign to reflect the fact that the applied force by the robot is in reverse to the movement direction; or in the case of Inertia, it models the inertia of a moving weight by producing the opposite applied force to the direction of movement when the acceleration is positive (increased speed) and vice versa.

$$\begin{aligned}
 F_{\text{Spring}} &= -Kx \\
 F_{\text{Inertia}} &= -ma \\
 F_{\text{Constant}} &= -F \\
 Kx_{\text{max}} &= ma_{\text{peak}} = F = 10\% \text{ or } 20\% \text{ MVC}
 \end{aligned}
 \tag{3.1}$$



Totally, there were nine task conditions conducted on each subject, with random order (Table 3.1). Note that where it is requested, the force generated by the robot is always opposite to the hand movement direction. Safety sensors were mounted on the table to cut down power of the robot as far as the generated force drove the arm out of a secure space.

All above protocol procedure including the informed consents was in place in the institutional review board (IRB) process for approval to make sure that the human participants are protected from physical and psychological harm.

### ***3.2.3 Data Acquisition and Analysis***

Subjects got used with the reaching task by 5 trials for each of two unilateral, and one bilateral without loading. Total of 15 familiarization trials were played with each participant. To warrant 10 valid good trials per each of nine conditions, we actually solicited 15 trials which included in it 5 for spare. The key outcome parameters for the study, listed in priority from high to low, were proposed as hand movement peak-to-peak and peak acceleration (PPA, PA), peak velocity (PV), time-to-peak (T2P), peak triceps RMS EMG (P-EMG), cross correlation (xCorr) lag, movement time (MT) and number of movement units (NMU). Additionally, the minor parameters are xCorr function value at zero-lag, xCorr function max value, and reaching-onset temporal synchrony, reaching-complete temporal synchrony, reaching-onset spatial synchrony.

The Northern Digital Inc's Optotrak Certus tracking system was used to collect kinematic data, at sampling rate of 100 Hz with six infrared sensors mounted on two hands, two elbows and target-bar's two ends. Bagnoli EMG system, with four double-differential surface electrodes adhered to triceps and biceps, used to measure force exerted by shoulder and elbow muscles of

both upper limbs, configured at sampling rate of 300 Hz, was made by Delsys. Each channel has a selected gain set to a factor of 1000. At these gain settings, the main amplifier unit filters the signals to a bandwidth between 20 Hz and 450 Hz and checks for excessive amounts of line interference as well as channel clipping due to over-amplified signals. The presence of these errors is signaled via yellow LEDs and through a user-enabled audio buzzer alarm. The two systems of Optotrak and Bagnoli were synchronized through system control and data acquisition units.

Cartesian position data recorded by tracking system was later transformed into angular joint displacement with centers of joints at the left or right elbow. The inverse homogeneous transform matrix was calculated as below and used to transform data from frame 0 (origin at the camera mounted on the wall) to frame 1 (origin right on the desk surface) and eventually to joint displacement.

$$\begin{bmatrix} -0.4 & 0.5 & 0.7 & 1,725.5 \\ 0 & -0.8 & 0.6 & 1,108 \\ 0.9 & 0.3 & 0.3 & 941 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned}
 Elbow_{Left} &= \begin{bmatrix} -296.9 & 18.1 & 47 \end{bmatrix} \\
 Elbow_{Right} &= \begin{bmatrix} 428.8 & 10.4 & 47 \end{bmatrix} \\
 Theta_{Left} &= \text{atan2}(y_{Left} - Elbow_{Left}(2), x_{Left} - Elbow_{Left}(1)) \\
 Theta_{Right} &= \text{atan2}(y_{Right} - Elbow_{Right}(2), |x_{Right} - Elbow_{Right}(1)|)
 \end{aligned} \tag{3.2}$$

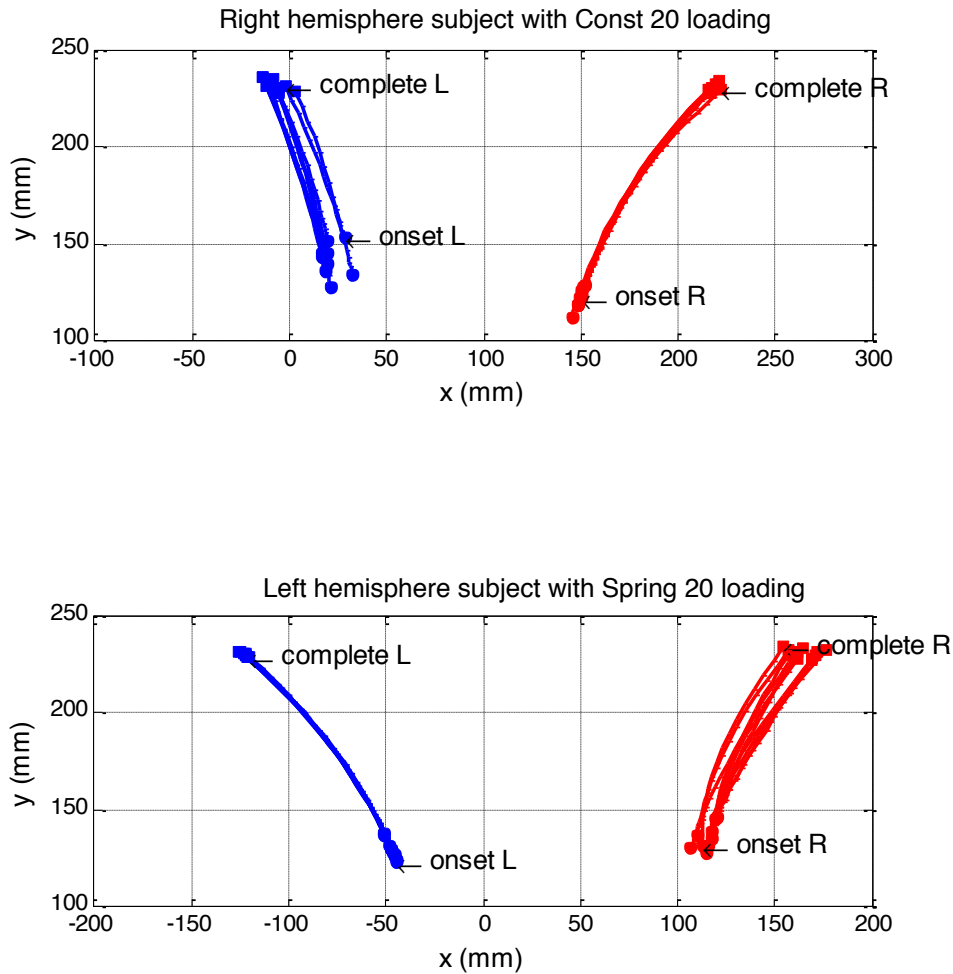


Figure 3.3. Typical pathways of the bilateral movements of two hands of the whole group of 9 subjects. Onset mark corresponds to the physical start position, while complete mark corresponds to the physical target position. Top row shows a right hemisphere subject (S01), with the weak right arm is free to move while the left good arm grasping the robot's handle during a Constant loading. Bottom row shows a left paretic subject (S05) in a Spring loading. Data is plotted with 10 trials per a task condition.

These joint displacement were used to search the time at which the reaching task was started (onset time) and the time at which the subject hit the target bar to complete the reaching task (complete time) (Figure 3.3). We used two windows, window one consists of 50 data samples, and window two consists of 5 samples, standing right next to the end of window one. These two windows ran through all joint data. Onset time was defined at the end time point of window one when average value of window two exceeds the mean baseline of window one (determined over 50 samples) by 5 standard deviations (of window one). While a target plane perpendicular to the movement plane was used to detect when the reaching movement crossed the target to define complete time. Subtraction of onset from complete time gave the movement time (MT) of reaching, in seconds.

Discrete parts of kinematic data could be missed during the collection due to physical blocking to infrared communication by the collectors. Two people helped with technical and instructional works. The missing data was processed by interpolation to keep time index in continuous to ease accurate calculation of velocity and acceleration, but eventually those missing parts were ignored from final synthesis. A trial containing this missing data was totally neglected from further analysis.

Peak velocity (PV), in meter per second, is defined as the peak of tangential velocity which is calculated as a vector sum of velocity in x and y directions of horizontal movement. Acceleration is the derivative of tangential velocity. Peak acceleration (PA), in meter per second square, is the peak value of acceleration in early phase of reaching movement which is counted from onset to the time at PV. Peak deceleration (PD), in negative sign, is the peak of acceleration

in breaking phase of movement which is counted from the time at PV to complete time. Peak-to-peak acceleration (PPA) is a subtraction of PD from PA with the count of its sign. Time-to-peak (T2P), in seconds, is specified as period from onset to the time at PV. Smoothness of movement is counted by number of velocity peaks between onset and complete time marks, denoted as

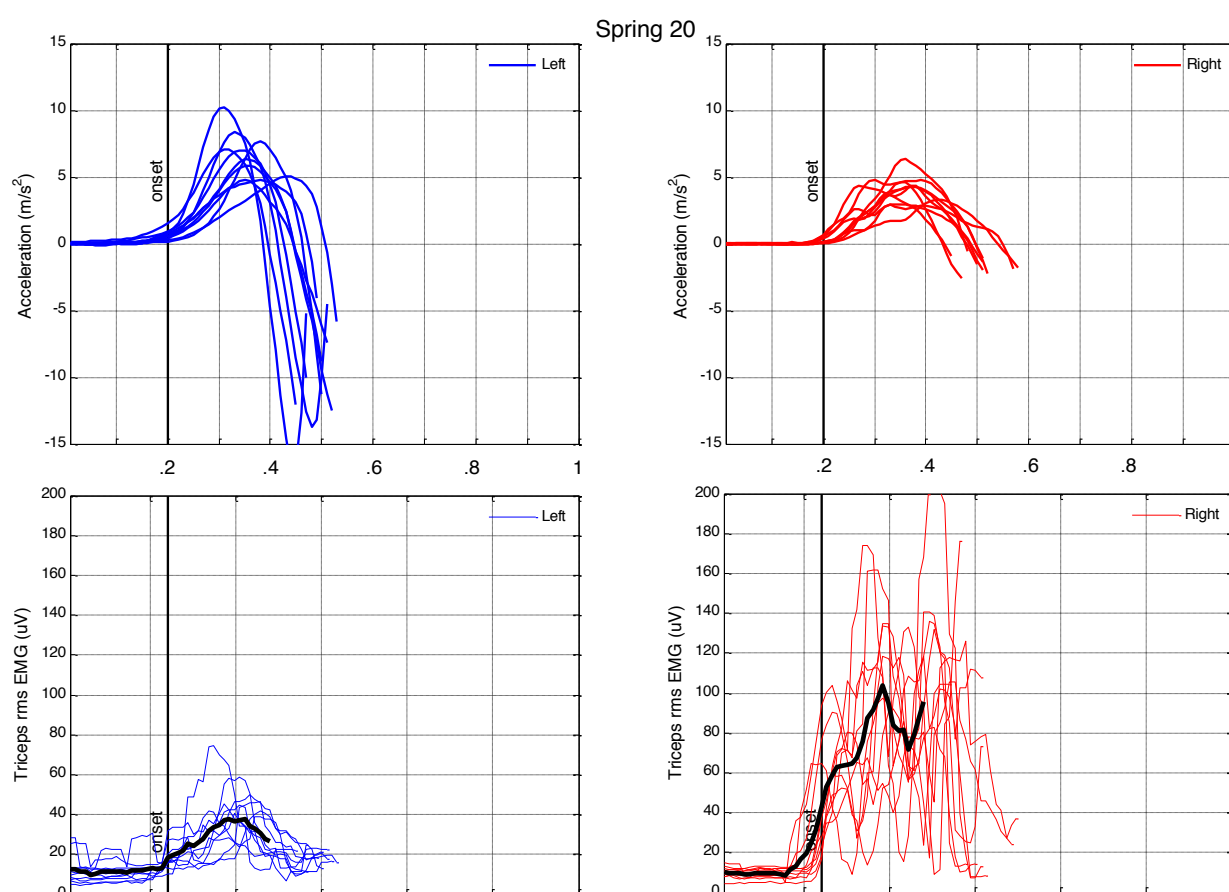


Figure 3.4. Triceps RMS EMG graphs in alignment with acceleration of two arms in a task condition (Spring 20 shown) of a single left hemisphere subject (S08) within the whole group. The bold black line of EMG is the average of ten trials and is used just for demonstration of how we verify the onset. EMG of each trial is used to check the onset of that trial only. Only peak RMS EMG is extracted from each trial to be used in the ten trial average.

number of movement units (NMU). To avoid collecting tiny jerks, a peak is considered as a valid one with preset minimum distance apart from another peak about one-seventh of the movement time.

The raw electromyography (EMG) signal, at sample  $t$  devoted as  $m(t)$ , in micro-volts (uV), was converted to root mean square (RMS) form using the norm and a selected smoothing window length  $n$  of 15 samples. The RMS EMG is used to majorly extract its peak (Figure 3.4). It also helps verify the onset of movements which was calculated earlier using kinematic approach.

$$m_{rms}(t) = \frac{1}{\sqrt{n}} \|m(t)\|_2 = \sqrt{\frac{1}{n} \sum_{i=t-n+1}^t m^2(i)} \quad (3.3)$$

Movement synchrony between two limbs was defined by distance-versus-time cross correlation (xCorr), but only within early phase of movement, from onset to peak velocity. Its correlation function's number of time lags, denoting the quantity of lags from its zero-lag, told us how well synchronous the two limbs is. The smaller this value the better synchrony it reflects. A zero value is with perfect synchronous movement. The cross correlation is alike in nature to the convolution of two functions. The input vectors denoted as  $f$  and  $g$  are  $N$  length vectors ( $N > 1$ ). If  $f$  and  $g$  are not the same length, the shorter vector is zero-padded to the length of the longer vector. The output vector  $c$  returns the cross correlation sequence in a length of  $2N-1$  vector.

$$\hat{R}_{fg}(m) = \begin{cases} \frac{1}{N} \sum_{n=0}^{N-m-1} f(n+m)g^*(n) & m \geq 0 \\ \hat{R}_{gf}^*(-m) & m < 0 \end{cases} \quad (3.4)$$

$$c(m) = \hat{R}_{fg}(m - N) \quad m = 1, 2, \dots, 2N - 1$$

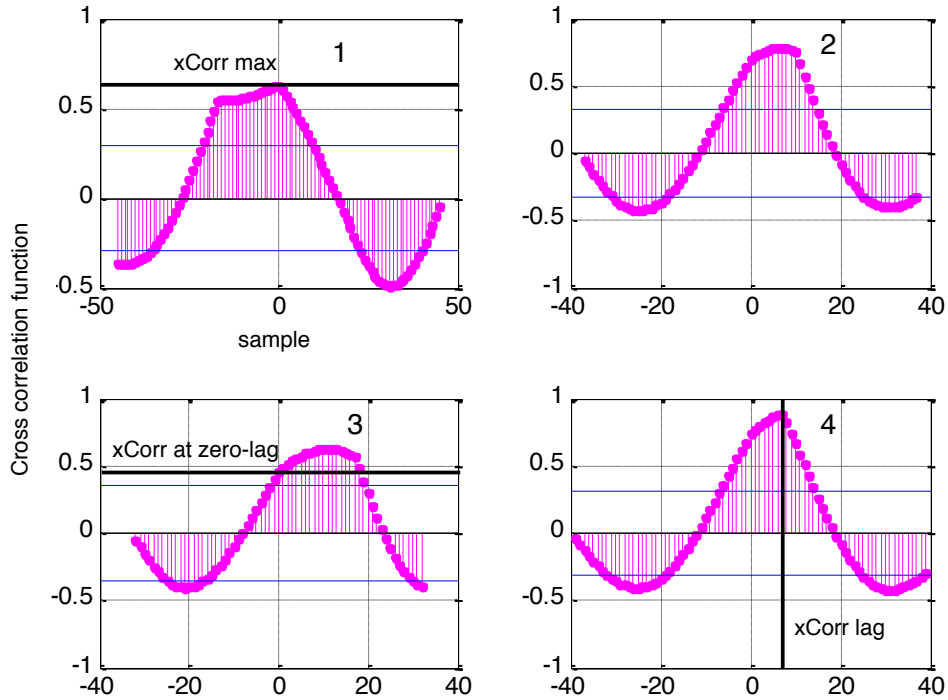


Figure 3.5. Typical cross correlation (xCorr) graphs of four trials in a bilateral task condition of a single left parietic subject (S10) within the whole group. Each sample is .01-s long. Cross correlation function (XCF) is normalized to one. The xCorr lag is defined as number of samples, including its sign, counted from the origin to the location of the peak of XCF. The xCorr at zero-lag is the value of XCF at origin. The xCorr max is the highest value of XCF.

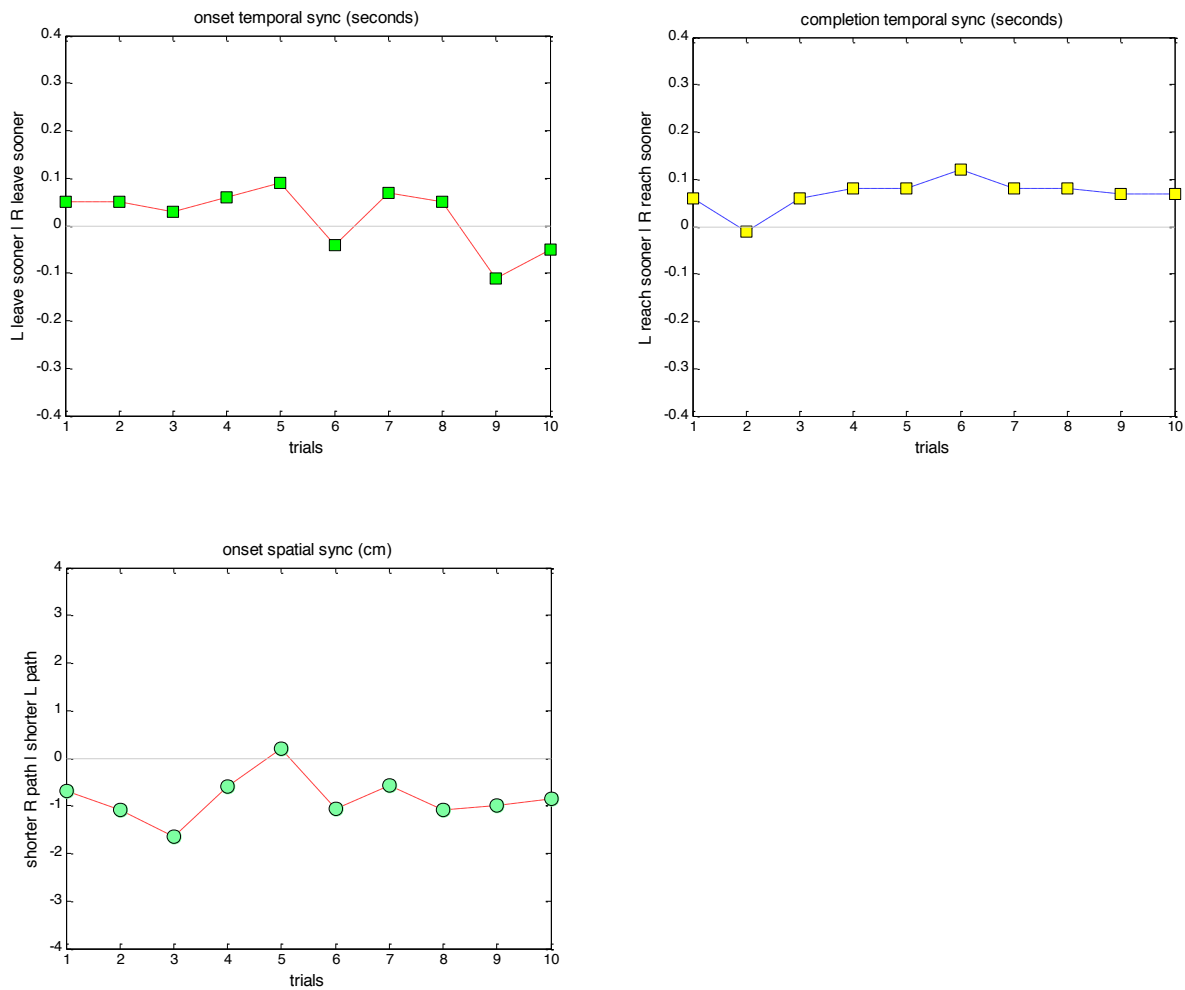


Figure 3.6. Temporal onset synchrony, temporal complete synchrony and spatial onset synchrony in a task condition (Inertia 20 shown) of a single left paretic subject (S02) within the whole group.

Five minor parameters were considered, though they all show no significant results. The xCorr function value at zero lag and xCorr function max (Figure 3.5) are as additional measure to judge the synchrony between limbs. The bigger these metrics the better synchronous



movements between limbs. The temporal synchrony of reaching-onset is the timely difference, in seconds, of movement starts between two arms. A zero value of onset temporal synchrony says two arms started to move at the same time (Figure 3.6). The temporal synchrony of reaching-complete is the timely difference, in seconds, of movement completeness between two arms. This parameter tells us which hand of the left or right sides hitting the target bar first. Finally, the reaching-onset spatial synchrony is the distance between left and right hand starting position at the onsets.

All above parameters were averaged over 10 trials within each task condition and from group of multiple subjects. All profile plots, if shown, of the parameters were normalized at their onsets (in this case, it is simply the alignment of all onsets to a common point). Statistical model use was repeated measure (RM) analysis of variance (ANOVA) and paired samples t-test running on IBM's SPSS software at significant p-value of .05.

### 3.3 Results

Kinematic parameters of the movements such as peak-to-peak, peak acceleration and peak velocity are considered as the first outcomes we want to get. The subject S10 is unique and in a different trend compared to other nine subjects (S01 to S09). It is the only subject that has all bilateral conditions producing better PV, PA and PPA compared to unilateral. In addition, based on the bilateral-unilateral difference (BUD) which is equal to the average of PV in all bilateral conditions minus PV of unilateral, we consider S10 as outlier in statistical analysis. After determining the first ( $Q_1 = -.144$ ) and the third ( $Q_3 = -.114$ ) quartiles and the interquartile

Table 3.2. Statistical analysis p-value outcome for the whole group of 9 subjects. The p-value in dark orange is noticed as significant, in light yellow is tendency towards significant, in x-mark as it is not significant and in dash-mark as either not applicable or no need to apply. PPA peak-to-peak acceleration, PA peak-acceleration, PV peak-velocity, P-EMG (T-RMS) peak-EMG of triceps in RMS form, T2P time-to-peak, NMU number of movement units, MT movement time and xCorr Lag cross-correlation time lag.

Significance	PPA	PA	PV	P-EMG (T-RMS)	T2P	NMU	MT	xCorr Lag	Name of conditions considered
1 3x2 ANOVA: interaction									
3 condition x 2 load level	0.005	0.004	0.02	x	0.002	0.07	0.01	-	Const, Inertia, Spring
2 2x2 ANOVA:									
load level	0.02	0.04	0.07	x	0.03	x	0.1	-	Const, Spring at 2 load levels
interaction	x	x	x	x	0.01	0.08	x	-	Const, Spring at 2 load levels
3 one-way ANOVA: condition									
no-load & 3 resistive 20%	0.02	0.03	0.09	-	0.07	-	x	-	Bil, Const 20, Inertia 20, Spring 20
no-load & 3 resistive 10%	x	x	x	-	x	-	x	-	Bil, Const 10, Inertia 10, Spring 10
no-load & 2 resistive 20%	0.04	x	x	-	0.1	-	x	-	Bil, Const 20, Spring 20
no-load & 2 resistive 10%	x	x	x	-	x	-	x	-	Bil, Const 10, Spring 10
3 resistive 20%	0.02	0.02	0.07	-	x	-	0.06	-	Const 20, Inertia 20, Spring 20
3 resistive 10%	0.05	x	x	-	x	-	x	-	Const 10, Inertia 10, Spring 10
4 t-test									
pair Bil-Const20	0.04	0.1	x	-	0.09	-	x	x	
pair Bil-Inertia20	x	x	x	-	x	-	x	0.001	
pair Bil-Spring20	0.08	x	x	-	x	-	x	x	
pair Uni-Bil	0.06	0.04	0.03	-	0.09	-	0.09	-	
pair Bil-Const10	x	x	x	-	x	-	x	x	
pair Bil-Inertia10	x	x	x	-	x	-	x	0.13	
pair Bil-Spring10	x	x	x	-	x	-	x	x	

( $IQR = .03$ ) ranges (see outlier quartile figure in appendix) then we get the lower and upper

fences:

$$\begin{aligned}
 LF &= Q1 - 1.5(IQR) \\
 UF &= Q1 + 1.5(IQR)
 \end{aligned}
 \tag{3.5}$$

The resulted fences varies from  $-0.188$  to  $-0.069$  allows us to separate S10, which has its BUD mean of  $.184$  and standard deviation of  $.069$ , from the subject group for statistical analysis.

The group analysis is therefore applied on a group of 9 stroke subjects for 3 loading conditions (Inertia, Constant, Spring), along with two progressive loading levels. Result of three-way RM 3x2 ANOVA interaction of 3 loading condition x 2 load level is significant in PPA ( $p=.005$ ), PA ( $p=.004$ ) and PV ( $p=.02$ ) (Table 3.2). This significant result of interaction allowed us to further investigate the effect of loading conditions compared to bilateral no-load condition.

Regarding the load level, the next step of running two-way RM 2x2 ANOVA of 2 loading (Constant and Spring) conditions x 2 load levels results in significant PPA ( $p=.02$ ), PA ( $p=.04$ ) and tendency towards significant PV ( $p=.07$ ) (Figure 3.7). This says the higher load level (from 10% to 20% MVC) subjects received during Constant condition, the better improvement in PPA (from 9.66 to 12.36  $\text{m/s}^2$ ), PA (from 5.07 to 6.12  $\text{m/s}^2$ ), PV (from .64 to .72  $\text{m/s}$ ) they are beneficial. In the same trend, those numbers with Spring condition are PPA (from 8.70 to 11.82  $\text{m/s}^2$ ), PA (from 4.76 to 5.77  $\text{m/s}^2$ ) and PV (from .64 to .69  $\text{m/s}$ ). In contrast, Inertia loading did not exhibit improvement trend in acceleration or velocity by this progressive loading.

Furthermore, the direct comparison between Bilateral and either Constant or Spring condition would further support the role of these two resistive loading conditions. T-test between Bil (9.25  $\text{m/s}^2$ ) and Const 20 (12.36  $\text{m/s}^2$ ) gave significant PPA difference ( $p=.04$ ). Bil (9.25  $\text{m/s}^2$ ) differs from Spring 20 (11.82  $\text{m/s}^2$ ) with a tendency towards significant PPA t-test ( $p=.08$ ). In contrast, t-test between Bil (9.25  $\text{m/s}^2$ ) and Inertia 20 (9.73  $\text{m/s}^2$ ) did not return significance.

In addition, one-way RM ANOVA of 3 task conditions (Bil, Const 20 and Spring 20) pointed out the significant PPA difference ( $p=.04$ ) between these types of conditions. One-way

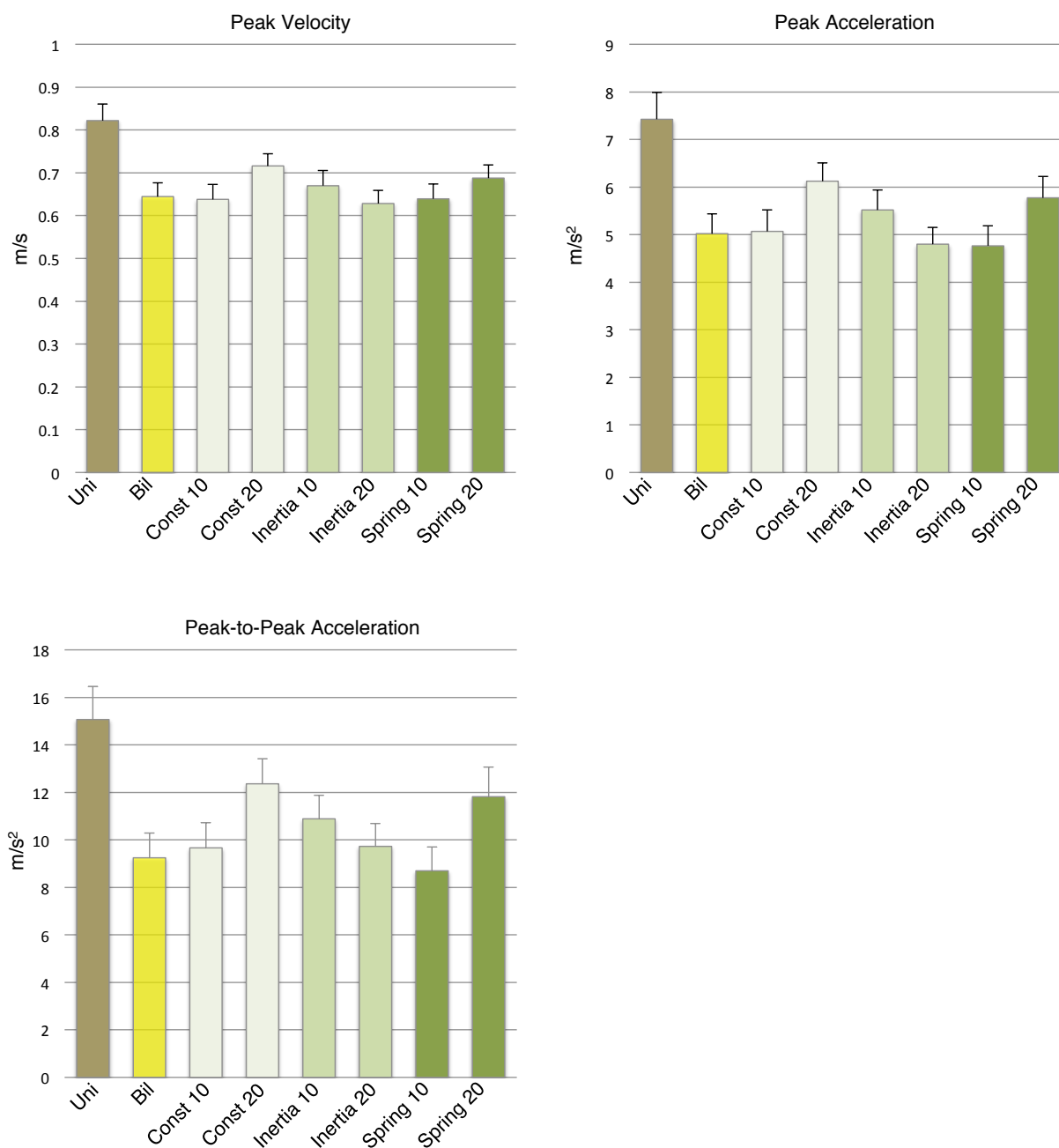


Figure 3.7. Bar plot of peak velocity and acceleration of the whole group of 9 subjects. For Constant and Spring loading task conditions, increasing load level from 10% to 20% (of MVC) resulted in higher peak-to-peak acceleration ( $p=.02$ ), higher peak acceleration ( $p=.04$ ) and a tendency towards higher peak velocity ( $p=.07$ ). Data is averaged across 9 subjects ( $\pm$ SE).

RM ANOVA of 3 task conditions (Const 20, Inertia 20 and Spring 20) showed the significant PPA difference ( $p=.02$ ) regarding the type of conditions. It is significant ( $p=.05$ ) even for one-way RM ANOVA of three lower loading conditions (Const 10, Inertia 10 and Spring 10).

With PA (Table 3.2), t-test comparison said that difference between Bil ( $5.02 \text{ m/s}^2$ ) and Const 20 ( $6.12 \text{ m/s}^2$ ) is just tendency towards significant ( $p=.1$ ). But, one-way RM ANOVA of three resistive conditions of Const 20 ( $6.12 \text{ m/s}^2$ ), Inertia 20 ( $4.80 \text{ m/s}^2$ ) and Spring 20 ( $5.77 \text{ m/s}^2$ ) exhibited the distinction among them ( $p=.02$ ). Besides, close-to-significant p-value of t-test and one-way RM ANOVA regarding PV can be seen from the same table.

One stroke subject with lowest movement (and FMA of 24, but not smallest score) compared to the others, showed big advantage of Constant loading (Figure 3.8). T-test running between Bil ( $1.48 \text{ m/s}^2$ ,  $.19 \text{ m/s}$ ) and Constant 20 ( $3.26 \text{ m/s}^2$ ,  $.34 \text{ m/s}$ ) delivered both PA ( $p<.001$ ) and PV ( $p<.001$ ) significant. As seen from the figure, peak velocity was great improved about double in Constant 20% loading compared to Bilateral no-load. Acceleration was improved even more than double. These rises in kinematic performance are about 200%. The profile graphs in the figure favor an intuitive look with velocity and acceleration waveforms.

Time to peak (T2P), counted from onset to the time at peak velocity, of paretic limb is another metric worth to be investigated. It determines how fast a subject speeds up their paretic arm movement from null to its maximal speed depending on different force patterns applied to healthy arm. Since there is a movement instruction remind per each task condition that the participant moves as fast as they can but not hitting the bar too strong. A shorter T2P may reflect

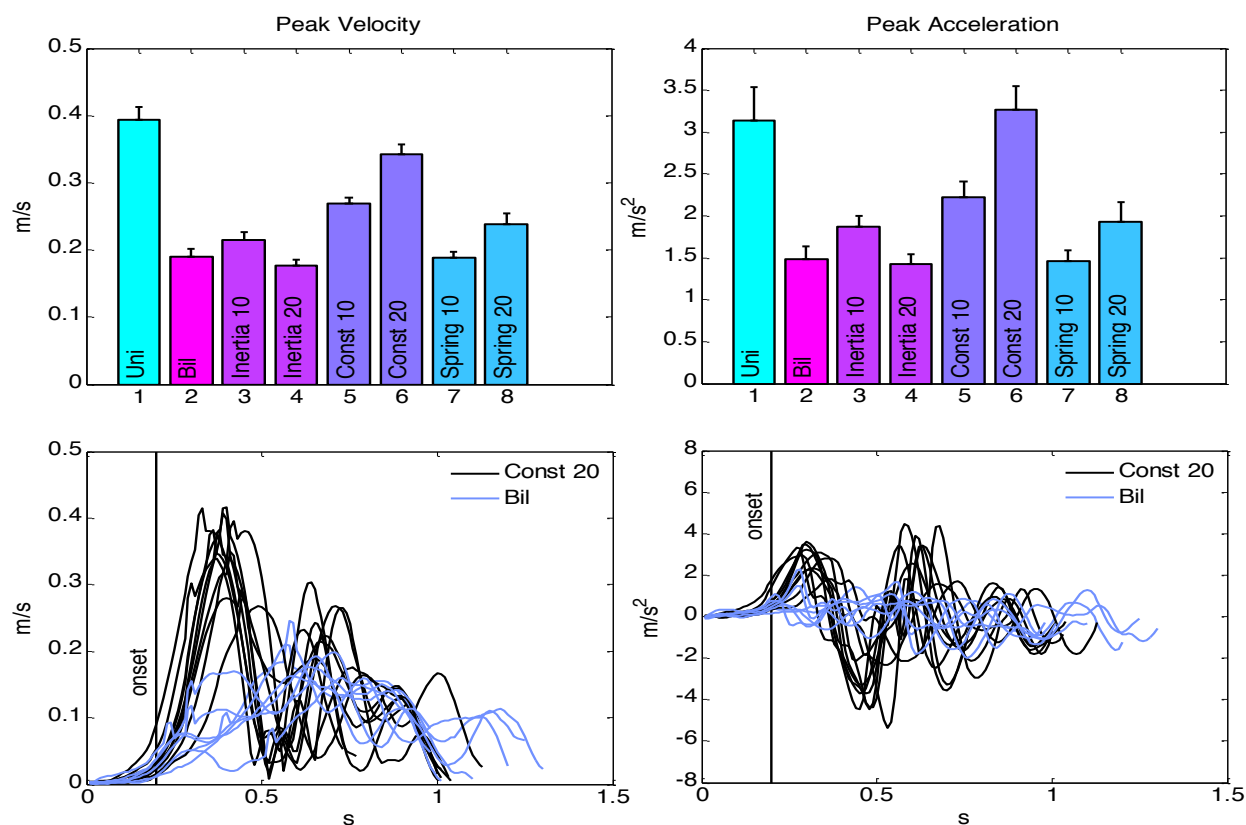


Figure 3.8. One low movement subject in bar & profile plots of velocity and acceleration. With the top row showing the peak value, and data is averaged across ten trials ( $\pm$ SE). The bottom row shows all ten trials to compare the profile graphs of the two task conditions, and data is normalized at the onset.

a better functioning feedforward mechanism appeared in that task condition. Figure 3.9 shows that T2P, in seconds, of Unilateral (.22) is better than Bilateral no-load (.26). Constant loading conveyed the largest T2P reduction in the paretic limb (.21 with Const 10, and .20 with Const 20), and Spring loading also helped lessen this parameter (.26 with Spring 10, and .23 with Spring 20) with significant ( $p=.03$  2x2 RM ANOVA) regarding the progressive loading level. However, the decrement of T2P in respect to loading level was not happened to Inertia. Though

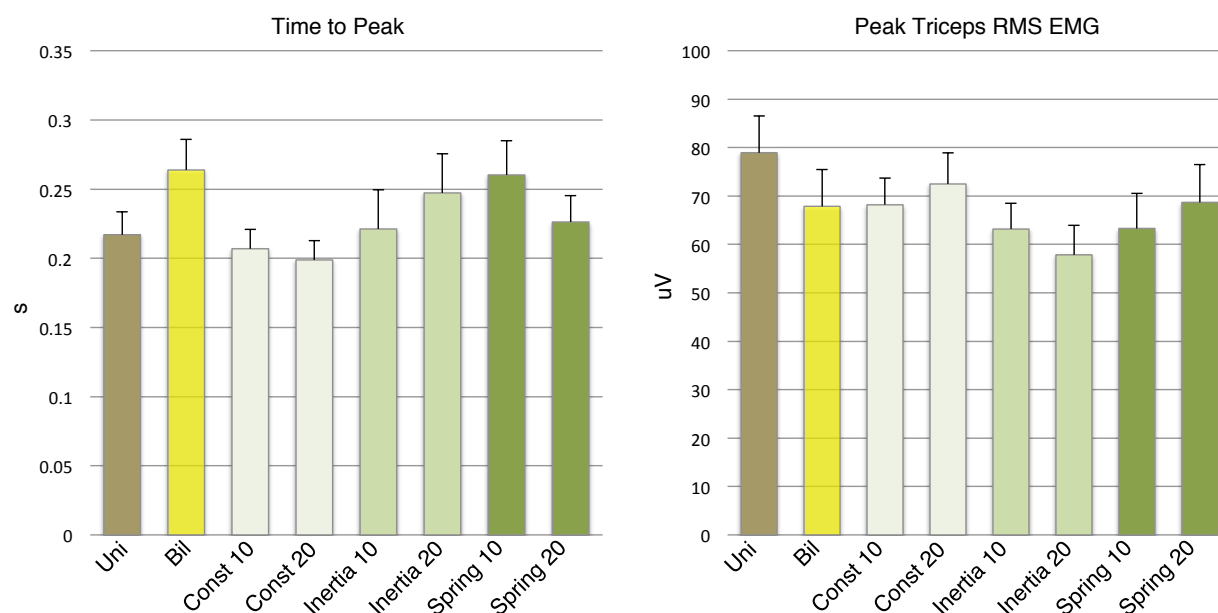


Figure 3.9. Time to Peak (T2P) and peak triceps RMS EMG (P-EMG) graphs of the whole group of 9 subjects. Data is averaged across 9 subjects ( $\pm$ SE). For Constant and Spring loading task conditions, increasing load level from 10% to 20% (of MVC) resulted in smaller T2P ( $p=.03$ ). T2P is best in bilateral conditions with Constant loading. P-EMG likely looks the same trend even though statistical run shows no significance.

in average, T2P was reduced from Bilateral no-load to Inertia (.23 seconds). The trend of T2P appears to be compatible to those of PPA, PA or PV discovered earlier.

Peak Triceps RMS EMG (P-EMG) measures electrical activity of the agonist muscle of the paretic limb during the reaching out of the movement. As in Figure 3.9, P-EMG for the whole group demonstrated the same tendency as it was with kinematics. Constant and Spring both brought benefit from its progressive loading, compared to Bilateral no-load. Nevertheless, our statistical run did not show significant from any t-test or RM ANOVA. This is perhaps caused by

our EMG system which we observed its electrodes not really in good contact with subject's skin surface at all time. Plus, we saw peak RMS EMG of biceps remained equally small for all conditions as it is just 21 uV (Figure 3.10). This might declare that the biceps did not contribute to the reaching-out movement at the loading levels of 10% and 20% MVC.

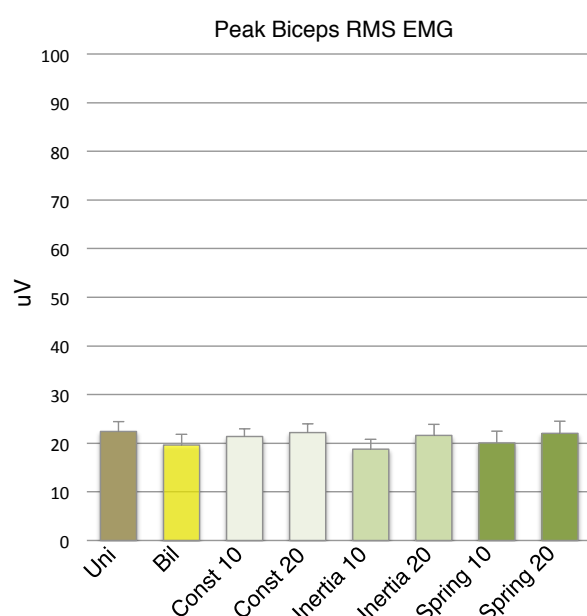


Figure 3.10. Peak biceps RMS EMG of the whole group of 9 subjects. Data is averaged across 9 subjects ( $\pm$ SE). All task conditions look like having equal value on the antagonist muscle.

Cross correlation time lag (xCorr lag) exposed interesting difference between bilateral conditions (Figure 3.11). With each sample is .01 seconds long, most of bilateral conditions with or without loading is having almost equal xCorr lag, in samples, i.e. 4.72 for Bil, 4.93 for Const 10, 5.07 for Const 20, 4.72 for Spring 10 and 5.30 for Spring 20. However, the xCorr lag of



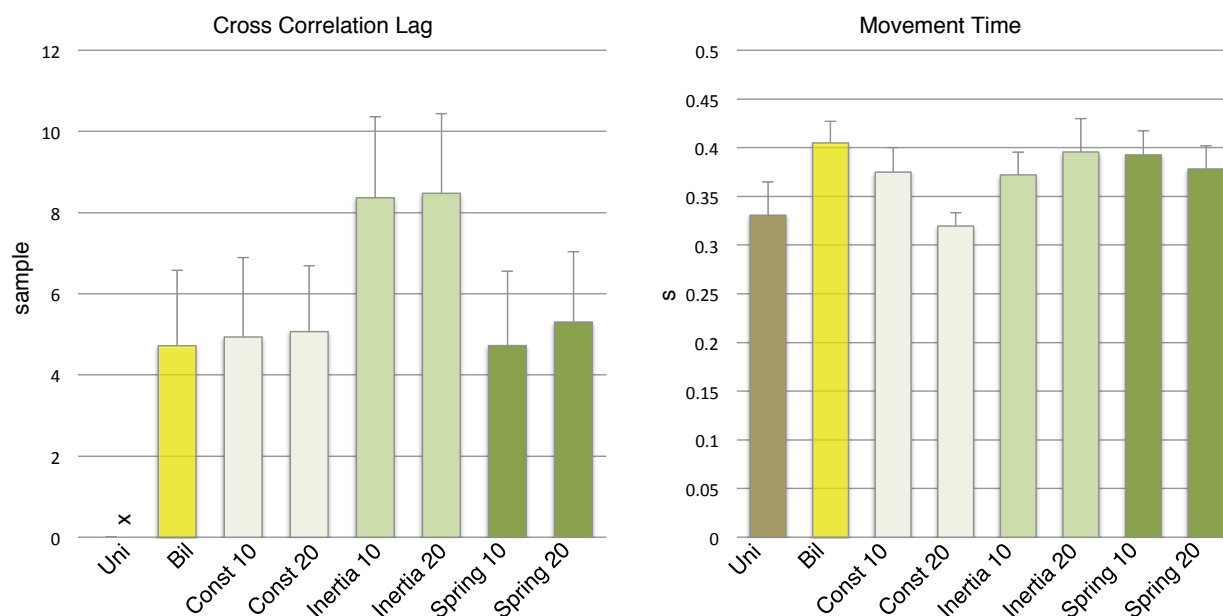


Figure 3.11. Cross correlation (xCorr) time lag and movement time (MT) graphs of the whole group of 9 subjects. XCorr lag describes how well the distance-vs-time of one hand correlated to another. Each sample is .01-s long. The xCorr lag reflects the worst correlation for Inertia loading compared to other bilateral conditions ( $p=.001$  with t-test between Bil and Inertia 20). The applied loading in Constant and Spring conditions does not effect the xCorr lag compared with Bilateral no-load. MT is in the same trend as velocity or acceleration behaves with statistical 3x2 ANOVA of 3 task conditions (Const, Inertia, Spring) x 2 loading levels (10 and 20% MVC) showing significant interaction ( $p=.01$ ). In average, xCorr lag varies from 13% to 23% percent of MT. Data is averaged across 9 subjects ( $\pm$ SE).

Inertia (8.37 and 8.48, at progressive loading levels) differs from the rest ( $p=.13$ ,  $p=.001$  respectively) following a represented Inertia-Bil t-test.

Scatter chart of xCorr lag versus velocity of all trials in all bilateral conditions with or without loading may report a trend of which the higher velocity the better (shorter) xCorr lag (Figure 3.12). The trend is remained the same either with total of 10 subjects or 12 subjects. The

10 subjects include the group of 9 subjects and 1 outlier (S10). The 12, furthermore, included the 2 removed stroke subjects which were considered as higher level of kinematics (as high as with the healthy). In the figure, the vertical solid black line represents the averaged peak velocity of all bilateral condition. The right side of the line has more points with lower lag compared to the left side. Since Figure 3.11 shows that Inertia loading has larger lag compared to other conditions, we classified the scatter graphs by having Inertia versus others (Figure 3.13). Despite this difference, the same trend of having lower lag at higher speed is remained in each of the charts.

As the last key metric, movement time (MT) show a significant interaction ( $p=.01$ ) from 3x2 RM ANOVA of 3 tasks (Const, Inertia, Spring) x 2 loading levels (10 and 20% MVC). MT exhibits tendency towards significant ( $p=.1$ ) respect to progressive loading from 2x2 RM ANOVA of 2 tasks (Const, Spring) x 2 loading levels. MT behaves similarly to the above three kinematic metrics. Average MT of all conditions is about .37 seconds, or equal to 37 samples. The xCorr time lag mentioned above varies from 13% (most of bilateral loading) to 23% (Inertia) of MT.

Within the group of 9 subjects, the one with lowest speed of movement gained most benefits from either Const or Spring compared to other conditions. T2P, in seconds, is high in Bil (.45), Spring (.46) and Inertia (.33) but low in Const (.18). P-EMG, in micro volts, is high in Spring (102) but lower in Bil (87), Inertia (88) and Const (89). MT, in seconds, is shortest with Const (.67) and longer with others, Bil (.80), Inertia (.77) and Spring (.83). Only xCorr lag is not distinct cross conditions.

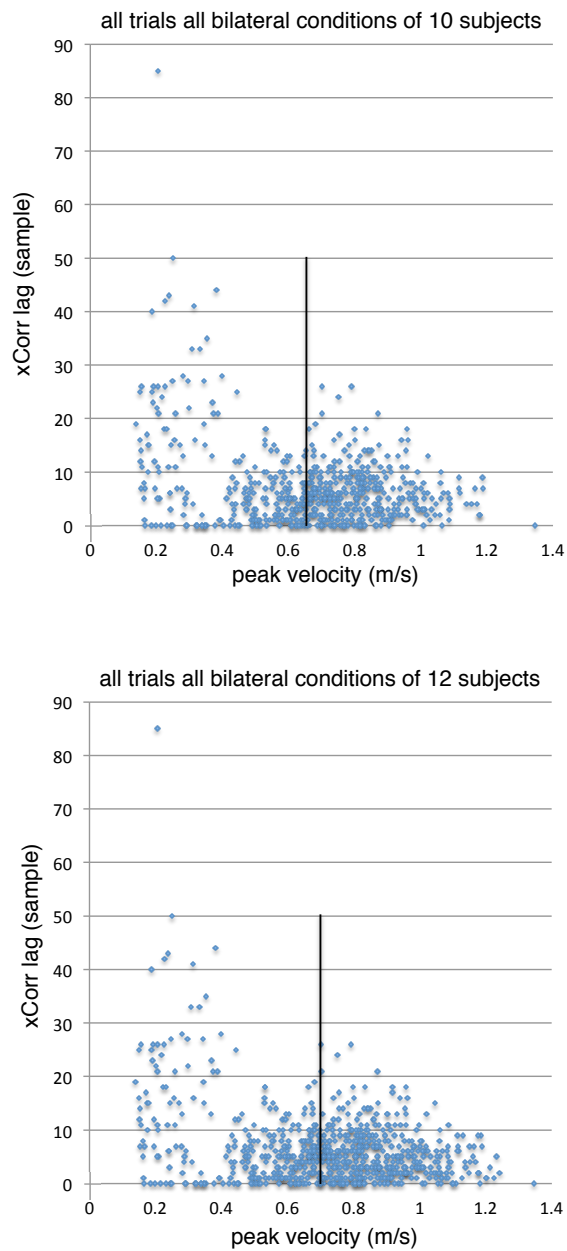


Figure 3.12. Scatter charts of cross correlation lag of 10 and 12 subjects (which include the 2 higher level subjects). Each sample is .01 seconds. The vertical solid black line marks the average peak velocity of all bilateral condition.

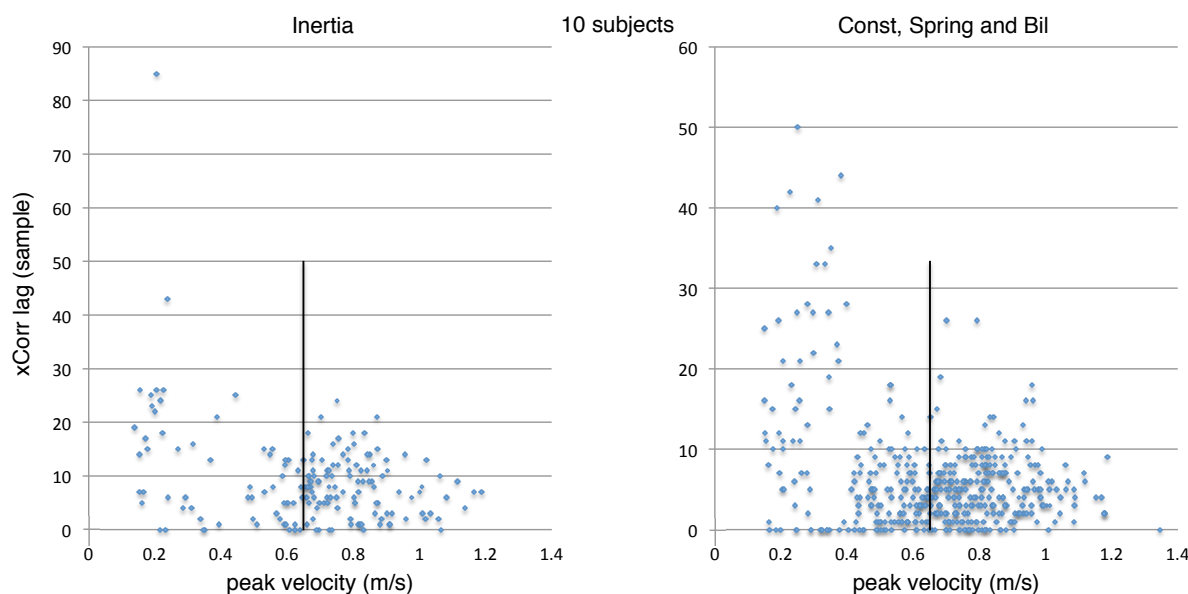


Figure 3.13. Scatter charts of cross correlation lag, classified by groups of conditions, of 10 subjects. Each sample is .01 seconds. The vertical solid black line marks the average peak velocity of all bilateral condition.

In addition to the above seven important metrics, there are a few minor metrics we investigated. Both of xCorr function value at zero lag and xCorr function max appear to be slightly varied across the task conditions but not significantly distinct, with mean  $\pm$  SE of  $.70 \pm .016$  and  $.82 \pm .0094$ , respectively. Note that the xCorr function is normalized to one. Since, we solely lean on the oral movement instruction and the cognitive behavior of the subjects in executing the task, we never have a perfect xCorr function max of 1.00 for any bilateral movements, even with Bilateral no-load. Visual feedback may help improving the case for the future work by noting the subject with red or green to say their speed at the target hitting is too slow or too high.

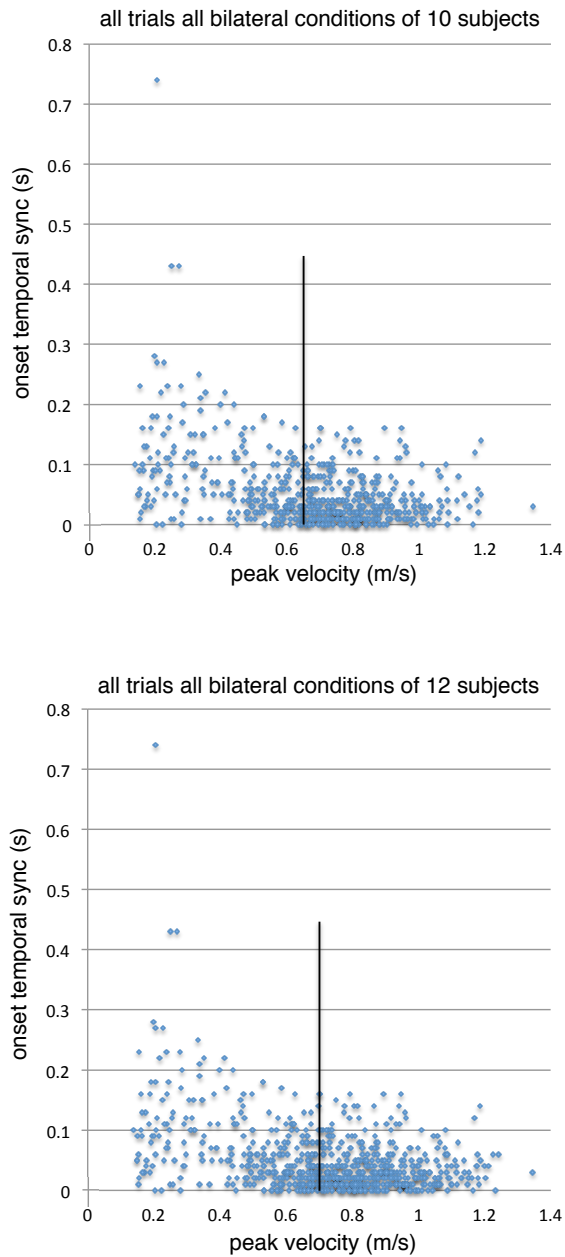


Figure 3.14. Scatter charts of onset temporal synchrony of 10 and 12 subjects (which include the 2 higher level subjects). The vertical solid black line marks the average peak velocity of all bilateral condition.

With the way we defined the other three minor parameters, reaching-onset or -complete temporal synchrony and reaching-onset spatial synchrony might not disclose a meaningful difference between bilateral no-load and loading conditions. Instead, these minor parameters were used as favor in eliminating those anti-phase movement trials that accidentally happened when the subjects forgot the movement instructions or out of focus as they said out.

The reaching-onset temporal synchrony seems to report an interesting trend versus peak velocity (Figure 3.14) as it happened to the xCorr lag mentioned earlier. Either with 10 subjects or 12 subjects, the trend of getting smaller onset temporal lag at higher speed of movement is remained. The 10 includes all 9 subjects and the outlier S10. The 12 additionally covers the 2 subjects we excluded due to their equivalent high kinematics as on healthy. There is vertical solid black line on the chart indicating the average peak speed of all trials of all bilateral conditions. We can see more points of having lower onset lag on the right side of the line. Overall, there is a tendency that we move along the horizontal axis on the chart for higher speed we will see lower lag. In addition, there is a little difference (not significant) of the onset lag (seconds) between (Bil+Inertia, lag=.05) and (Const+Spring, lag=.03). But the trends appear similar between these groups (Figure 3.15).

There is an outlier subject (S10), in which all bilateral conditions either with or without loading always gave favor of improving all key metrics (PV, PPA, P-EMG, NMU, xCorr lag) compared to unilateral (figure 3.16 and 3.17). Const and Spring loading did not give the consistent same trend effect caused by progressive loading as in the 9-subject group. Increasing

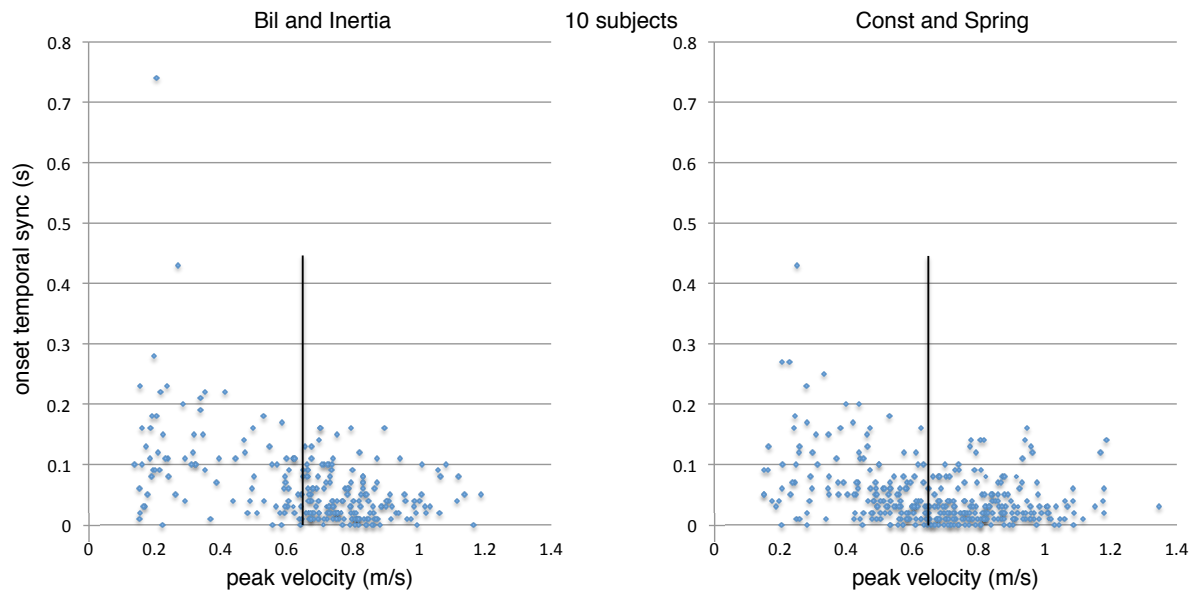


Figure 3.15. Scatter charts of onset temporal synchrony, classified by groups of conditions, of 10 subjects. The vertical solid black line marks the average peak velocity of all bilateral condition.

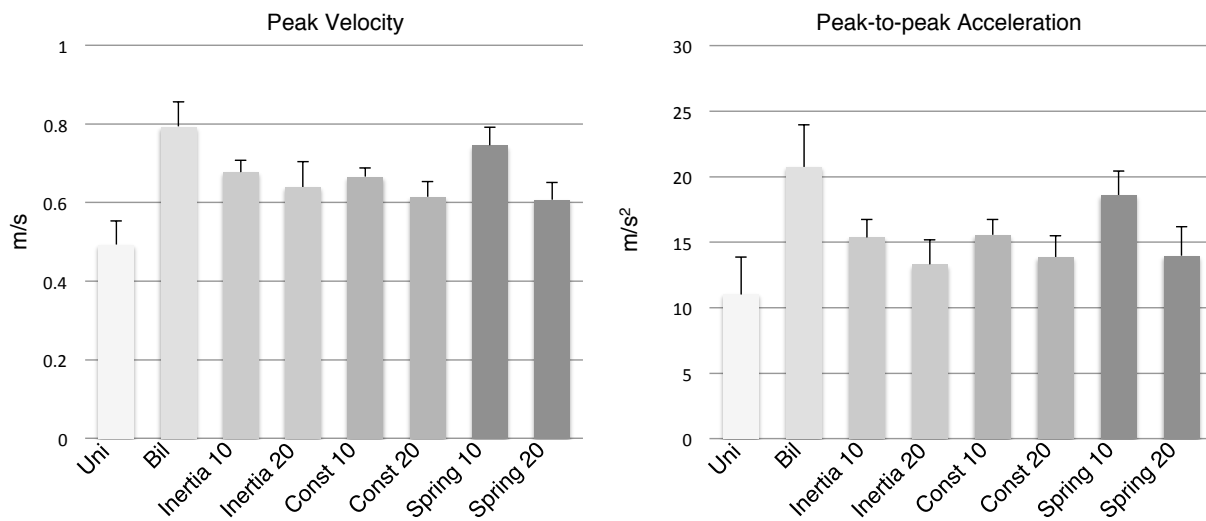


Figure 3.16. Plot of peak velocity and peak-to-peak acceleration of one distinct subject (S10). The two parameters of seven bilateral conditions are better than those of unilateral for this unique subject. Data is averaged across ten trials ( $\pm$ SE).

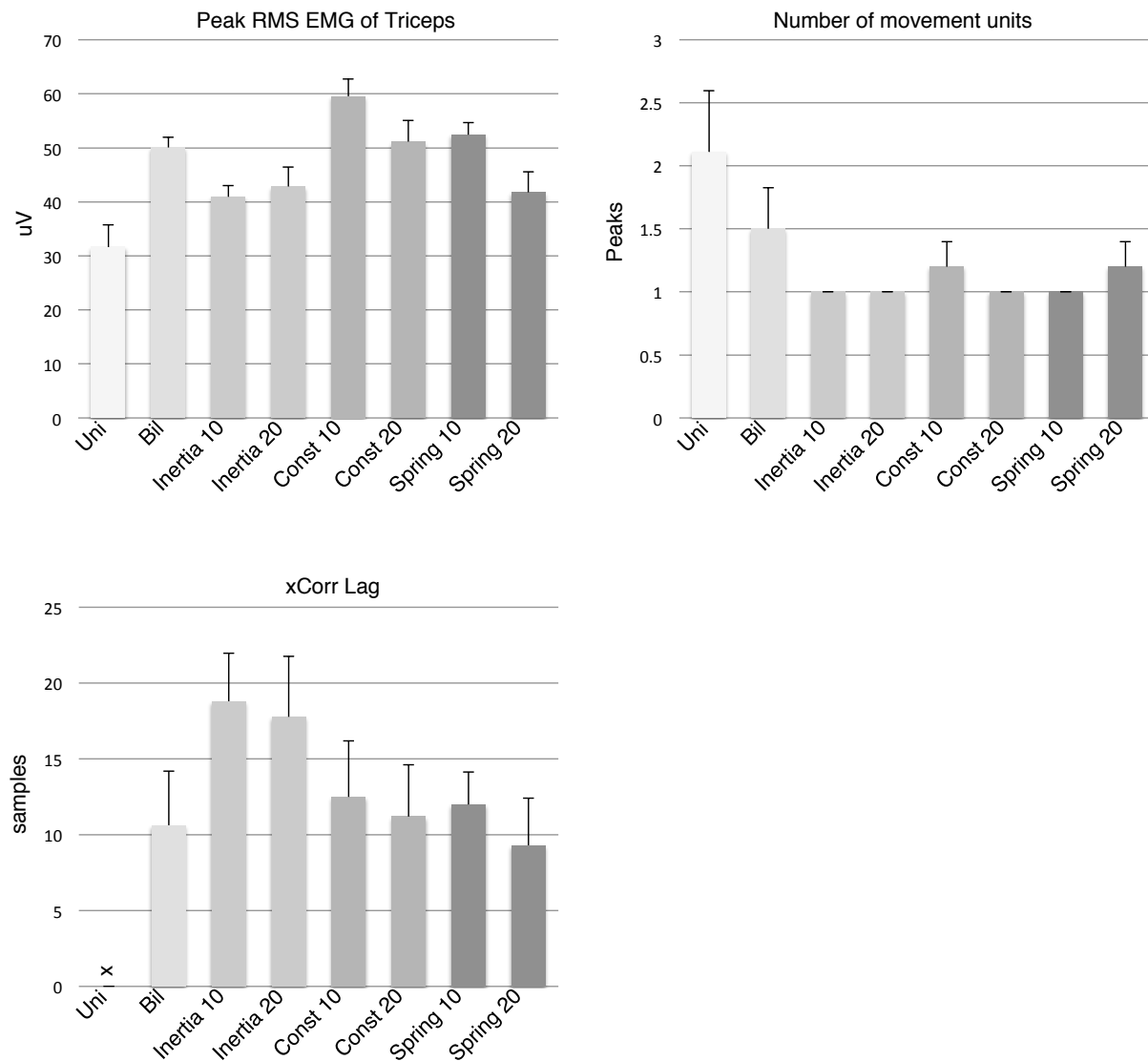


Figure 3.17. Peak triceps RMS EMG (P-EMG), number of movement units (NMU), and cross correlation time lag (xCorr lag) of one distinct subject (S10). The EMG and NMU of seven bilateral conditions are better than those of unilateral for this distinct subject. The xCorr lag is worst with Inertia loading while it is equal for all other bilateral task conditions. Data is averaged across ten trials ( $\pm$ SE).



the load in these conditions make worse kinematic metrics. As it turns out the bilateral no-load may be the best of producing highest PPA and PV. However, Const and Spring loading might help elevate the movement smoothness as the NMU indicated. NMU of Uni (2.1) and Bil (1.5) was advanced towards better in Const (1.1) and Spring (1.1). Finally, xCorr lag remained the same behavior as we observed in the group of 9 subjects. The Inertia is the worst while all other conditions are the same in terms of the lag of cross correlation.

The complete velocity (CV) is the speed of movement when two hands hit the soft target bar. We expected this to be the same for all subjects and so any comparisons between conditions will be more accurate. However, due to the lack of a visual feedback about this speed per each

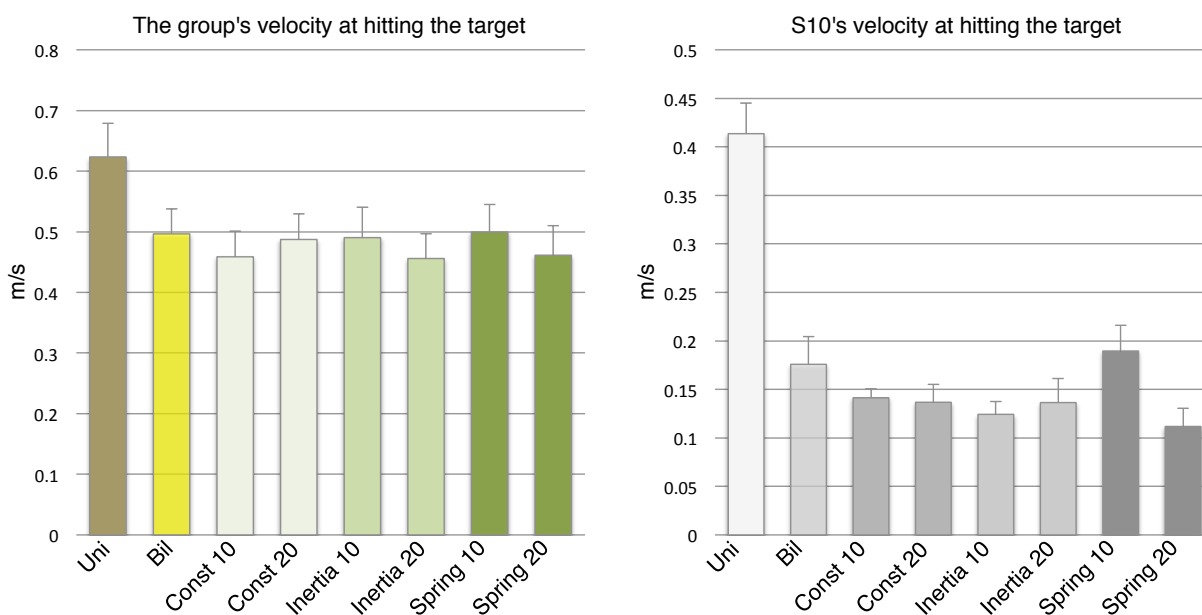


Figure 3.18. Velocity at the time the target is reached by soft hitting (as the oral instruction suggested) of the whole group of 9 subjects compared to the outlier subject S10.

trial, we got CV as in the figure 3.18. With the 9-subject group, CV of Uni is higher than that of other conditions. This may help unilateral kinematic parameters (PPA, PA, PV) higher than those in bilateral conditions. However, the outlier subject S10 is again an exceptional case. Though CV of unilateral is higher than all bilateral, S10 still produce better performance in bilateral conditions compared to unilateral.

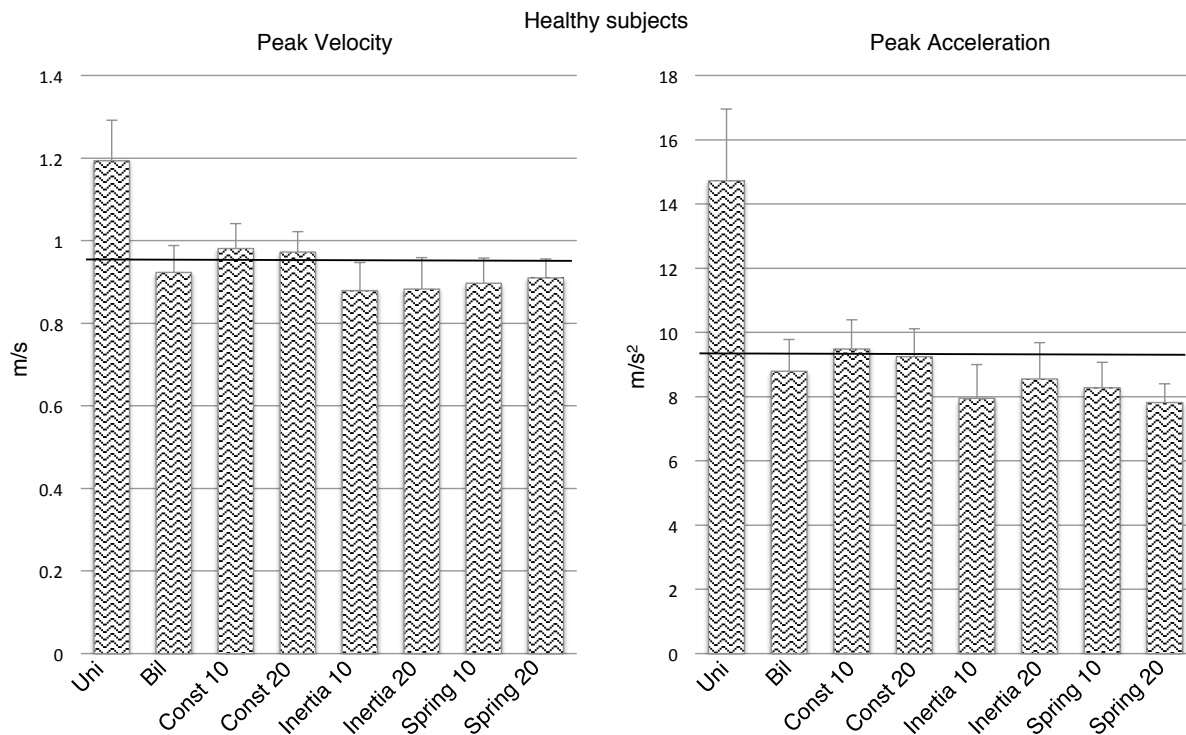


Figure 3.19. Bar plot of peak velocity and acceleration of the group of 2 healthy subjects. Data is averaged across 2 subjects ( $\pm$ SE). Progressive loading in each resistive condition results in no significant change in kinematic parameters of the non-dominant limb. No significance was found between bilateral conditions. The horizontal solid black lines are the average cross all conditions.

The healthy group exhibited a result of velocity and acceleration in a totally different way compared to the patient group above. Loading was applied to the dominant hand while the non-dominant kinematic data was analyzed. There was no effect of progressive loading in Constant and Spring at all. All peak velocity and peak acceleration in these two conditions are almost equal of those in Bilateral or Inertia (Figure 3.19). Although we recorded only two healthy subjects, but this perhaps suggests that our resistive loading methods would bring effect on stroke subjects, but not on healthy.

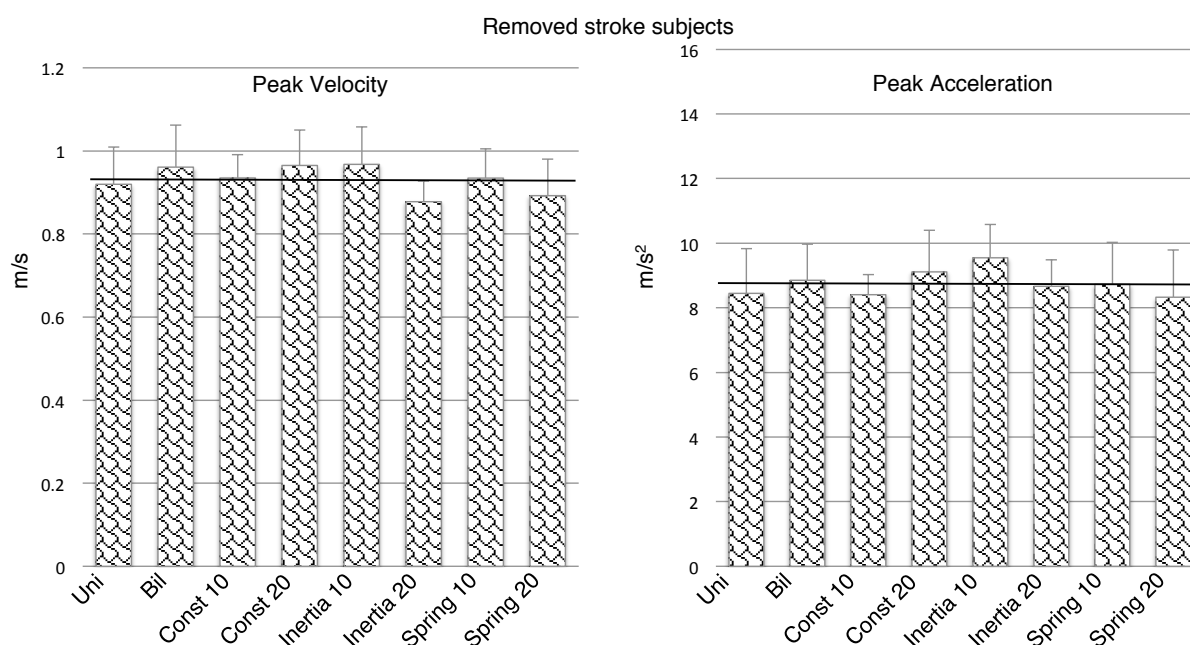


Figure 3.20. Bar plot of peak velocity and acceleration of the group of 2 removed stroke subjects. Data is averaged across 2 subjects ( $\pm$ SE). As the horizontal solid black lines show the average cross all conditions, these stroke subjects have velocity (m/s) and acceleration (m/s<sup>2</sup>) as high as the healthy, .93 versus .95 and 8.76 versus 9.35, respectively. Within bilateral no-load, they also have approximately equal velocity and acceleration values, .96 versus .92 and 8.85 versus 8.79, respectively.

There were two stroke subjects withdrawn from the group of eleven subjects for further analysis. As averaged cross all conditions, these removed stroke subjects have velocity (m/s) and acceleration ( $\text{m/s}^2$ ) as high as the healthy, .93 versus .95 and 8.76 versus 9.35, respectively (Figure 3.20). Within bilateral no-load, they have approximately equal velocity and acceleration, .96 versus .92 and 8.85 versus 8.79, respectively.

Last note is about the 9 subjects in the above whole group to be divided into two sub-groups based on FMA. Table 3.3 shows p-value significance for each of higher (37-53) and lower (15-24) FMA sub-groups. The higher FMA sub-group has five subjects of more mild upper extremity motor impairment. The lower one has four subjects of more severe motor impairment. Compared to those of the whole group in Table 3.2, the kinematics PPA, PA and PV of each sub-

Table 3.3. Statistical analysis p-value outcome for two sub-groups of subjects divided by higher (37-53, five subjects) and lower (15-24, four subjects) Fugl-Meyer Assessment (FMA). The p-value in dark orange is noticed as significant, in light yellow is tendency towards significant, in x-mark as it is not significant and in dash-mark as either not applicable or no need to apply. PPA peak-to-peak acceleration, PA peak-acceleration and PV peak-velocity.

		Significance	PPA	PA	PV	Name of conditions considered
Higher FMA group	1	3x2 ANOVA: interaction 3 condition x 2 load level	0.071	0.045	0.109	Const, Inertia, Spring
	2	2x2 ANOVA: load level	x	x	-	Const, Spring at 2 load levels
		interaction	x	x	-	Const, Spring at 2 load levels
Lower FMA group	1	3x2 ANOVA: interaction 3 condition x 2 load level	0.098	0.131	0.128	Const, Inertia, Spring
	2	2x2 ANOVA: load level	0.047	-	-	Const, Spring at 2 load levels
		interaction	x	-	-	Const, Spring at 2 load levels

group exposed much less significance spots at larger p-value. Since the result on these first three key parameters was poor, we did not seek to investigate other parameters within the sub-groups.

### **3.4 Discussions**

The bimanual setup has its minor asymmetry that the subjects could see. Its robot intervention attached to only one side of the two limbs while the paretic limb moved freely. This may affect the expected spatial coupling between two limbs in bimanual task conditions. In addition, in combination with audible signal, there was a bold X marked on the target bar which is still during the movement. We verbally required the subjects looking at this X when they executed the movements. The X was only moved a little bit when the target bar was hit. We also required the subjects moving as fast as possible. Therefore at most of the time, they hit the target bar quite strong but not soft as we wanted. Missing something in motion for the eyes to track down according to hand movement could be a reason to explain why speed and acceleration in all bilateral were lower than unilateral. Somehow, the temporal coordination between limbs was not best exploited yet.

We saw the subjects with highest FMA score did not exhibit highest movement speed. There were two subjects with FMA of 40 being removed as in Table 3.1 because their speeds matched the healthy ones. Meanwhile the subjects with FMA of 50 and 53 were appropriately included in group of nine subjects since their speeds were clearly lower than healthy subjects. Similarly at the lower bound, at FMA of 14, the subject moved so slow that not able to be

analyzed. While speed of the next low subject with FMA of 15 was in the mid-range of the group. And the subject with FMA of 24 turned out to be the slowest.

Compared the profile velocity of our study with Cunningham et al., 2002, both studies showed that unilateral produced higher speed than bilateral despite loading or not. Their study said the smoothness was improved with bilateral for 3 of 6 subjects compared to unilateral but our study did not assure any difference in smoothness over conditions although we can see it may be improved as in Figure 3.8, the profile velocity and acceleration. Their study's inertial loading also helped improved 5 of 6 subjects in smoothness compared with unilateral. Our resistive loading (not inertia) improved several parameters compared to bilateral no-load, but not to unilateral. Their research observed that efficiency of facilitation (smoothness of movement) should be considered on individuals rather than group; and our outcome also exposed this, there was a particular individual strongly beneficial (speed, acceleration) from the resistive loading.

Harris-Love et al., 2005 with task of reaching a box's side, showed bilateral no-load attained higher speed and accelerator than unilateral while its inertial bilateral loading did not bring further improvement in paretic arm movement. Their study came to conclude that the interlimb coupling effects during bimanual reaching were retained even after chronic stroke and can be used to provide an immediate improvement in paretic arm reaching performance. Our study did not disclose this conclusion. This can be enlightened by further investigation based on our earlier understanding in this discussion section.

With rapid coffee mug reaching by Chang et al., 2006, the bilateral no-load had lower speed than unilateral while its inertial loading exhibited even worse. This is similar to our study.

Our Inertia 20% had the lowest speed, even though, we thought this is likely due to marginally unstable robotic behavior. Their study found that moderate group's maximal velocity came earlier than mild group. While we found that time-to-peak of Constant and Spring loading were shorter than Bilateral no-load.

The temporal coordination between limbs during bilateral no-load and resistive loading was well existed even after hemiparesis. But this was not demonstrated with our inertia loading as two limbs showed movement times unequal. Similar finding of bilateral no-load temporal interlimb coupling was seen from Waller et al., 2006, when simultaneous bilateral was compared with sequential movement. We believe this temporal coordination can be further improved if the subjects' eyes traces on something that moves along the hands during their task. This in combined with having our hands or arms direct towards the same motor target may help coordination signals sent simultaneous better to their effectors.

There were two stroke subjects been removed from further analysis but they are not the ones with highest FMA. These subjects produced equivalent kinematic quantities as high as with the healthy subjects. Looking into the assessment scoresheet, FMA measure on motor impairment recovery after stroke does not cover upper extremity single muscle movement at maximal voluntary speed. Therefore, FMA score is probably not proportional to the movement velocity and acceleration of the upper limb movement. This likewise help explained why the division the whole group into two sub-groups did not classify the significance on the kinematic quantities among the sub-groups.

### 3.5 Conclusions

The verbal instruction and nature of experimental setup seems to have decreased bilateral movement speed compared to unilateral. Despite this decrease, we saw the temporal coordination remained existing in bilateral with either no-load or resistive loading. The results of this chapter suggested that within bilateral task conditions, the novel constant loading should be most recommended, followed by the novel spring loading. Compared to unloaded bilateral and inertia loading, these novel resistive conditions with a robotic device helped improved the paretic limb performance with outcome key measures ( $p < .02$ ) of peak-to-peak acceleration, peak velocity, movement time, time-to-peak, cross correlation lag and EMG. Increasing load level in the non-paretic limb improved most of those parameters in the paretic limb within resistive loading under spring and constant force patterns. Progressive resistive loading suggested their potential to future studies related to bimanual training. The constant loading appears to be the most effective of the bimanual conditions, likely most influent on the slow movement subjects but not on the ones with the low FMA score (Figure 3.21). The subject with lowest speed of movement was the one gaining most from the constant and spring loading ( $p < .001$ ) in key measures. Meanwhile, increasing inertial loading actually decreased the speed of the paretic limb. Hopefully, the works gradually lead us to specify the kind of bimanual intervention with the highest potential and consequently, we may develop a new bimanual training paradigm on a robotic device (Nef et al., 2009).



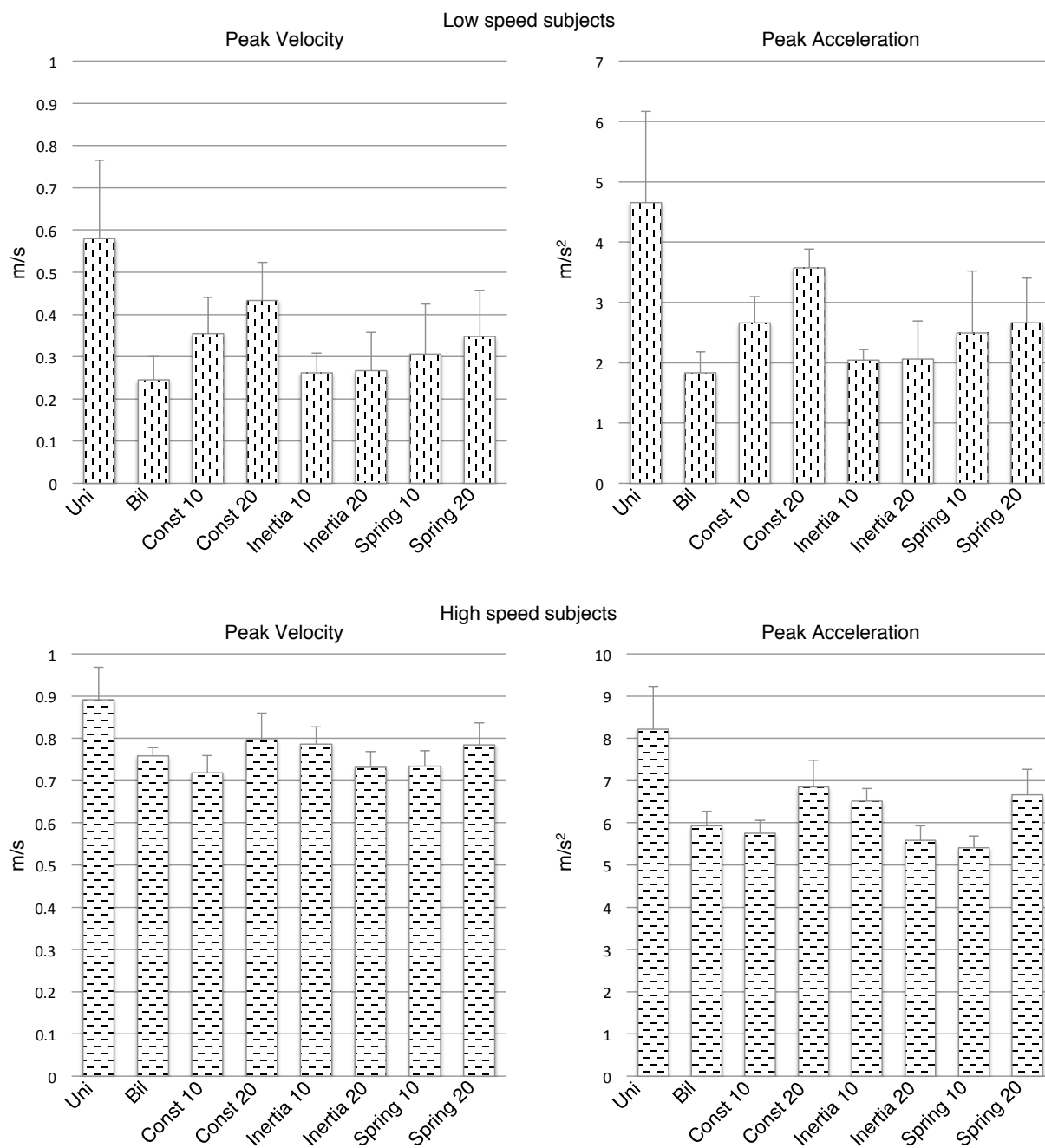


Figure 3.21. Low (2 subjects) versus high speed group (7 subjects). Data is averaged across all subjects ( $\pm$ SE). Low speed group shows a strong trend of progressive loading effect by Constant and Spring while no effect with loading on Inertia. High speed group shows similar but much weaker trend on Constant and Spring, especially compared to bilateral no-load.

In the study of this chapter, the two limbs were coupled by the instruction to move the two limbs in synchrony. Data from this chapter suggested that paretic limb speed was highest in trials where the two limbs were highly synchronized. In a follow on study in healthy controls in the next chapter, we studied different methods to enforce greater synchrony between the two limbs. An EMG coherence method was used to assess the degree of interlimb coupling as a function of different coupling methods.

## **Chapter 4**

### **EMG Coherence in Bilateral Tasks with Visual Feedback on Healthy**

#### **4.1 Introduction**

Bimanual upper limb task with different mechanical setup and visual feedback were hypothesized to produce distinct EMG coherence results. Two mechanical independence conditions of isometric contractions and one mechanical dependence condition with a shared pivot to allow two handles detached for tiny free movement (Free), are performed. Accordingly, graphic user interface is slightly different; isometric conditions have the cursor bar tilt-able and either with or without visual ball-balance (bb) rolling on; while in Free, the cursor bar is simply horizontal. Twelve adult healthy subjects participated in randomized six conditions divided into two groups of extension (1 bbEx, 2 Ex, 3 ExFree) and flexion (4 bbFl, 5 Fl, 6 FlFree) elbow task in horizontal plane with adjustable arm supports. High loading target of 50% maximum voluntary contraction was used. A running rectangular was used as visual guidance for subjects to accurately apply force to move the cursor bar.

Coherence determines the degree of correlations between two signals in frequency domain (Rosenberg et al., 1989). EMG-EMG coherence is the focus of this study. This coherence was found between bilateral arm muscles during fatiguing contractions, i.e. elbow flexor or

extensor against upward/downward a steel bar (Boonstra et al., 2007). The coherence was exposed at range of 8-12 Hz and increased with fatigue, clearer for extension than flexion. In an isometric contraction task of abducting the thumb against the manipulandum, the analysis said that the EMG-EMG coherence is maximal at range 1-12 Hz (Farmer et al., 2007). While the coherence is high at range 16-32 Hz that depends on cortical drive to motoneurone and is coherent with cortical oscillation of around 20 Hz. The coherence, especially at around 20 Hz, is detected very little for 4-9 year-old children, but gets clearer when it comes to ages 12-14 years that reach adult. EMG coherence is stronger for extrinsic than intrinsic muscle pairs for a series of 3-digit (thumb, index and middle fingers) grasp tasks in 8 subjects. In addition, the coherence is not affected by force (Poston et al., 2010).

It showed peak EMG coherence at range of 15-30 Hz for a 'hold' precision grip task at about two Newtons. While during the ramp task (hold-ramp-hold precision grip), the coherence at the same range is significantly reduced during the ramp movement, but significantly increased in the second hold period, relative to the initial hold (Kilner et al., 1999). Note that they use rectified-EMG that may shake up the results.

The following study also used rectified-EMG that may alter the results. But anyway, EMG bilateral coherence has been found in frequency band of 7-13 Hz for about one second in the movement phase of increasing force of both hands in a thumb-index-fingers grip task to a stable force production (Boonstra et al., 2009). However, the EMG synchronization is diminished or absent during the stable force phase of grip. Target force was one Newton and task duration was 4.5 seconds in which the ramp was about one second. Thirteen healthy participants

were recruited. EMGs on the first dorsal interosseous (FDI) muscle and flexor pollicis brevis (FPB) muscle of both hands were recorded.

Though this is not directly related to pure EMG coherence analysis, in an isometric contraction task on 6 healthy subjects, 15-33 Hz cortical-muscle (MEG-EMG) coherence was found for upper and lower limb muscles (Salenius et al., 1997). Note that this used rectified-EMG. Electrocorticogram - electromyogram coherence during isometric contraction of wrist extensor is observed between primary motor cortex and EMG at 12-18 Hz in all 8 subjects (Ohara et al., 2000). Six subjects do isometric contraction with four muscles (Gross et al., 2000) and it found cortico-muscular coherence around 20 Hz as well as phase lag support the guess that primary motor cortex drives the motoneuron pool. The study showed strongest cortico-muscular coherence in isometric contraction in range of 14-40Hz as well as particular role of beta band (they define as 13-24 Hz) in movement control.

To decide a movement task is either objective (apply force to move over a distance) or isometric (apply certain force without movement). Though contradictory, the above review implied that more studies mentioned that isometric movement may be a better choice to produce higher coherence. In addition, slower movement is likely better in producing coherence (Evans et al., 2003; McAuley et al., 1999). With index finger in-phase movements, bilateral coherence was most exhibited within the transition from a force ramp to stable phase of contractions (Evans et al., 2003). In contrast, during a grip task, the cortical muscle coherence was significantly smaller when the task was performed under an isometric condition compared with a compliant condition in which subjects moved the levers against a spring-like load (Kilner et al., 2000). Furthermore,

larger displacements during the compliant condition of the grip task produced higher coherence, and so isometric condition gave least coherence. EMG coherence was found between leg muscles of the walking task or standing even though it is not strong as within leg (Boonstra et al., 2008; Halliday et al., 2003). Note that Boonstra used rectified-EMG in this study.

As mentioned in the previous chapter, the cross correlation may not clearly show up due to the lack of visual feedback, the task was controlled a lot by oral instruction. EMG coherence was improved after visuo-motor (ankle dorsi-plantarflexion movement) training session in 8 out of 11 subjects at range of 15-35 Hz (Perez et al., 2006). Similarly, EEG-EMG coherence around the same range was significantly increased in 9 out of 11 subjects after the training, specific for trained muscles. The coherence is remained unchanged for untrained muscles. We want to vary constraints of visual feedback from loose to tight to see visual role.

Hypotheses of the study chapter are (1) Which one of the six task condition exhibits most coherence. The difference between isometric condition and tiny movement condition should be exhibited; (2) Ball balance (bb) task should give more coherence development compared to no-bb due to a tighter control of the balance between limbs; and (3) Relationship of this study (on healthy subjects) should be somehow connected to interlimb coupling study (on stroke subjects) to bring direction for future works.

## **4.2 Methodology**

### ***4.2.1 Procedure and Apparatus***

The subjects were recruited through public ads to seek for healthy volunteer students in the campus community as well as healthy colleagues at the Medstar National Rehabilitation Hospital in collaboration with the university; “The ads said about seeking new findings in the stroke treatment, currently in the phase of looking for healthy volunteers, ages 18-50, no history of neurological or psychiatric illness and no orthopedic conditions, to join a study of bimanual arm movement coherence. The study involves repetitive movements with two arms extending or flexing against physical objects with visual feedback. It requires one single visit to the hospital and lasts for less than two hours.” All recruitment and protocol documents were aligned with Medstar IRB including adverse event reports.

The subjects sit on a height-adjustable chair with their shoulder securely trapped to help torso kept still during the task to reduce shoulder involved in the arm movement. Twelve adult healthy subjects, ages ranged from 19 to 34, participated in the study, including male and female. All participants are volunteers and not excluded on the basis of race or ethnicity. But they must have no history of neurological or psychiatric illness and no orthopedic conditions that would prevent them from performing the upper extremity motor tasks required in the study.

Maximum voluntary contraction (MVC) of dominant elbow extension was recorded with a hand-held force muscle tester microFET2. The measure was carried out at limb posture aligned

with the isometric task described below. Noraxon's electromyography (EMG) was used with surface electrodes on biceps and triceps muscles.

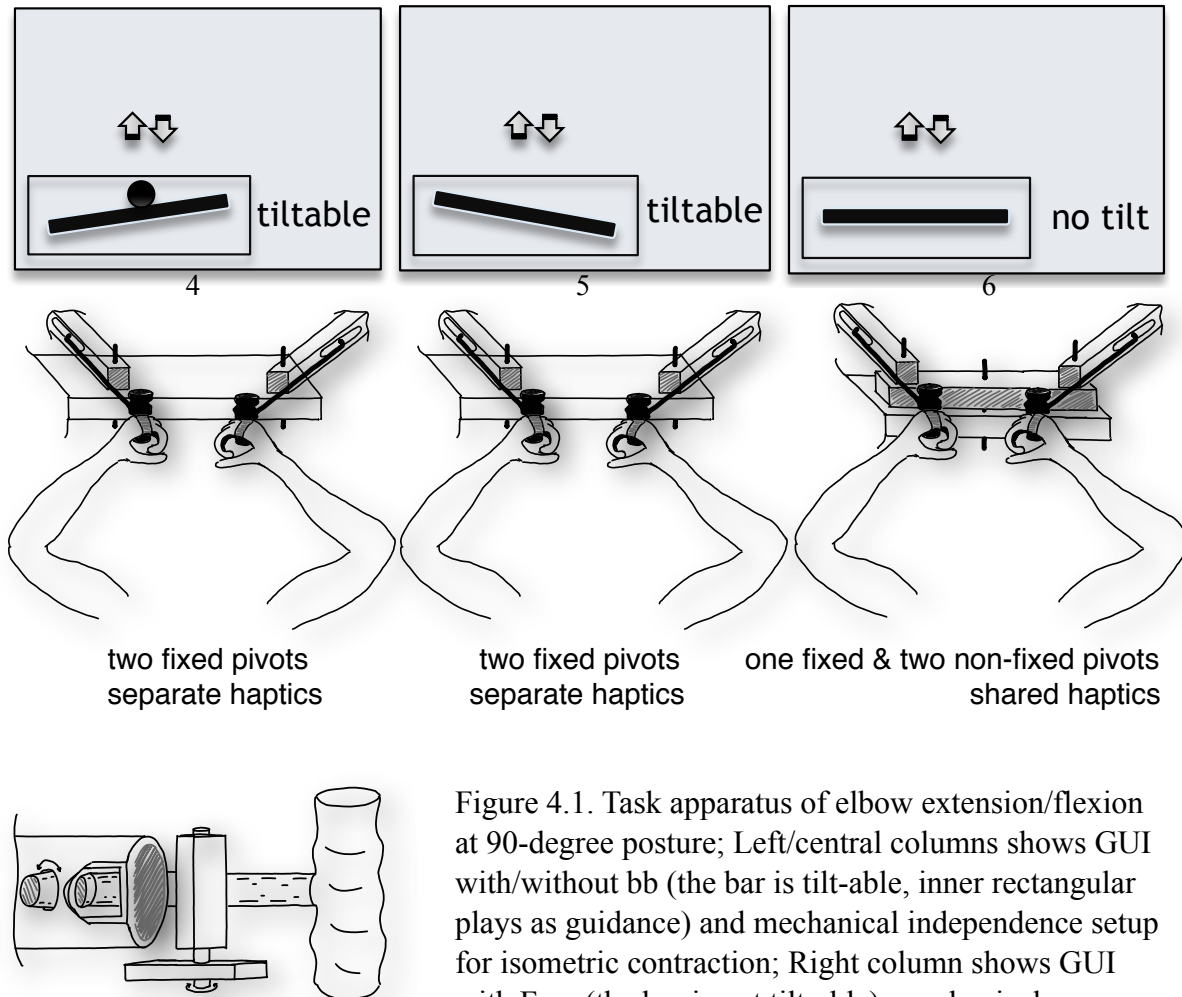


Figure 4.1. Task apparatus of elbow extension/flexion at 90-degree posture; Left/central columns shows GUI with/without bb (the bar is tilt-able, inner rectangular plays as guidance) and mechanical independence setup for isometric contraction; Right column shows GUI with Free (the bar is not tilt-able), mechanical dependence setup for tiny free movements; Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free; Bottom part shows an inside look for one of the four force sensor mechanisms.



For comfortable posture, subjects start with elbow flexed at 90 degrees (Figure 4.1). The required task is to conduct bimanual movements on horizontal plane of elbow extension (Ex) and flexion (Fl). The subject is instructed to look at a graphic user interfaces (GUI) cursor bar on screen. There are three constraints among three conditions:

- Isometric contraction with mechanical independence for two arms using separate force sensing rigidly attached to each arm, associating the GUI either:
  - with the cursor bar tilt-able and be balanced by force applied by two arms (abbreviated as blank)
  - as above but added a rolling ball on the bar (ball balance, abbreviated as bb)
- Two arms are detached to allow tiny free movements while in mechanical dependence by sharing a physical link pivoted at its center (abbreviated as Free), associating GUI:
  - with the cursor bar not tilt-able but simply horizontal

and, all is targeted to reach high level of loading, 50% of MVC.

In all cases, the GUI cursor bar is required to follow and must be within a guidance rectangular. For those cases of which the cursor bar is tilted then its tilt radius must be within the guidance rectangular. Totally, six conditions of the task will be executed, numbered (abbreviated) as, for extension: 1 (bbEx), 2 (Ex), 3 (ExFree), and for flexion: 4 (bbFl), 5 (Fl), 6 (FlFree). The three conditions in each group are gradually arranged from difficult to easy level of execution. The four conditions of those six are isometric while the other two are close to isometric because tiny movements are permitted.

Figure 4.1 describes the three GUI screens used for a trial for three conditions of either extension or flexion. Subject is homed at the start position by the guidance rectangular. When the sound alert is given, the guidance rectangular starts to move slowly from start towards target position. Subject listened to the sound and is required to produce force by elbow extending or flexing to move the cursor bar to track its guidance. This tracking is to keep the ramp-phase of movement identical for all trials. The stronger the subject isometrically applies to the handles by their hands, the higher the cursor bar will go up. The cursor bar will reach its preset value (when muscle force arrives at 50% MVC) at target position. Distance from start to target on computer screen is remained fixed, thus different MVC from different subjects is appropriately scaled to adapt the GUI. In four conditions, the difference in force applied by two arms will decide which end of the cursor bar will be higher than the other to describe the tilt. While in other two conditions, the cursor bar is simply horizontal and its position is the average force applied by two arms. The whole cursor bar, despite its tilt, must be inside the guidance rectangular all time of movement to be considered as a perfect successful trial. To complete a trial, subject must remain their force stable to stay at the target position as long as the guidance rectangular is there. Finally the subject releases their forearm push or pull completely when the guidance rectangular is back homed to the start position.

In the screen of GUI with ball balance (bb), a visual ball will be rolling on the cursor bar which will be the most challenging condition for the subject. The virtual ball rolling is coded to work like the physical one. The subject must obey all the requirement from the isometric condition without ball, plus, must control the ball rolling within dimension of the cursor bar. Ball

balance is expected to increase the coupling of two arms in real time in an adaptive manner. The subject can see the ball rolling back and forth to the left or right depending on the incline angle and speed of the bar which reflects how good coupling of their two limb movements. A successful trial is now marked by the additional requirement of the ball rolling on a bar without dropping out the bar's two ends. Mathematical equation for the ball roll at time  $i$  is defined as below, in which  $F_L$  and  $F_R$  are forces applied by left and right hands,  $c$  is central location of the bar on which the ball rolls on,  $E_R$  and  $E_L$  are the edge locations of the bar,  $\alpha$  is the tilt angle of the bar,  $d$  is the distance of the ball measured from the center of the bar,  $g$  is gravity acceleration,  $r_{ball}$  is radius of the ball, mass of the ball is implicit in the  $R_{roll}$  as rolling ratio to adjust fast or slow roll speed (in addition to the tilt angle).  $R_{roll}$  was empirically selected.

$$\begin{aligned}
 \sin(\alpha) &= \frac{F_L - F_R}{\sqrt{(F_L - F_R)^2 + (E_R - E_L)^2}} \\
 d(i) &= R_{roll} \frac{g}{2} \sin(\alpha)(t_i - t_{i-1}) + d(i-1) \\
 x_{ball}(i) &= c + d(i) \\
 y_{ball}(i) &= (c + r_{ball}) + \frac{F_R - F_L}{E_R - E_L}(c + d(i)) + F_L - \frac{F_R - F_L}{E_R - E_L} E_L
 \end{aligned} \tag{4.1}$$

A Tekscan's super thin loadcell is used for the lightness of the apparatus. We expect the mechanical enclosure, including the four force sensors, four rotatable mounting templates, four rods, and the handles to be light so it does not contribute any weight or inertia to the task during Free conditions. To help the sensor operated at its best accuracy, a special mechanism with two

perpendicular shafts, carrying two flat surfaces, was built to keep the contact of two surfaces always in parallel (Figure 4.1). The outer flat surfaces rotate around a vertical shaft while the inner two surfaces are from the shared cylinder which can rotate around a horizontal shaft. The cylinder is connected to the handle through a strong steel rod. Each of four sensors was individually calibrated (see appendix). Elbow supports and holders, which are not shown in the figure, can be adjustable for each subject with different length of forearms. The hand is in closed fist with its bottom used to push the handle in elbow extension, while another side used to pull the handle in elbow flexion. Thus, instead of grasping the handle, a velcro is used to strap the subject's fist to the handle.

The order of the task conditions is random and unique for each subject (Table 4.1). Fifteen trials of data collection are expected per each condition. Though a study on finger grip said that the level of loading 5%, 40% or 80% does not significantly affect the resulted EMG coherence (Poston et al., 2010). Our selected level of loading is high which is because we believe high enough contraction may produce better coherence. This was a poll decision after a pilot of another different set of eight subjects. To get 15 valid trials we conduct a total of 21 trials in which a pattern of move-rest of 5-3 trials was applied to help subjects enough time to rest to avoid fatigue. However, the subjects can rest at anytime they feel fatigue or continuously go complete all trials if they always feel good. A single trial includes in it a ramp phase of 3.6 seconds, a stay phase of 3.4 seconds and a break of 7 seconds. We empirically chose these values based on the data and tests after tests on half a dozen of real subjects with their responses. When a rest trial is told, the subject gets 14 seconds doing nothing and staying at complete rest. About

Table 4.1. Task conditions in random order and unique for each subject

Subject #	1	2	3	4	5	6	7	8	9	10	11	12
Task conditions (time order)	1	6	1	6	2	5	2	5	3	4	3	4
	4	3	4	3	5	2	5	2	6	1	6	1
	2	5	3	4	1	6	3	4	1	6	2	5
	5	2	6	1	4	3	6	1	4	3	5	2
	3	4	2	5	3	4	1	6	2	5	1	6
	6	1	5	2	6	1	4	3	5	2	4	3

five minutes will be consumed per each condition. When all other time spending on EMG placement and test, MVC collection, mechanical switching between conditions, data extraction, instruction, familiarization to be counted, each subject will play about 90 minutes per visit.

#### ***4.2.2 Data Acquisition and Analysis***

MVC was recorded, at the isometric comfortable limb posture, using Hoggan Scientific LLC microFET2 transducer. The sensor is an accurate, portable force evaluation and testing (FET) device, designed specifically for taking objective, reliable and quantifiable muscle testing measurements. The National Instruments Data Acquisition USB-6210 is used to collect force signals to Matlab GUI, at sampling rate of 100 Hz. The force sensor is equipped with an amplifier by Phidgets adaptor. Northern Digital Inc's Optotrak with extended ODAU II data acquisition board and its First Principle software were used to collect EMG signals at 1,000 Hz sampling rate. The system was in sync with GUI through serial communication. Starting the GUI on one computer will trigger EMG run on another computer at the same time. Noraxon U.S.A. Inc MyoSystem 1400A unit was used to measure electrical activities of biceps and triceps muscles on both upper limbs. The bandwidth of the unit is 10 to 500 Hz for surface EMG

electrodes. No notch filters (50/60 Hz) are used. The 25 pin D-sub connector of the Myosystem is partially wired to BNC connectors on the ODAU II. All EMG channels are gained set at a factor of  $500 \times 10$  in which 500 is fixed and scale of 10 was selected.

The GUI's data was used to guide the subjects to do expected trials and, also been extracted to verify the force variation across the task conditions. We can later look at the data graph and check if the cursor bar locates inside the guidance rectangular at all time during a trial. We can see if a subject started their trial on time, whether they track the guidance well during the ramp phase, they stably hold during the stay phase, or in some conditions if they keep the ball rolling back and forth inside the cursor bar dimension. (Figures 4.2, 4.3, 4.4)

Muscle pair coherence of EMG signals was analyzed with support from Neurospec 2.0 program (by D. M. Halliday, 2008) type 0, written in Matlab. The spectral estimation is grounded on weighted periodogram estimates, in which each periodogram is computed from discrete Fourier transforms (DFT) containing  $S$  points, where  $S$  is a power of 2. Each DFT segment contains  $T$  data points, where  $T \leq S$ . Zeros are appended to the  $T$  data points to create a segment of length  $S$  in case  $T < S$ . Type 0 was used for a single unbroken stretch of data. The data is split into  $L$  non overlapping segments. Each segment consists of  $T$  data points. This number of points is then used for DFT analysis as well,  $T=S$ . The total number of samples analyzed is product of  $L$  and  $T$ . Only complete segments are analyzed, therefore data points at the end of the data collection that do not form a complete segment are not included in the analysis. The segment length  $S$  is specified as an input argument, as a power of 2.

The analysis counted in all trials including those just completed part of a trial. The partial trial might exist when the subject saw they either failed to catch the guidance box or failed to keep the ball in the bar, and thus decided to stop the contraction immediately. The segment power was assigned as 9 and so the segment length  $S$  was 512 ( $2^9$ ) and  $T$  was then assigned as 500. For the ramp phase only, each contained 3,600 points. With EMG signals from all 15 trials in the ramp phase to be concatenated to a large signal, total points would be 54,000 and hence,  $L$  should be 108 even (if we assign  $S=T=512$ , then  $L$  is about 105, plus there is the last incomplete segment which would not be included in the analysis). Accordingly, the analysis resolution was described with frequency bin a bit less than 2 Hz. The EMG inputs were in raw form after offset removal to bring the baseline to zero. But we also used root mean square EMG (with a choice of 45 samples of moving average calculation) to check the synchrony between GUI force data on one computer with EMG data on another computer (Figure 4.5). Peak RMS EMG was peeled to be compared over different task conditions and over the ramp or stay phase of movement. It was also used to check if the EMG within-limb cross-talk exists. In addition, we avoid using rectified EMG because it can alter the coherence result (Neto et al., 2010). The mean intensity of point-process  $N$  (corresponding to one of the EMG inputs) is now denoted as  $P_N$  and,  $P_{NM}$  as the cross-product density at lag  $u$ . Following Bartlett et al., 1963, the cross-spectrum between two point-processes at frequency  $\lambda$ ,  $f_{NM}(\lambda)$  is defined as

$$f_{NM}(\lambda) = \frac{1}{2\pi} \int_{-\infty}^{\infty} q_{NM}(u) e^{-i\lambda u} du \quad (4.2)$$

$$q_{NM}(u) = P_{NM}(u) - P_N P_M$$

thereof with a single point-process, the auto-spectrum,  $f_{NN}(\lambda)$  is defined as

$$f_{NN}(\lambda) = \frac{P_N}{2\pi} + \frac{1}{2\pi} \int_{-\infty}^{\infty} q_{NN}(u) e^{-i\lambda u} du \quad (4.3)$$

According to Rosenberg et al., 1989, the coherence  $|R_{NM}(\lambda)|^2$  of processes N and M at frequency  $\lambda$  is defined as

$$|R_{NM}(\lambda)|^2 = \frac{|f_{NM}(\lambda)|^2}{f_{NN}(\lambda)f_{MM}(\lambda)} \quad (4.4)$$

The coherence is symmetric in N and M. It may also be appeared to be bounded between zero and one. The null happens where knowledge of M-process is not used to linearly predict the N-process. The one corresponds to the case of zero mean-squared error of the prediction or saying, the linear prediction is perfect. Following Halliday et al., 1996, the 95% confidence limit was involved on the graph of coherence to purposely serve for another factor extraction, with definition as

$$1 - (.05)^{1/(L-1)} \quad (4.5)$$

Frequency bands, in Hz, of common drive 0-8, alpha 8-16, beta 16-35, gamma 35-51 and their combinations are considered for coherence analysis. It seems that for a small movement task, the corticomuscular coherence is most exhibited in beta band (Boonstra et al., 2009) during the changes in the force grip. This band is presumed to have a primary motor cortical origin



(Poston et al., 2010; Boonstra et al., 2009). While the alpha band looks like related to motor unit discharge (Vallbo and Wessberg 1993). The synchrony of discharge is stronger and observed more often in muscles involved tiny movement (Datta and Stephens, 1990; Smith et al., 1999). Gamma band is believed in association with sub-cortex in several contradictory aspects such as bilateral movement degradation (Serrien et al., 2002), or isometric bimanual contraction during

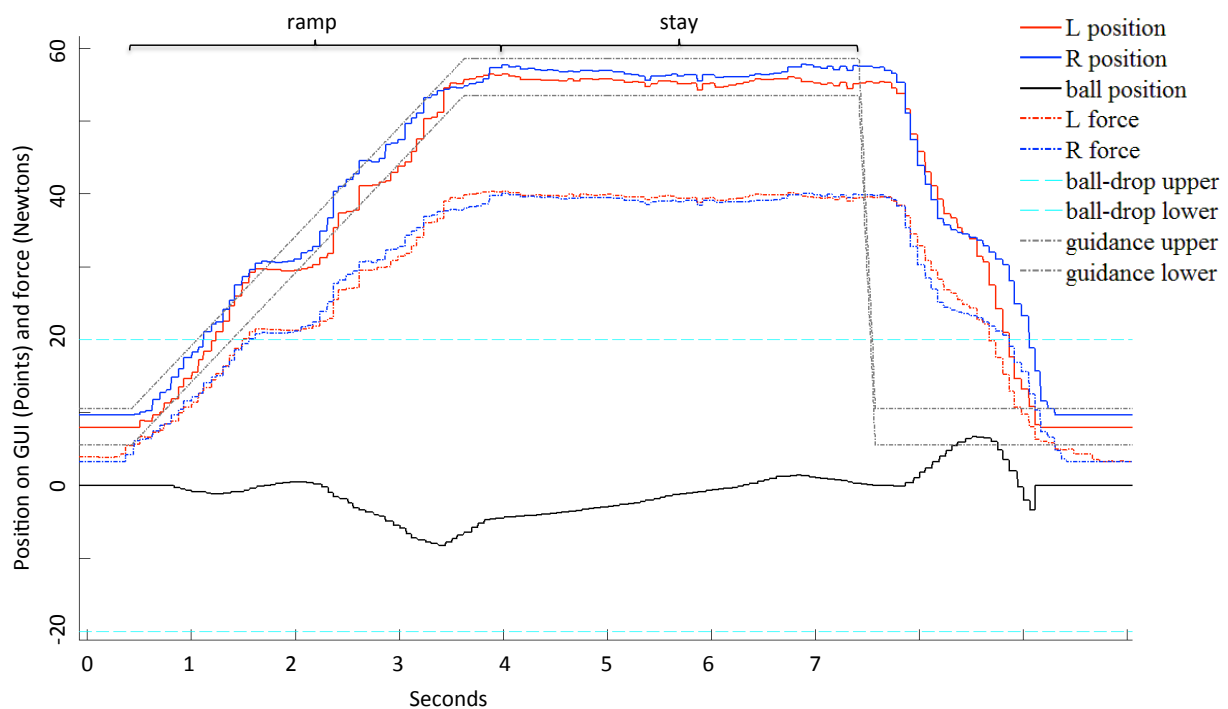


Figure 4.2. GUI's data collection of a successful trial in task condition 1 (bbEx - extension with ball balance) or 4 (bbFl - flexion with ball balance). A trial is considered to be successful when GUI's ball position falls within lower and upper ball-drop bounds and, GUI's left (L) and right (R) hand positions are between lower and upper bounds of the guidance in a duration of 7 s. The corresponding push/pull force, in Newtons, of L and R will reach 50% MVC when it enters the stay-phase. Length of the task's ramp phase is 3.6 s while of the stay phase is 3.4 s. To start a trial, the guidance starts to move at the same time with an audible tone play at .4 s counted from origin. To end a trial, the guidance will return from its target to start at 7.4 s counted from origin.

high force loading (Mima and Hallett, 1999; Omlor et al., 2007), and common drive band is presumed as associated with central origin (De Luca et al., 1982). In this study, we combined the alpha-beta-gamma as we believe each of these bands contribute to the coherence. We also investigated the alpha and beta bands only because each of them may dominate the coherence sum and if so, they may play a special role.

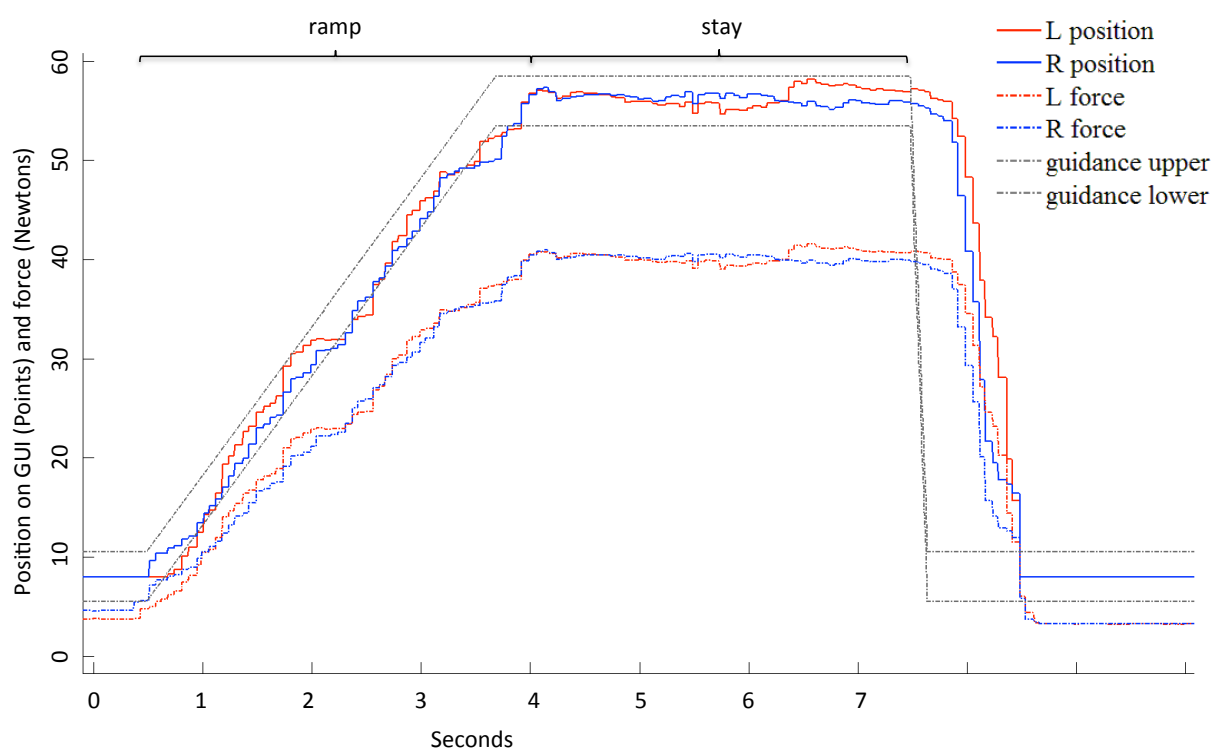


Figure 4.3. GUI's data collection of a successful trial in task condition 2 (Ex - extension without ball balance) or 5 (Fl - flexion without ball balance). A trial is considered to be successful when GUI's left (L) and right (R) hand positions fall between lower and upper bounds of the guidance in a duration of 7 s. The corresponding push/pull force, in Newtons, of L and R will reach 50% MVC when it enters the stay-phase. Length of the task's ramp phase is 3.6 s while of the stay phase is 3.4 s. To start a trial, the guidance starts to move at the same time with an audible tone play at .4 s counted from origin. To end a trial, the guidance will return from its target to start at 7.4 s counted from origin.

Z-transform was applied to convert the coherence into the value of coherence integral and peak. This is to allow statistical running purpose. A little piece of math was applied so that the integral does not cover the area of which the coherence was exposed below the 95% confidence limit. This limit was embedded in the Neurospec, following the computation method proposed in

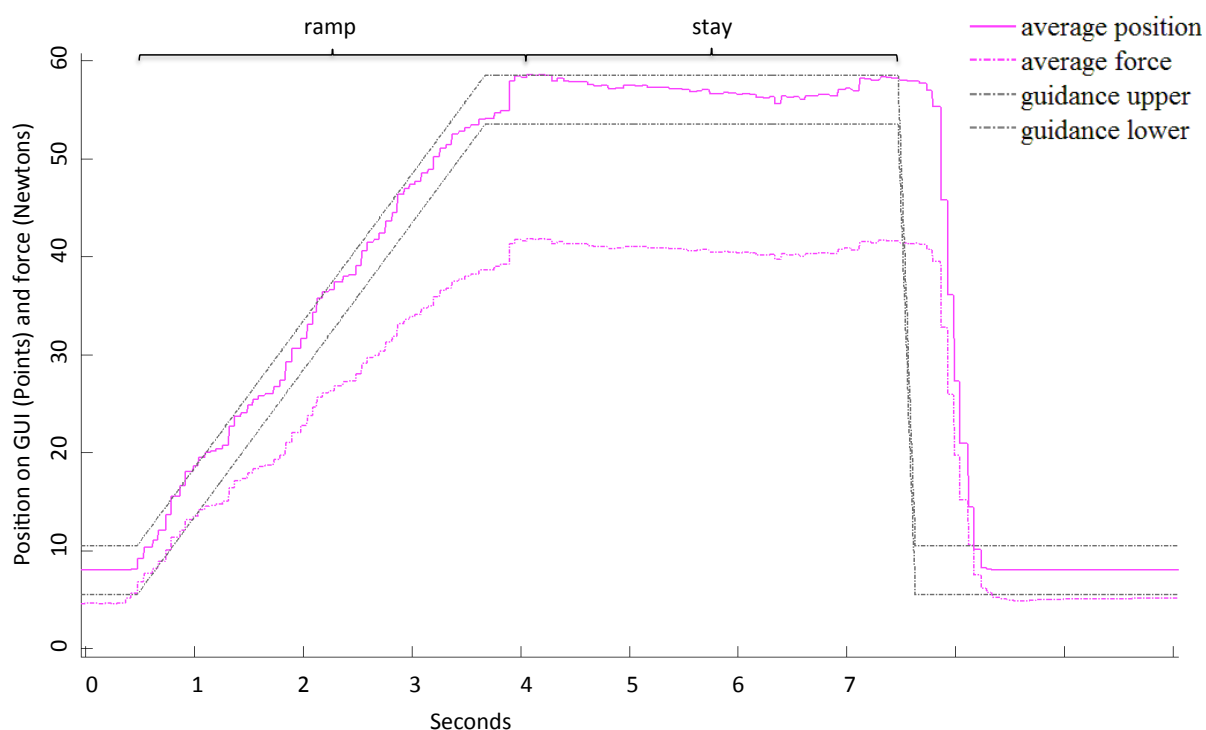


Figure 4.4. GUI's data collection of a successful trial in task condition 3 (ExFree - extension free) or 6 (FIFree - flexion free). A trial is considered to be successful when average of GUI's left (L) and right (R) hand positions fall between lower and upper bounds of the guidance in a duration of 7 s. The corresponding average force between L and R, in Newtons, will reach 50% MVC when it enters the stay-phase. Length of the task's ramp phase is 3.6 s while of the stay phase is 3.4 s. To start a trial, the guidance starts to move at the same time with an audible tone play at .4 s counted from origin. To end a trial, the guidance will return from its target to start at 7.4 s counted from origin.

the Fourier approach to the identification of the functional coupling between neuronal spike train (Rosenberg et al., 1989). The coherence  $R_{xy}$  is transformed by determining the arc hyperbolic tangent (Fisher transformation) as below.

$$z_{trans}(R_{xy}) = \tanh^{-1}(R_{xy}) = \frac{1}{2} \ln \left( \frac{1 + R_{xy}}{1 - R_{xy}} \right) \quad (4.6)$$

SPSS statistical repeated measure (RM) ANOVA and paired sample t-test were then able to be applied on the transformed data. We only run t-test for those with significant 3x1 RM ANOVA of 3 conditions x 1 task.

Muscle pairs of triceps-triceps are the agonist for elbow extension while biceps-biceps are the agonist for elbow flexion. We also calculated agonist-antagonist muscle pairs as minor support to perhaps better interpret the results. The ramp (incremental force) and stay phases (holding force) of the task were examined (Figures 4.2, 4.3, 4.4). The subject IDs linked to personal information were generated in sequence. Very brief personal information was collected, i.e. name, age, gender, dominant side, and yes or no answer to neurological or psychiatric illness and orthopedic condition. Any publication relevant to the subjects is in a non-identifiable manner.

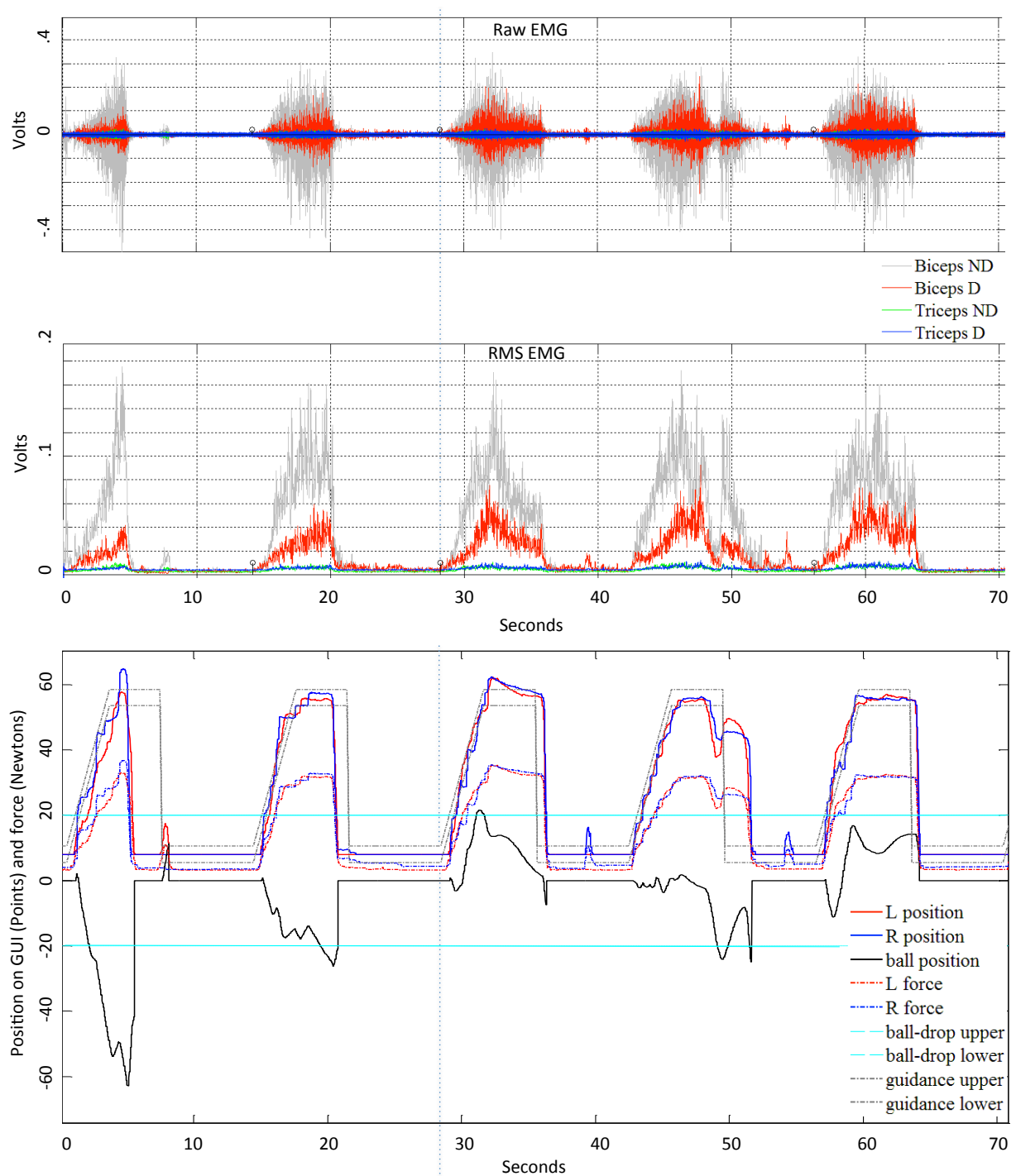


Figure 4.5. Plot of raw EMG and RMS EMG of five trials in a task condition, and corresponding GUI data. EMG onset is shown as a dotted vertical tick.

### 4.3 Results

Figure 4.6 shows the combined alpha-beta-gamma coherence integral (Coh) results for *ramp* (r) phases of *agonist* muscle pairs, i.e. triceps-triceps (TT) for elbow extension in upward-diagonal, and biceps-biceps (BB) for elbow flexion in solid-grey. Regarding the agonist muscle pairs during ramp phase, only extension task shows significant 3 conditions x 1 task (extension)

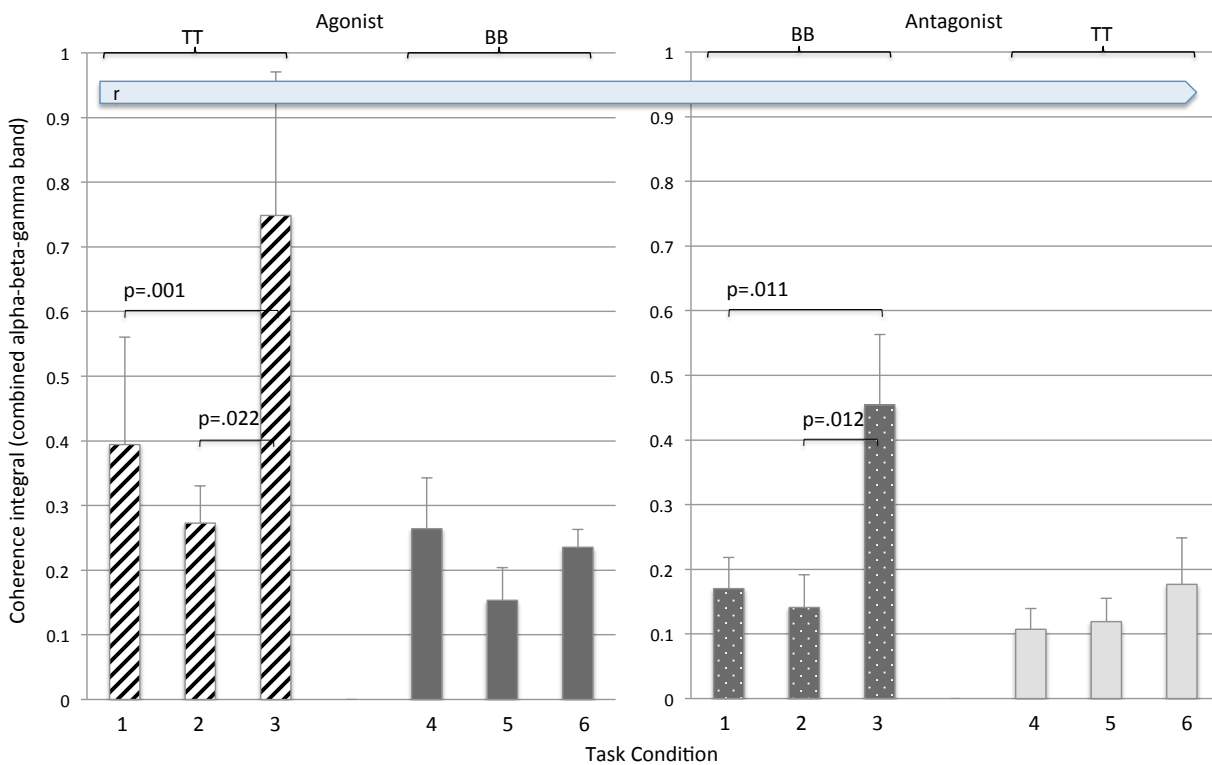


Figure 4.6. Combined-alpha-beta-gamma coherence integral on **ramp** (r) phases of triceps-triceps (TT) and biceps-biceps (BB); Left half: the agonist pairs for extension in upward-diagonal and for flexion in solid-grey; Right half: the antagonist pairs for extension in dotted-grey and for flexion in light-grey. Pairs with significant t-test p-value are shown. Note that the coherence integral is calculated above significance level. Data is averaged of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

repeated measure (RM) ANOVA ( $p=.006$ ), abbreviated as rTT agonist in table 4.2. This allows us to go through paired t-test comparisons within the extension. The ramp agonist coherence of condition 3 (Coh=.75) is significantly twice times larger than condition 1 (Coh=.39,  $p=.001$ ) and three times larger than condition 2 (Coh=.27,  $p=.022$ ). It means that with extension task, the tiny free bilateral limb movement produces best ramp agonist coherence compared to the isometric

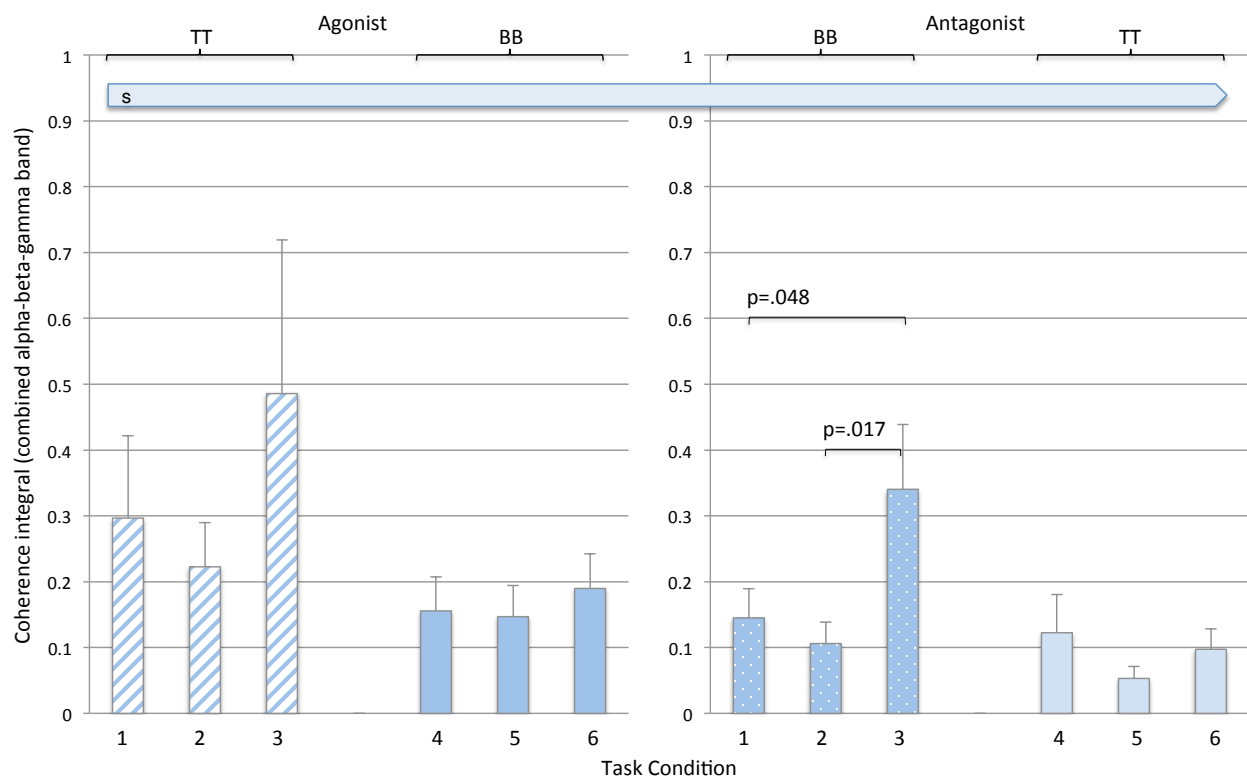


Figure 4.7. Combined-alpha-beta-gamma coherence integral on stay (s) phases of triceps-triceps (TT) and biceps-biceps (BB); Left half: the agonist pairs for extension in upward-diagonal-blue and for flexion in solid-blue; Right half: the antagonist pairs for extension in dotted-blue and for flexion in light-blue. Pairs with significant t-test p-value are shown. Note that the coherence integral is calculated above significance level. Data is averaged of all subjects ( $\pm SE$ ). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

movement with or without ball balance visual feedback. The coherence in ramp phase on the antagonist exposes a similar trend but smaller in all task conditions. The 3x1 RM ANOVA on antagonist in ramp phase of extension show significance ( $p=.004$ ), abbreviated as rBB antagonist in the table. As seen on the figure, the ramp antagonist coherence of condition 3 (Coh=.45) is significantly higher than condition 1 (Coh=.17,  $p=.011$ ) and condition 2 (Coh=.15,  $p=.012$ ). In

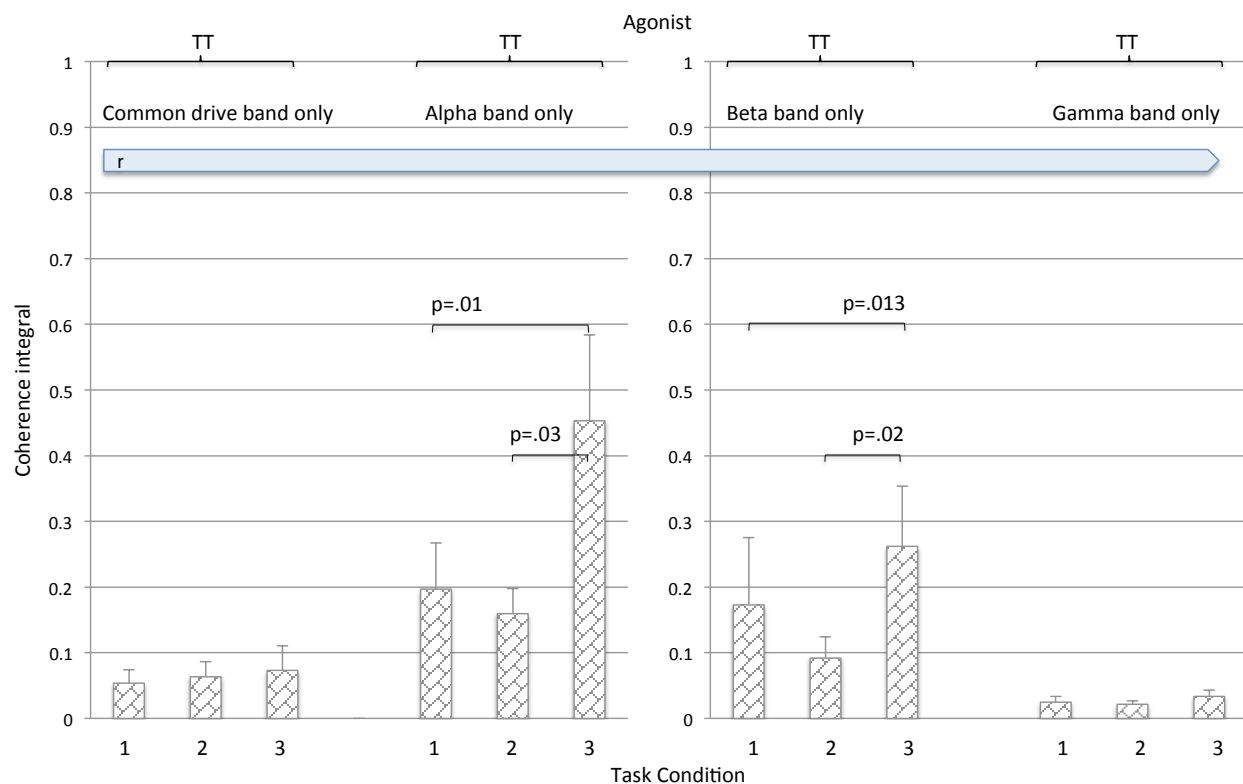


Figure 4.8. Coherence integral on agonist ramp (r) triceps-triceps (TT) for extension for each of the four frequency bands, in diagonal-brick. Pairs with significant t-test p-value are shown. Note that the coherence integral is calculated above significance level. Data is averaged of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.



Table 4.2. Statistics on combined alpha-beta-gamma band. Abbreviation: ext: extension, flex: flexion; r: ramp, s: stay, BB: biceps-biceps, TT: tricep-triceps; FrSE: ramp-phase force standard error, FsSE: stay-phase force standard error; nd: non-dominant, d: dominant. Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free. The \* marks the significance ( $p < .05$ ).

Combined alpha-beta-gamma band							
3x1 ANOVA				Sig.			
ext	rTT *	agonist	0.006	ext	rBndTd *	agonist-antagonist	0.035
	sTT	agonist	0.208		rTndBd *	"	0.002
	FrSE *		0.026		sBndTd	"	0.069
	FsSE *		0.001		sTndBd	"	0.319
	rBB *	antagonist	0.004				
	sBB *	antagonist	0.008				
flex	rBB	agonist	0.376	flex	rBndTd	"	0.185
	sBB	agonist	0.703		rTndBd	"	0.366
	FrSE		0.255		sBndTd	"	0.645
	FsSE		0.294		sTndBd	"	0.914
	rTT	antagonist	0.547				
	sTT	antagonist	0.273				
t-test							
ext	rTT	rTT-1 - rTT-2	0.377	ext	rBndTd	rBndTd-1 - rBndTd-2	0.603
	(agonist)	rTT-1 - rTT-3 *	0.001			rBndTd-1 - rBndTd-3 *	0.030
		rTT-2 - rTT-3 *	0.022			rBndTd-2 - rBndTd-3 #	0.058
	rBB	rBB-1 - rBB-2	0.690		rTndBd	rTndBd-1 - rTndBd-2	0.307
	(antagonist)	rBB-1 - rBB-3 *	0.011			rTndBd-1 - rTndBd-3 *	0.012
		rBB-2 - rBB-3 *	0.012			rTndBd-2 - rTndBd-3 *	0.004
	sBB	sBB-1 - sBB-2	0.184				
	(antagonist)	sBB-1 - sBB-3 *	0.048				
		sBB-2 - sBB-3 *	0.017				
	FrSE	FrSE-1 - FrSE-2	0.435				
		FrSE-1 - FrSE-3 *	0.034				
		FrSE-2 - FrSE-3 *	0.016				
	FsSE	FsSE1 - FsSE2 *	0.013				
		FsSE1 - FsSE3	0.668				
		FsSE2 - FsSE3 *	0.000				

contrast, the flexion task conditions numbered as 4, 5 and 6 does not exhibit the significant distinction in ramp phase on both agonist and antagonist muscle pairs. Additionally, the ramp coherence in flexion group is much smaller compared to the extension.

Breaking down the combined-alpha-beta-gamma band into each of the smaller bands will peel out which band contribute more compared to another to the total coherence. It is also for the advantage of predicting neuron or muscular origin of the coherence. Figure 4.8 states that alpha band and beta band are major parts contributing to the total coherence. Each of these two bands reports a similar trend as with the ramp agonist total coherence.

With alpha band ramp agonist muscle pairs, 3x1 (three conditions x extension task) RM ANOVA exhibits the significance ( $p=.008$ ), indicated as rTT agonist on table 4.3. This leads to t-test run between the conditions in the extension. It exposed that ramp agonist alpha band coherence in condition 3 ( $Coh_{\alpha}=.45$ ) is significantly higher than condition 1 ( $Coh_{\alpha}=.20$ ,  $p=.01$ ) and condition 2 ( $Coh_{\alpha}=.16$ ,  $p=.03$ ). Sum of the three conditions, alpha band coherence dominates 57% of the total coherence. Alpha band was believed in relationship with motor unit discharge.

Table 4.3. Statistics on alpha and beta bands separately. Abbreviation: ext: extension; r: ramp, s: stay, BB: biceps-biceps, TT: tricep-triceps. Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free. The \* marks the significance ( $p<.05$ ), and # as it closes to significance.

Alpha band only				Beta band only			
3x1 ANOVA			Sig.	3x1 ANOVA			Sig.
ext	rTT *	agonist	0.008	ext	rTT *	agonist	0.029
	sTT	agonist	0.201		sTT	agonist	0.507
	rBB #	antagonist	0.054		rBB	antagonist	0.338
	sBB #	antagonist	0.055		sBB	antagonist	0.418
t-test				t-test			
ext	rTT	rTT-1 - rTT-2	0.593	ext	rTT	rTT-1 - rTT-2	0.302
	(agonist)	rTT-1 - rTT-3 *	0.010		(agonist)	rTT-1 - rTT-3 *	0.013
		rTT-2 - rTT-3 *	0.030			rTT-2 - rTT-3 *	0.020

With beta band ramp agonist muscle pairs, 3x1 (three conditions x extension task) RM ANOVA also display the significance ( $p=.029$ ), indicated as rTT agonist on table 4.3. Another step of t-test statistics points out similar trend as with alpha band. Condition 3 ( $\text{Coh}_{\text{beta}}=.26$ ) leads the beta band coherence compared to other two, condition 1 ( $\text{Coh}_{\text{beta}}=.17$ ,  $p=.013$ ) and condition 2 ( $\text{Coh}_{\text{beta}}=.09$ ,  $p=.02$ ). By adding all three conditions, beta band coherence contributes 37% of

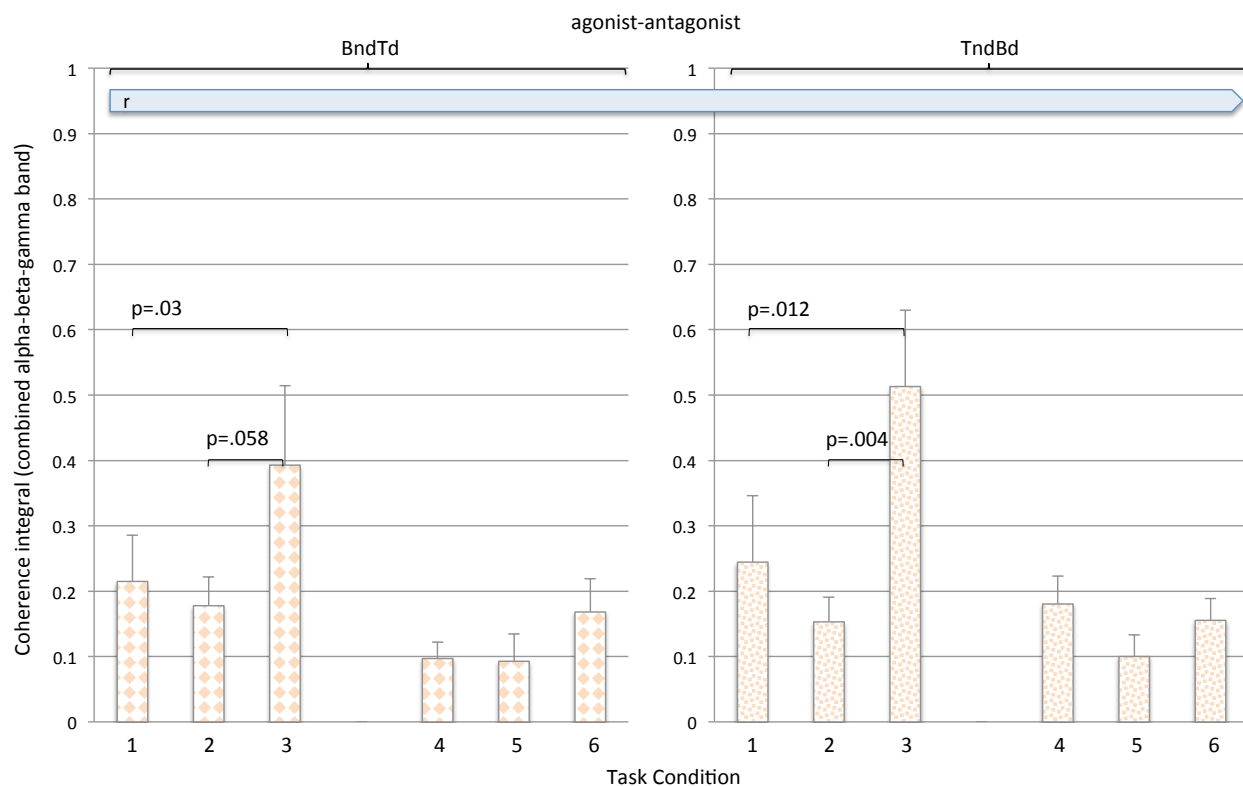


Figure 4.9. Combined-alpha-beta-gamma coherence integral on ramp (r) phases of biceps-non-dominant-triceps-dominant (BndTd) and triceps-non-dominant-biceps-dominant (TndBd), in diamond and confetti respectively. Pairs with significant t-test p-value are shown. Note that the coherence integral is calculated above significance level. Data is averaged of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

the total coherence. Beta band was presumed in relations to corticomuscular coherence, during the changes in the force. This band is confided to have a primary motor cortical origin.

In contrast, both alpha and beta bands do not show significance from ANOVA analysis during stay phase on agonist muscle pairs. These are because the bands are believed in relations with force transitions rather than force holding. During ramp phase and on agonist, we found no significance from statistical analysis for common drive band and gamma band, either with extension or flexion. Additionally, the table shows that no significance is found from the antagonist pairs for both bands, but p-value is tendency towards .05 for the alpha band.

Back with the combined alpha-beta-gamma band, the agonist-antagonist muscle pairs between limbs appears to expose the same coherence trend across three conditions with the extension task. The 3x1 RM ANOVA on ramp phase is significant for both pairs of Biceps-non-dominant limb and Triceps-dominant limb ( $p=.035$ ), and of Triceps-non-dominant and Biceps-dominant ( $p=.002$ ). These agonist-antagonist pairs during ramp phase are denoted as rBndTd and rTndBd on the table 4.2, respectively. The table and figure 4.9 discloses t-test statistics for rBndTd, with condition 3 at highest coherence ( $\text{Coh}=.39$ ) compared to condition 1 ( $\text{Coh}=.21, p=.03$ ) and condition 2 ( $\text{Coh}=.18, p=.058$ ), and for rTndBd, with the same trend, condition 3 at larger coherence ( $\text{Coh}=.51$ ) than condition 1 ( $\text{Coh}=.24, p=.012$ ) and condition 2 ( $\text{Coh}=.15, p=.004$ ).

On the other hand, the combined alpha-beta-gamma coherence during *stay* phase looks quite similar (Figure 4.7). However only the antagonist muscle in the extension shows significance from 3x1 RM ANOVA ( $p=.008$ ), denoted as stay biceps-biceps (sBB) antagonist on the table. Going further with t-test reveals the difference in coherence between condition 3

(Coh=.34) and condition 1 (Coh=.14,  $p=.048$ ) and condition 2 (Coh=.11,  $p=.017$ ). The difference between condition 3 and condition 1 is quite close to the .05 limit. The levels of coherence of all these three conditions are lower than that of the minor (antagonist) muscle during ramp phase. Therefore, it is clear that the ramp phase extension movement produced larger spread of coherence over muscle pairs and at higher level compared to the stay phase. The flexion group statistical analysis gives no significance during stay phase, as it does with the ramp.

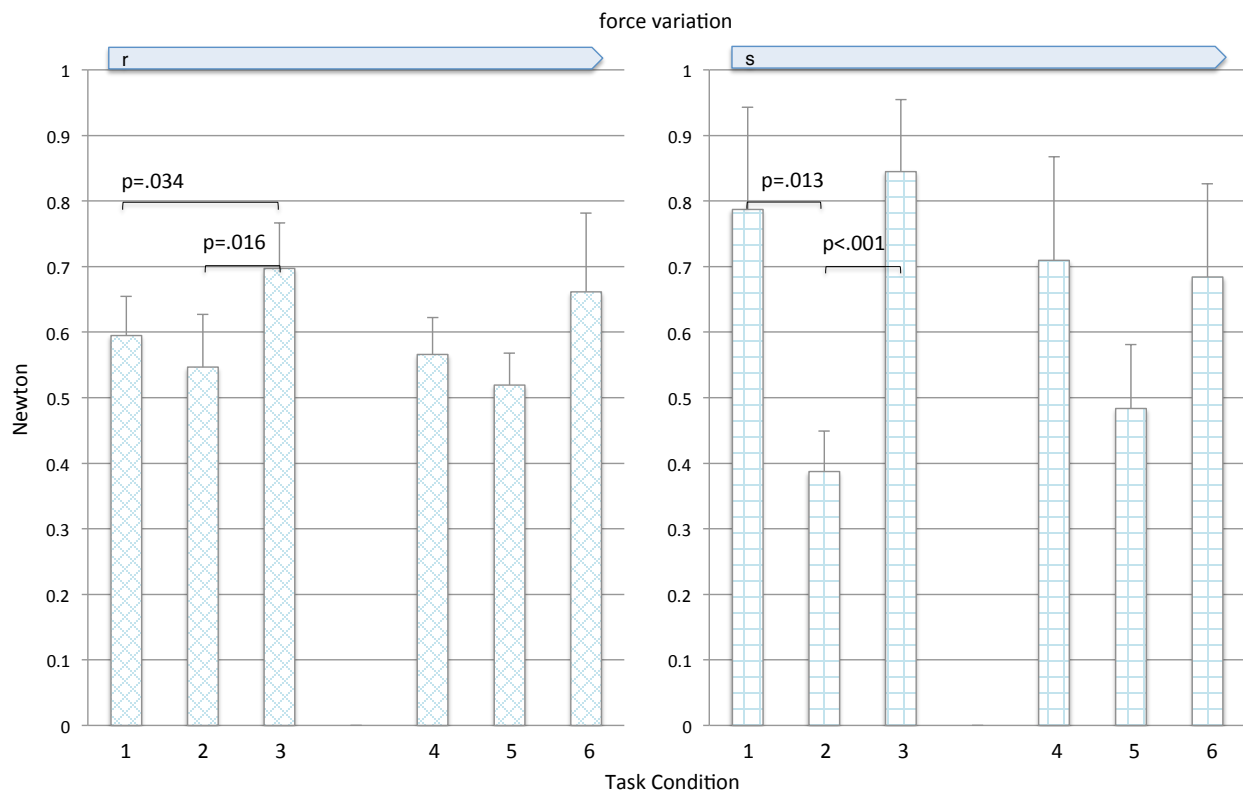


Figure 4.10. Force variation, in Newton, on ramp (r) and stay (s) phases. FrSE: ramp-phase force standard error, FsSE: stay-phase force standard error. Pairs with significant t-test p-value are shown. Data is averaged of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

The results suggest that if we expect highest coherence on ramp phase of the elbow task, we should do extension free in which the mechanical setup allows tiny movements, coming along with loosest visual feedback GUI (the cursor bar is not tilt-able but just simply horizontal). Compared to other two task conditions, the free condition is the easiest one which requires least effort to finish the task execution. All subjects' trials in this condition was close to the perfect trial definition (the cursor bar is within the guidance box all the time). The coherence trend across extension conditions is the same (highest with free condition and lower in other two isometric conditions), but larger with agonist muscle pairs and smaller with antagonist and agonist-antagonist pairs.

The variation of force (Figure 4.10), which is actually the standard error of forces averaged across all trials in each condition, is approximately the same between six conditions. It is all bounded under one Newton. This may support the fact that the quality of task execution over conditions is quite homogeneous. During the ramp phase, the condition 3 with tiny movements seems to have highest force variation compared to the other two isometric conditions ( $p=.034$ ,  $p=.016$ ) even though the difference is small. This is logical as it may be distinct between tiny movement and no movement (isometric). However during the stay phase, conditions 3 ( $p<.001$ ) and 1 ( $p=.013$ ) have higher force variation than condition 2. This perhaps can be explained as with condition 1, the subjects try hard to hold the ball not dropping out of the cursor bar.

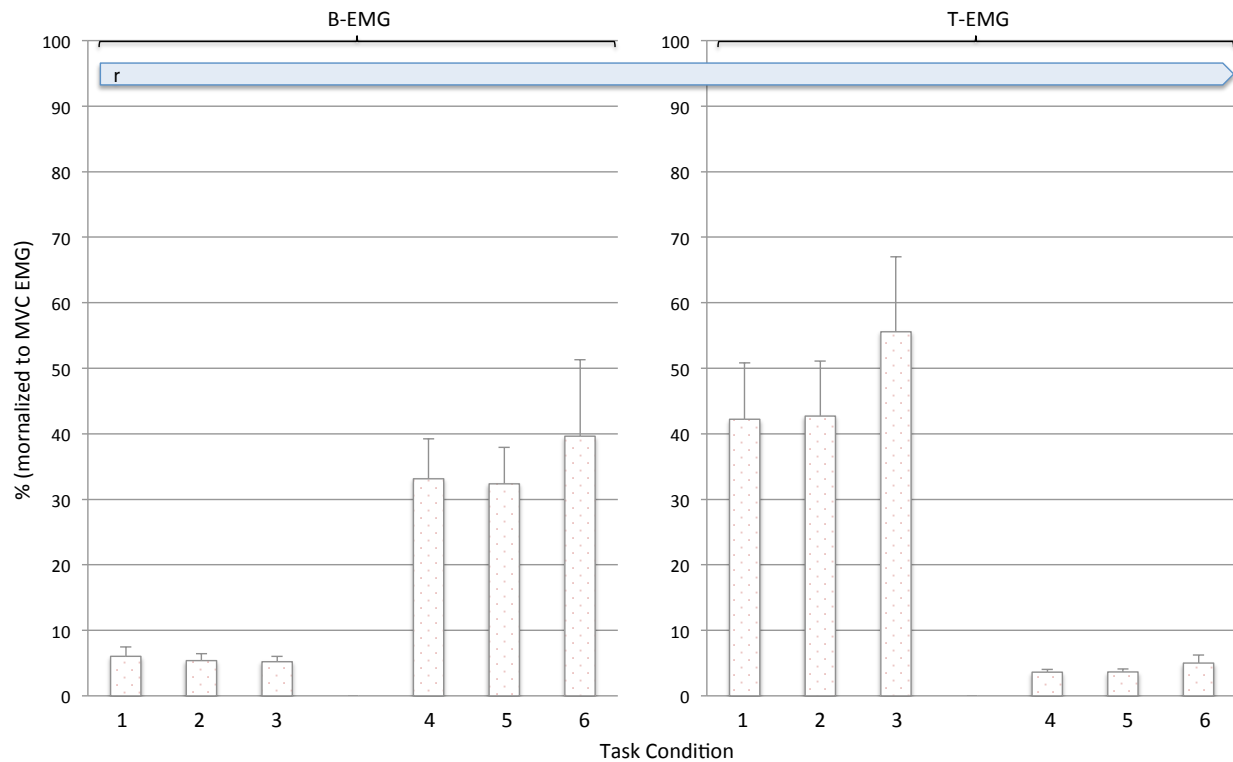


Figure 4.11. RMS EMG of ramp (r) Biceps (B) and Triceps (T). Data is averaged of two limbs and of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

The averaged peak of RMS EMG within each of the six conditions was shown as well (Figure 4.11 and 4.12). The chart maybe just served for observing if the cross-talk of the EMG signals within limb exists. As the agonist muscle pairs in each task of extension and flexion established much more of EMG level during the ramp (47%, 35%) and so dominate those of the antagonist muscles (6% and 4%, respectively). The similarity exists during the stay phase, (50%, 40%) versus (6%, 4%). These numbers eliminate the within-limb EMG cross-talk. This confirms that the coherence has no distortion caused by the cross-talk.

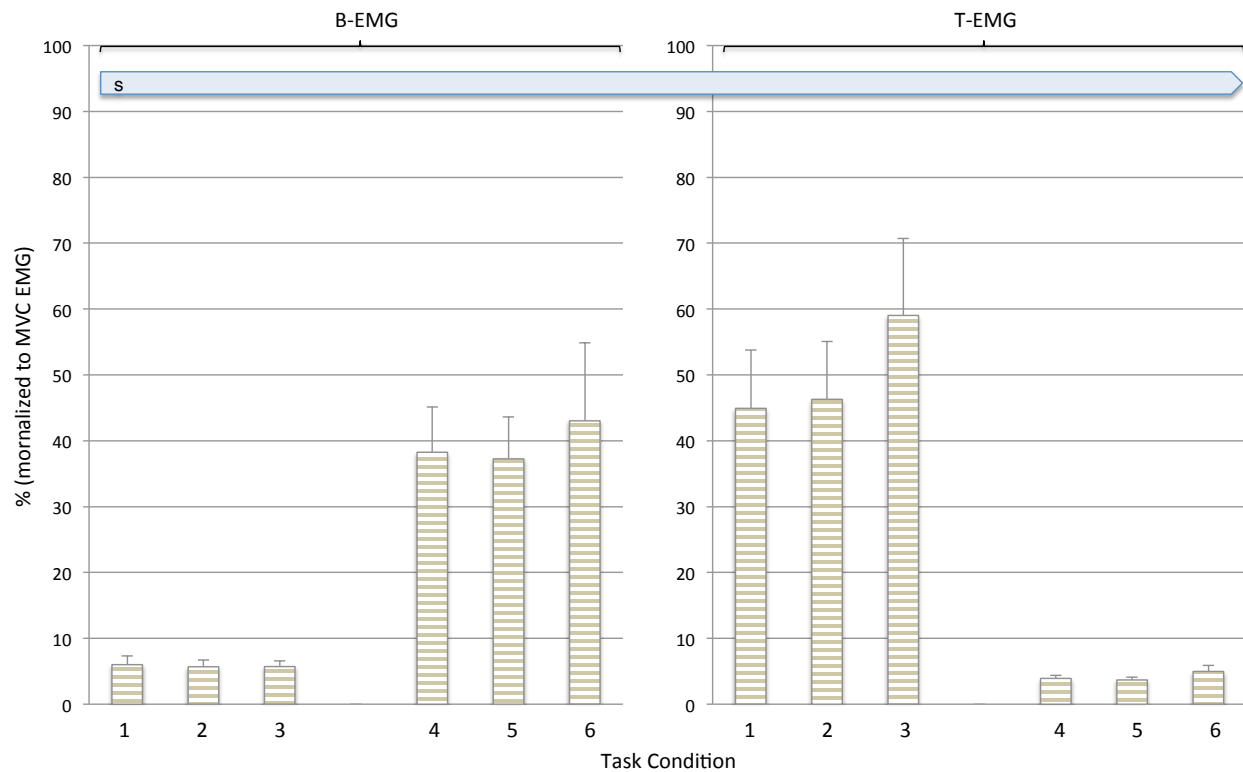


Figure 4.12. RMS EMG of stay (s) Biceps (B) and Triceps (T). Data is averaged of two limbs and of all subjects ( $\pm$ SE). Task conditions are numbered for extension, as 1 with ball-balance (bb), 2 without bb, 3 Free and for flexion, as 4 with bb, 5 without bb, 6 Free.

The figures from 4.13 to 4.16 show how coherence charts look like for each of the task condition for a single subjects. The 95% coherence limit was greater than zero, and calculated in the neurospec with the equation described earlier, but has been mathematically processed and shifted to zero in these figures to demonstrate the coherence integral computation.



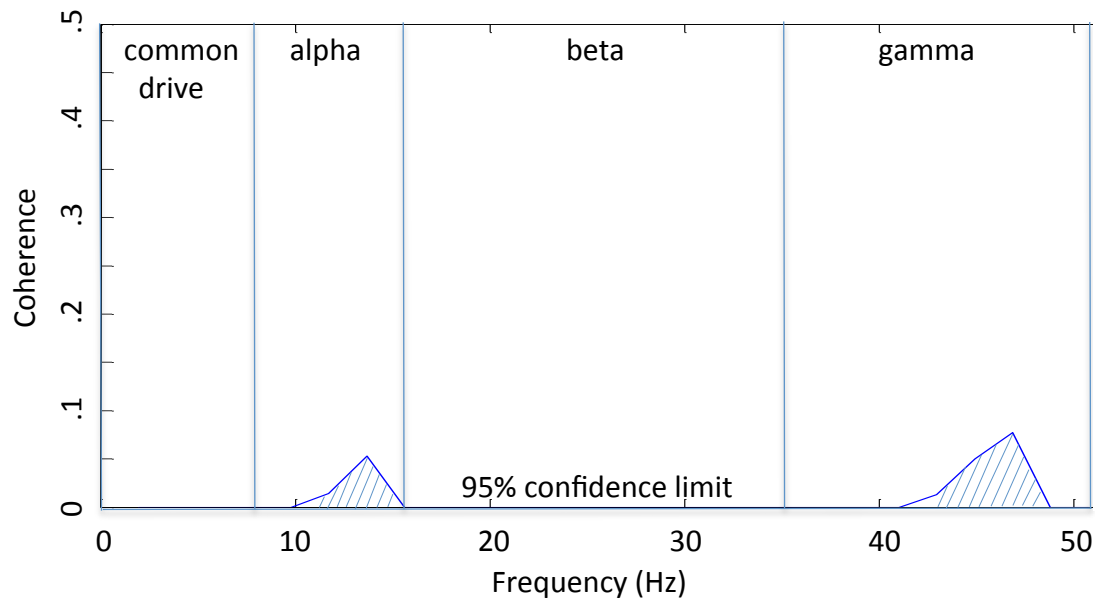


Figure 4.13. Coherence plot versus frequency 0-51 Hz of the task condition 1 (bbEx) in one subject. EMG data is averaged from 10 trials and used as inputs to calculate the coherence of type 0. When z-transform is applied, the coherence integral is calculated above 95% confidence limit only, as shaded upward-diagonal. The coherence graph below 95% confidence limit is removed in the figure. Four frequency bands, in Hz, are shown as common drive 0-8, alpha 8-16, beta 16-35, gamma 35-51.

Figure 4.17 presents the form of raw EMG signals for high versus low coherence, in ramp phase within the same subject. Note that the raw EMGs were used for coherence calculation, as we mentioned earlier with explanation along with equations. This presentation is to help understanding how raw EMGs look like for those signal pairs with strong and weak coherences. The peaks of two raw EMGs are well aligned for strong coherence while poorly aligned for weak coherence.

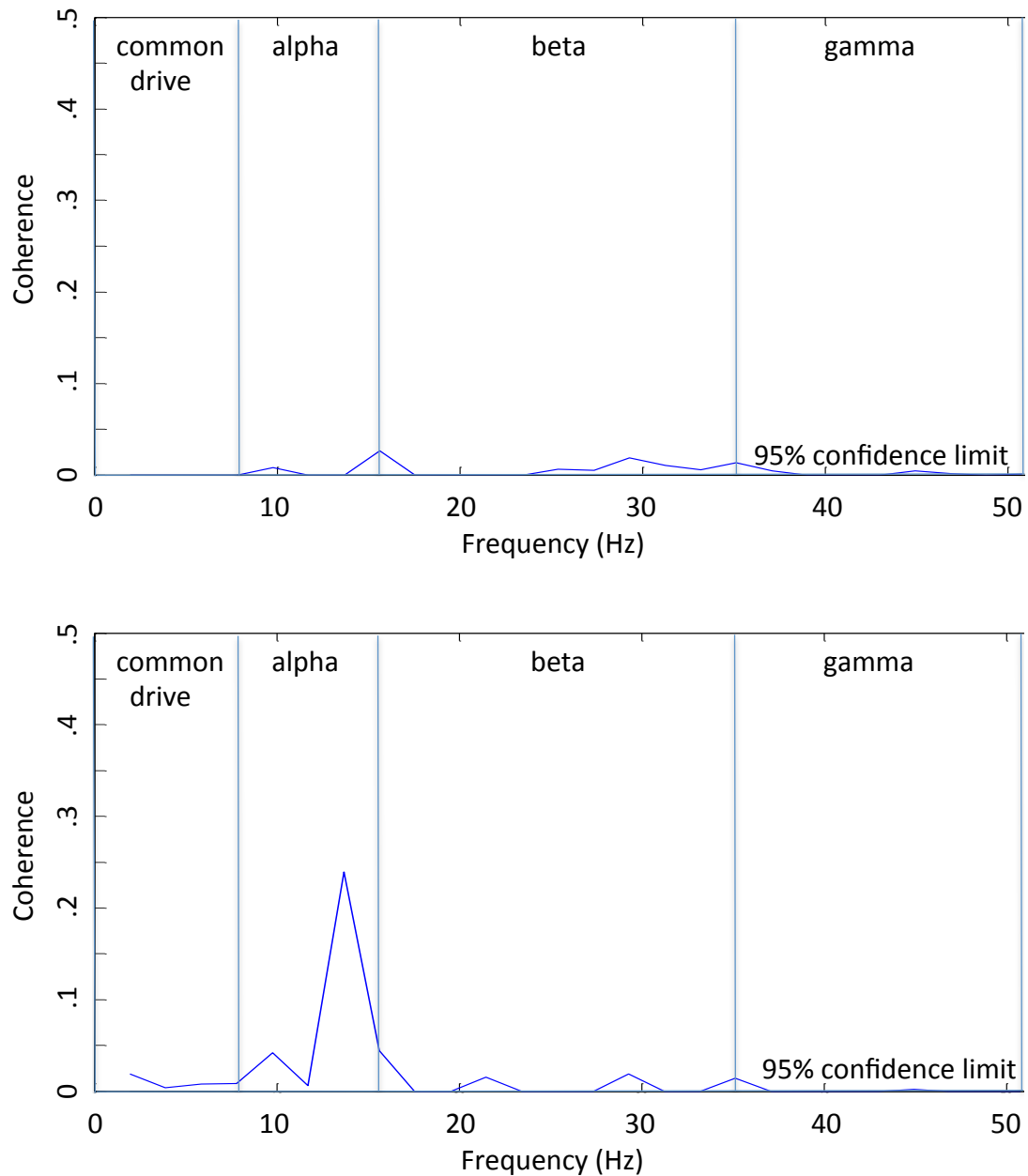


Figure 4.14. Coherence plot of the task condition 2 (Ex) in one subject (upper row) and of the task condition 3 (ExFree) in one subject (lower row). EMG data is averaged from 10 trials and used as inputs to calculate the coherence of type 0. Z-transform coherence integral is calculated above 95% confidence limit only. The coherence graph below 95% confidence limit is removed in the figure.

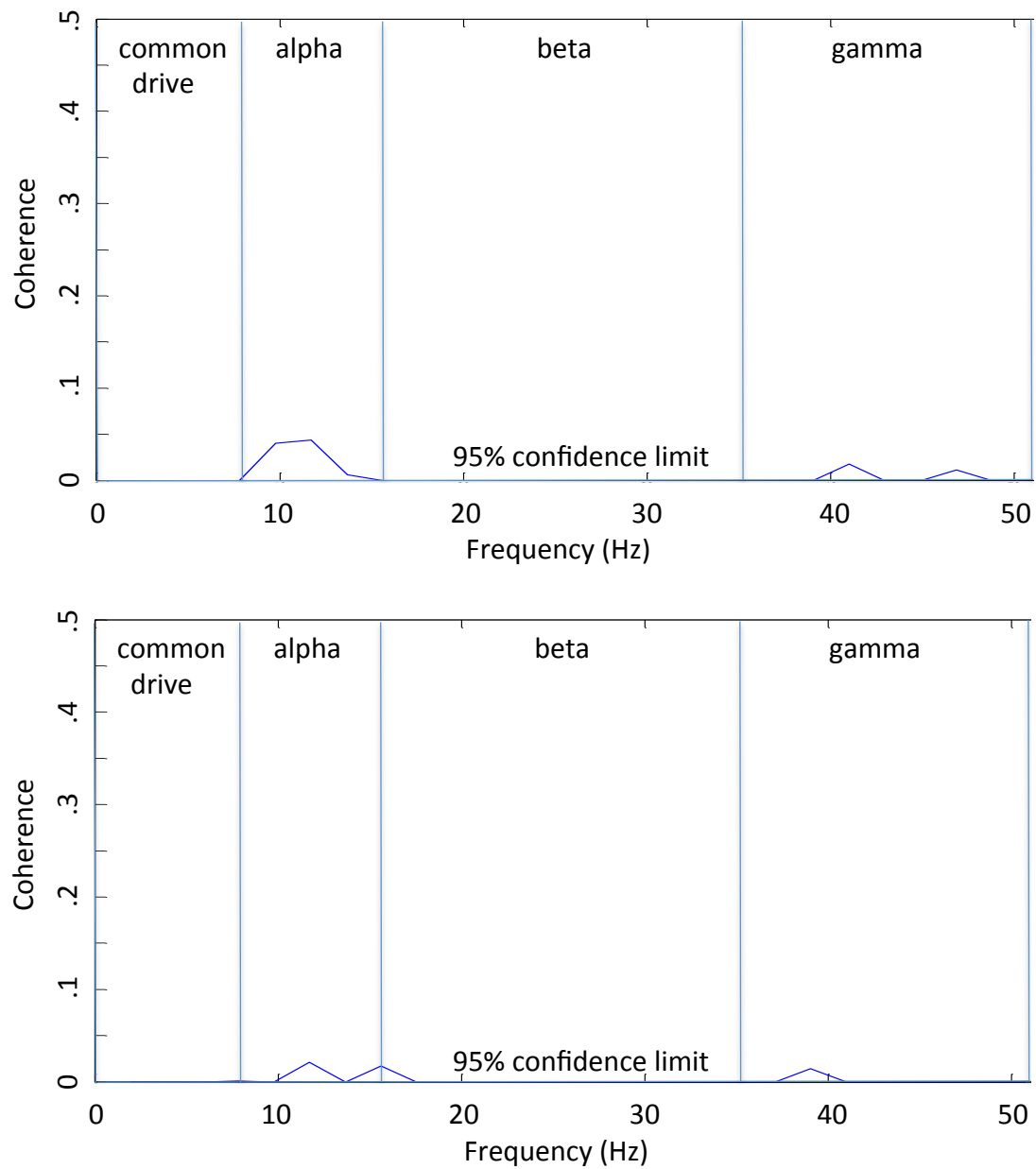


Figure 4.15. Coherence plot of the task condition 4 (bbFl) in one subject (upper row) and of the task condition 5 (Fl) in one subject (lower row). EMG data is averaged from 10 trials and used as inputs to calculate the coherence of type 0. Z-transform coherence integral is calculated above 95% confidence limit only. The coherence graph below 95% confidence limit is removed in the figure.

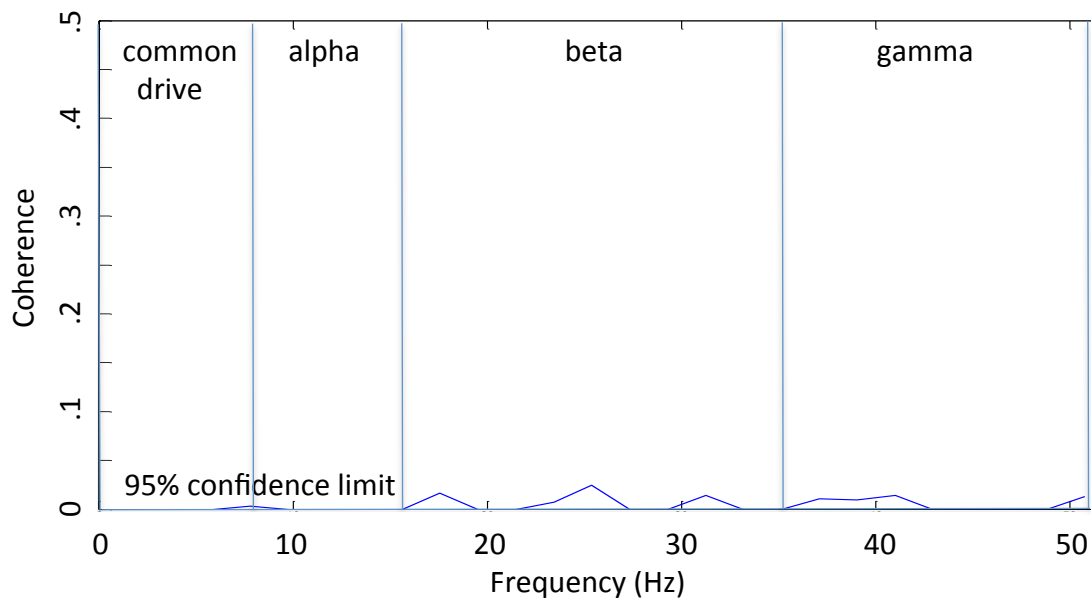


Figure 4.16. Coherence plot of the task condition 6 (FIFree) in one subject. EMG data is averaged from 10 trials and used as inputs to calculate the coherence of type 0. Z-transform coherence integral is calculated above 95% confidence limit only. The coherence graph below 95% confidence limit is removed in the figure.

#### 4.4 Discussions

With extension task, the tiny free bilateral limb movement produces best agonist coherence in ramp phase compared to the isometric conditions with or without ball balance visual feedback. The coherence in ramp phase on the antagonist exposes a similar trend but smaller in all task conditions. In contrast, the flexion task conditions did not exhibit the significant distinction in ramp and stay phases on both agonist and antagonist muscle pairs. Additionally, the ramp coherence in flexion group is much smaller compared to the extension.

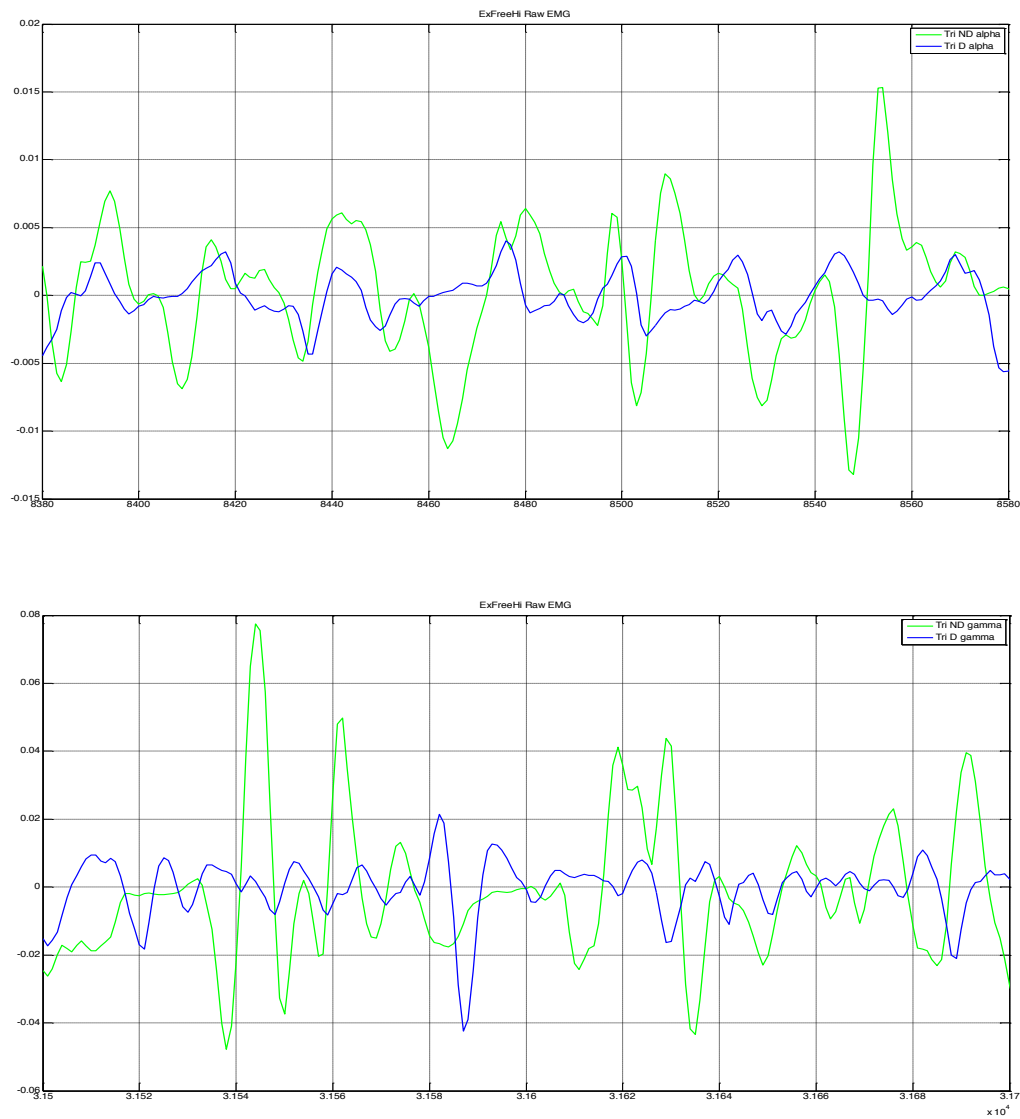


Figure 4.17. Partial presentation (6% of the the ramp phase) of non-dominant and dominant limb time-domain raw EMGs in high/low coherence for the same subject: alpha-pass-filtered EMGs for a high coherence integral of .6 (upper); gamma-pass-filtered EMGs for a low coherence integral of .02 (lower).

Alpha band coherence is the largest part of total coherence. The Free (tiny movement) condition has highest alpha and total coherence compared to other two conditions (isometric). This is aligned with the study of Datta and Stephens, 1990, Smith et al., 1999.

Alpha band coherence can be further understood by the review of motor unit. Motor unit (MU) is a part of the neuromuscular system that contains an anterior horn motor neuron, its axon, and all of the skeletal muscle fibers that it innervates (Buchtal et al., 2012). The single motor neuron is of alpha-type which is a lower motor neuron of the brainstem and spinal cord. It is also responsible for initiating muscle contraction. All of the motor units within a muscle are considered a motor pool. The central nervous system combines two control mechanisms to regulate the force a single muscle produces. The first is recruitment and the second is rate coding. Progressive activation of muscle fibers by successive recruitment of motor units to achieve increased intensity of muscle contraction is known as motor unit recruitment. During a muscle contraction, small motor units (slow conduction velocity/fewest fibers) are excited first and as the strength of muscle contraction increases, larger motor and larger units (fast conduction velocity/most fibers) are recruited last (Milner-Brown et al., 1973). As a result, the number of motor units recruited increases. This process is called Henneman's size principle. This principle is important because it allows gradual increase of muscle force during weak contractions whereas changes are larger when large force is desired (Robinson et al., 2009). Small motor units are fatigue resistant and they provide fine control for most activities. On the other hand, big units fatigue easily and are only used occasionally in instances such as running when fine control is not required. Muscle fibers belonging to one motor unit overlap spatially other motor units in

micro bundles of three to fifteen muscle fibers. Therefore, separate motor units contract in support of one another due to this inter-digitation. The activation of one motor neuron will result in a weak but distributed muscle contraction. The activation of more motor neurons will result in more muscle fibers being activated, and therefore a stronger muscle contraction. It is important to note that different motor units are driven asynchronously by the spinal cord so that contraction alternates among motor units one after another and as a result a smooth contraction is provided even at low frequencies (Guyton, 2010). MU firing produces a short mechanical twitch about four milliseconds in duration. Subsequent firing is continuously required in order to sustain a contraction.

Contractile and metabolic properties of the motor units can be divided into three types (McArdle et al., 2000; Plowman et al., 1998). SO (I) is denoted for slow twitch fibers innervated by alpha 2 motor neurons, smaller of the two  $\alpha$  motor neurons, and these fibers rely mainly on oxidative pathways. FG (IIb) is denoted for fast twitch fibers innervated by alpha 1 motor neurons, larger of the two  $\alpha$  motor neurons, and these fibers rely mainly on glycolytic pathways. FOG (IIa) is denoted for fast twitch fibers which have higher excitation threshold and faster conduction velocity, and they possess characteristics like SO and FG.

Distribution of fiber types in several muscles may help interpret our study results, biceps brachii is on average 52% SO, 33% FOG, 14% FG (this is similar to vastus lateralis of the thigh), soleus may have as much as 85% SO, triceps brachii may have as few as 33% SO, and 32% FOG, 34% FG (Sanchis-Moysi et al., 2010, McArdle et al., 2000, Plowman et al., 1998).

Muscles of fine control (smaller motor units) have a stronger cortico-spinal projection than more postural muscles (Penfield and Rasmussen, 1952). The synchrony of motor unit discharge is greater when involving fine movement. It seems that cortico-spinal neurons supply the shared input to motoneurons necessary to generate the observed synchrony of discharge (Datta and Stephens, 1980). The thought of shared cortico-spinal input to motor neurons is supported by work in monkeys showing that single cortico-spinal neurons project to motor neurons innervating several intrinsic hand muscles (Lemon, 1990).

The triceps motor units, of which their fibers are distributed as 1/3 SO and 2/3 FOG-FG, were in connected with the alpha coherence at fine movement. We may conclude that during force transition (ramp phase) of our slow tiny movement of elbow extension task, the alpha band coherence plays the key role. (One may predict: The biceps with 2/3 SO and 1/3 FOG-FG, associated with 2/3 slow motor units and 1/3 fast motor units, may require more activation of the slow-conducting nerve fibers, thus a slower force transition should be applied within a fine movement elbow flexor task. If MVC of flexor is greater than extensor, the applied loading level should be probably higher compared to that of extensor.)

The beta band coherence, which is presumably originated from the primary cortex, may be related to both the compliance of the object movement and the isometric contraction. More importantly, this band is coherent with the motor unit firing (represent the alpha band). Furthermore, this band coherence is probably mediated by fast corticospinal axons and their monosynaptic connections to spinal motor neurons. Hence it seems that the increased cortico-muscular coherence is somehow related to the process of learning the motor task. The primary



motor cortex is likely involved in the early consolidation of motor memory and that it is essential in the early stages of acquisition of novel motor tasks.

The oscillatory neuronal network also includes the descending output neurons of the motor cortex (pyramidal tract neurons), and cortical oscillations have been shown to be coherent with oscillatory EMG activity in arm and hand muscles (Kilner et al., 2000; Conway et al., 1995; Baker et al., 1997; Salenius et al., 1997; Hari and Salenius, 1999; Kilner et al., 1999). The investigations showed that coherence was particularly marked during steady grip of a compliant, spring-like load (Baker et al., 1997, 1999). The changes in oscillatory synchronization in the 15–30 Hz bandwidth between human motor cortex and hand muscles that varies according to the time course of the task and the level of compliance of the gripped object. Interacting with such springy objects, which are a common feature of everyday life (spring clips, bottles of shampoo, etc.), requires precise coordination of both digit position and grip force. Interestingly, the synchronization difference between brief and protracted hold periods was not present for cortex–muscle coherence during grasp under isometric conditions, once again underlining the importance of coherent oscillations during grasp of compliant objects.

Electroencephalogram–electromyogram (EEG–EMG) coherence showed that the cortical signal recorded from over primary motor cortex interacts with the muscle signal at ~20 Hz (Farmer et al., 2007, Conway et al., 1995; Salenius et al., 1997; Halliday et al., (1998); Brown et al., 1998; Mima and Hallett, 1999). Salenius et al., 1997 stated the relationship between beta band and alpha band coherence. The 15- to 33-Hz MEG signals, originating from the anterior

bank of the central sulcus, i.e., the primary motor cortex, were coherent with motor unit firing in all subjects and for all muscles (including triceps and biceps) during isometric contraction. They suggested that the motor cortex drives the spinal motoneuronal pool during sustained contractions.

Perez et al., 2004 observed an increased excitability of the cortical representation of the tibialis anterior muscle following the visuo-motor task. Similar findings have been reported in a number of studies for the hand area of the motor cortex (Pascual-Leone et al., 1994, 1995; Elbert et al., 1995; Classen et al., 1998; Lotze et al., 2003; Perez et al., 2004; Jensen et al., 2005). It has been suggested that corticomuscular coherence is probably mediated by fast corticospinal axons and their monosynaptic connections to spinal motoneurons (Farmer et al., 1993; Conway et al., 1995). It therefore seems more likely that the increased cortico-muscular coherence is somehow related to the process of learning the motor task. There is good evidence that the primary motor cortex is involved in the early consolidation of motor memory and that it is essential in the early stages of acquisition of novel motor tasks (Muellbacher et al., 2001, 2002). The available data also suggest that increased cortical excitability and the increased representation of specific muscles in the early stages of motor learning in all likelihood reflect this early acquisition and consolidation (Muellbacher et al., 2001, 2002; Pascual-Leone et al., 1995).

## 4.5 Conclusions

The results showed that only elbow extension task (but not flexion), exhibited a significant difference trend cross the three conditions. The combined-alpha-beta-gamma (8-51 Hz) total coherence of condition 3 (tiny free movement) is highest compared to the other two conditions 1 and 2 (isometric with/without ball-balance). Within this trend, the coherence is largest with agonist muscle pairs during the ramp of movement (ANOVA  $p=.006$ ). The same trend of coherence exists but smaller at three other muscle pairs. Those are the antagonist pairs during ramp (ANOVA  $p=.004$ ), and during stay (ANOVA  $p=.008$ ) and the agonist-antagonist pairs during ramp (ANOVA  $p=.035$  and  $p=.002$ ). The t-test p-value of these significant ANOVA always produced the significant trend in which the free condition was with best coherence. Two bands of alpha (8-16 Hz) and beta (16-35 Hz) are major contributions to the total coherence, which represents the motor unit discharge and primary cortex, respectively.

## **Chapter 5**

### **Conclusions**

CQF and CBD, the two inertia compensation algorithms we developed, reduced interaction forces (up to 64% in directions with highest intrinsic impedance) and anisotropy (up to 74% in tangential force anisotropy) compared to no compensation condition. CQF outperformed CBD in decreasing tangential forces in the high-impedance directions and in decreasing anisotropy during high-speed movements. CQF algorithm can be potentially embedded into other control algorithms to minimize the intrinsic impedance and anisotropy of the InMotion2, and might be extended to other robots with analogous architectures. Studies that do not implement algorithms of this type risk contamination of results by the inherent dynamics of the InMotion2 robot.

The interlimb coupling study on subjects with stroke was designed to determine the type of bimanual robot-based loading that could help improving the paretic limb performance. The compensation algorithms were included for precise loading of the non-paretic limb. The novel resistive loading with a robotic device produced benefits in the paretic limb of increase of speed, acceleration, EMG and earlier onset of peak velocity compared to the bilateral no-load condition. Increasing load level in the non-paretic limb improved these measures in the paretic limb within resistive loading under spring and constant force patterns. The constant loading appears to be the

most effective of the bimanual conditions, and likely is most effective in subjects with slow movement speeds more so than subjects with low FMA score. No effects were seen in higher functioning subjects with movement speeds that approached normal levels.

The interlimb coupling study also showed that larger gains in the paretic limb occurred in trials with better between-limb movement synchrony. That led us to the third study where we tested several levels of tight and loose task constraints on bilateral synchrony during bilateral elbow extension and flexion isometric force generation. We used EMG coherence in the frequency domain to assess the degree of coupling between muscles in the two limbs. We found that with isometric elbow extension force generation, the haptic coupling condition (called the tiny free bilateral limb movement condition in the chapter) produces best agonist coherence in the ramp phase compared to the low difficulty visual coupling and high difficulty visual coupling conditions (called isometric conditions with or without ball balance visual feedback in the chapter). The coherence in the ramp phase in the antagonists exposes a similar trend but values were smaller in all task conditions. In contrast, the flexion task conditions did not exhibit the significant distinction in ramp and stay phases on both agonist and antagonist muscle pairs. Additionally, the ramp coherence in flexion group muscles was much smaller compared to the extension muscles. The different neural control mechanisms for flexors and extensors may explain this difference in coherence, and should be an area of future study.

Breaking down the frequency band into each of the smaller bands may help predict the neuromuscular origin of the coherence. The results state that alpha band and beta band are the major parts contributing to the total coherence. Coherence in each of these two bands mirrors

the total coherence values reported in agonists during the ramp phase. Alpha band coherence, believed to be related to motor unit discharge, dominates about two-third of the total coherence. On the other hand, beta band coherence contributes around one-third of the total coherence. Beta band is related to corticomuscular coherence, during force transitions rather than force holding. Coherence in this band is believed to have its origins in activation of the primary motor cortex, and is the primary target for bilateral training of stroke patients.

Future works on the bimanual task on subjects with stroke would be recommended as,

- Bilateral elbow extension focused at slow transition of force (ramp phase) with haptic coupling (like two hands grasping same object) setup would be recommended. The required force generation of paretic and non-paretic limbs could be different, as this did not affect the coherence results in healthy control subjects.

- Tasks of high speed bimanual reaching-out (single muscle, shared reaching target) movements with constant loading on non-paretic limb should include the intuitive simple visual GUI and should be applied on slow movement subjects rather than the ones with low FMA score.

- The above task paradigms can be implemented to build a robot-assisted bimanual training device, or can be extended as part of the ARMin III or InMotion2 robots. Progressive force generation and progressive resistive loading are suggested through out the training period to gradually see the paretic performance improvement.

## Appendix

### *The Experimental Desk and Chair*



Figure A.1. The table and adjustable-height chair with straps which were used for all three study chapters.

### *MicroFET2 Muscle Tester*

This muscle tester is designed specifically for precise force evaluation of the muscle testing measurements. This portable and hand-held unit is battery operated, weighs less than a pound, and is ergonomically designed to fit comfortably in the palm of human hand. The tester was said of using sophisticated digital technology to achieve its high degree of accuracy and reliability. Strain gauges elements in the transducer could react independently to measure

external forces from multiple angles. The patented structure enables the gauge to detect even subtle changes in force, regardless of the applied force direction. This feature is really important for us, since we use this sensor for all our three chapters. The first chapter, it was used to calibrate the force transducer of the robot mounted underneath the handle. The second chapter with interlimb coupling, this sensor was used to measure MVC. The investigator of the study hold the sensor against the subject's limb extension at a certain arm posture.



Figure A.2. Hand-held force sensor microFET2 and its neuromuscular-musculoskeletal usage in force evaluation and testing (FET). Selectable units of measure can be pounds (lbs.), Newtons (N), kilogram-force (kgf). The threshold is variable and settable from .8 lb to 300 lbs. Information from the gauge is displayed in two LCD windows, peak forces and duration/sec.



During the force pushing out of the subject, the contact between human hands can be suspended back and forth in the air and so the above patented feature of the sensor was important to precisely collect peak force of the contraction. The third chapter with coherence study, microFET was placed on the fixed handle and so the subject extended their fist of the hand against it. The above feature of the transducer again plays important role to accurately received MVC.

### ***Resistive Loading Test***



The constant loading condition was tested by both the program data collection of force and distance, and manually verified by the microFET2 sensor along the movement path. At the time constant loading was tested, the ATI force transducer was not needed to be calibrated. The same thing was applied to Spring loading in the two movement paths of the interlimb coupling study.

Figure A.3. The ATI force sensor.

The ATI force sensor attached underneath the robot handle was calibrated. The following calibration equations were also used for the algorithm chapter.

Once data in the below table was collected,  $F_{xC_{true}}$  and  $F_{yC_{true}}$  was calculated by the collected data in comparison with force assigned by the program. The force measured by microFET2 was used to verify the assigned force.

$$\begin{aligned}
 Fx C_{true} &= (4.519774) FxC + 7.118644 \\
 Fy C_{true} &= (-4.25532) FyC + 14.46809
 \end{aligned}
 \tag{A.1}$$

Table A.1. ATI calibration

Force assigned in the program (N)	Force measured by microFET2 (N)	Experimental collection of FxC at Simulink scope 10	Experimental collection of FyC at Simulink scope 08
0	0	-1.23	3.4
5	5.1	-0.06	2.3
10	9.9	0.7	1
20	20.2	2.85	-1.3
-20	-19.9	-6	8.1

Several random single values were assigned to the implementation of inertia loading program and the robot was operated to test the Newton equation  $F=ma$ . For example in Simulink program, assigning mass parameter by 1.5 kg and then the program was translated and downloaded. The operator holds the robot handle and recorded a peak acceleration of  $7.95 \text{ m/s}^2$  together with peak force extraction of 12 N. Similarly another tested trial was done with 1.1 kg assignment, and the corresponding correct values of peak acceleration and peak force were  $5.4 \text{ m/s}^2$  and 6 N, respectively, etc...

Testing of direction of the perturbed force by the robot was done based on components of acceleration ( $a_x, a_y$ ) and force ( $F_x, F_y$ ). The experimental force versus acceleration were graphed following the trajectory from start to stop locations.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = - \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix} \quad (\text{A.2})$$

### *Delsys Bagnoli EMG Unit*

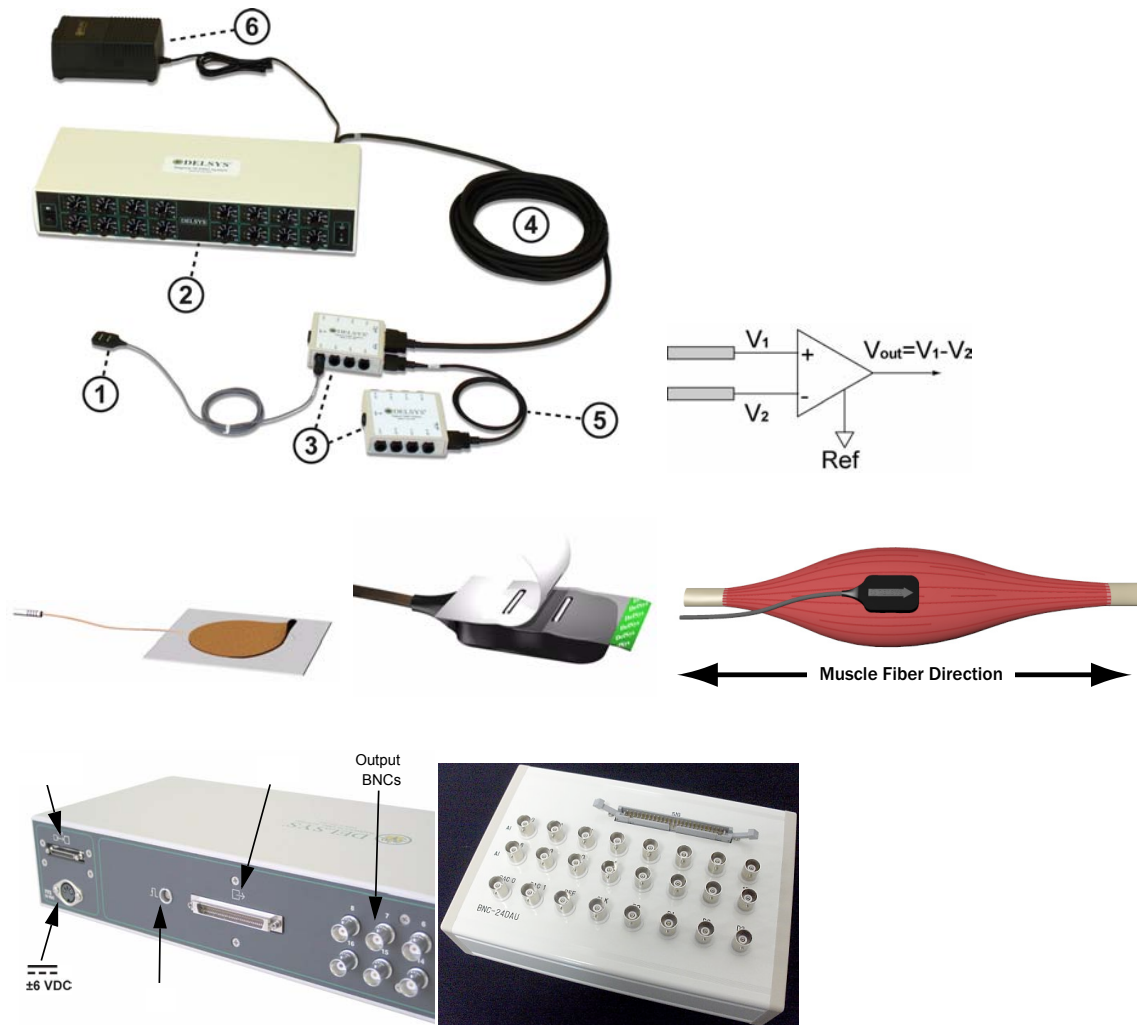


Figure A.4. The Bagnoli-16 was shown, with single differential surface EMG sensor (1) being used in connection with the input units (3), and the internal circuit of the EMG sensor. The middle row shows reference electrode, the adhesive sensor interface and electrode mounting notice for biceps and triceps muscles. The third row shows that Bagnoli's output BNCs (left) are used to connect output signals to the NDI Optotrak ODAU unit (right).

## FMA Scoresheet

### APPENDIX B FUGL-MEYER ASSESSMENT OF PHYSICAL PERFORMANCE

Motor Function Upper Extremity				
TEST	ITEM	SCORE		SCORING CRITERIA
		Pre	Post	
I. Reflexes	Biceps			0-No reflex activity can be elicited
	Triceps			2-Reflex activity can be elicited
II. Flexor Synergy	Elevation			0-Cannot be performed at all
	Shoulder retraction			1-Performed partly
	Abduction (at least 90°)			2-Performed faultlessly
	External rotation			
	Elbow flexion			
	Forearm supination			
III. Extensor Synergy	Shoulder add./int. rot.			0-Cannot be performed at all
	Elbow extension			1-Performed partly
	Forearm pronation			2-Performed faultlessly
IV. Movement combining synergies	Hand to lumbar spine			0-No specific action performed 1-Hand must pass anterior superior iliac spine 2-Performed faultlessly
	Shoulder flexion to 90°, elbow at 0°			0-Arm is immediately abducted, or elbow flexes at start of motion 1-Abduction or elbow flexion occurs in later phase of motion 2-Performed faultlessly
	Pronation/supination of forearm with elbow at 90° & shoulder at 0°			0-Correct position of shoulder and elbow cannot be attained, and/or pronation or supination cannot be performed at all 1-Active pronation or supination can be performed even within a limited range of motion, and at the same time the shoulder and elbow are correctly positioned 2-Complete pronation and supination with correct positions at elbow and shoulder
V. Movement out of synergy	Shoulder abduction to 90°, elbow at 0°, and forearm pronated			0-Initial elbow flexion occurs, or any deviation from pronated forearm occurs 1-Motion can be performed partly, or, if during motion, elbow is flexed, or forearm cannot be kept in pronation 2-Performed faultlessly
	Shoulder flexion 90-180°, elbow at 0°, and forearm in mid-position			0-Initial flexion of elbow or shoulder abduction occurs 1-Elbow flexion or shoulder abduction occurs during shoulder flexion 2- Performed faultlessly
	Pronation/supination of forearm, elbow at 0° and shoulder between 30-90° of flexion			0-Supination and pronation cannot be performed at all, or elbow and shoulder positions cannot be attained 1-Elbow and shoulder properly positioned and pronation and supination performed in a limited range 2-Performed faultlessly
VI. Normal reflex activity	Biceps and/or finger flexors and triceps (This item is only included if the patient achieves a maximum score on all previous items, otherwise score 0)			0-At least 2 of the 3 phasic reflexes are markedly hyperactive 1-One reflex is markedly hyperactive, or at least 2 reflexes are lively 2-No more than one reflex is lively and none are hyperactive

TEST	ITEM	SCORE		SCORING CRITERIA
VII. Wrist	Stability, elbow at 90°, shoulder at 0°			0-Patient cannot dorsiflex wrist to required 15° 1-Dorsiflexion is accomplished, but no resistance is taken 2-Position can be maintained with some (slight) resistance
	Flexion/extension, elbow at 90°, shoulder at 0°			0-Volitional movement does not occur 1-Patient cannot actively move the wrist joint throughout the total ROM 2-Faultless, smooth movement
	Stability, elbow at 0°, shoulder at 30°			0-Patient cannot dorsiflex wrist to required 15° 1-Dorsiflexion is accomplished, but no resistance is taken 2-Position can be maintained with some (slight) resistance
	Flexion/extension, elbow at 0°, shoulder at 30°			0-Volitional movement does not occur 1-Patient cannot actively move the wrist joint throughout the total ROM 2-Faultless, smooth movement
	Circumduction			0-Cannot be performed 1-Jerky motion or incomplete circumduction 2-Complete motion with smoothness
VIII. Hand	Finger mass flexion			0-No flexion occurs 1-Some flexion, but not full motion 2-Complete active flexion (compared with unaffected hand)
	Finger mass extension			0-No extension occurs 1-Patient can release an active mass flexion grasp 2-Full active extension
	Grasp I - MCP joints extended and proximal & distal IP joints are flexed; grasp is tested against resistance			0-Required position cannot be acquired 1-Grasp is weak 2-Grasp can be maintained against relatively great resistance
	Grasp II - Patient is instructed to adduct thumb, with a scrap of paper interposed			0-Function cannot be performed 1-Scrap of paper interposed between the thumb and index finger can be kept in place, but not against a slight tug 2-Paper is held firmly against a tug
	Grasp III - Patient opposes thumb pad against the pad of index finger, with a pencil interposed			0-Function cannot be performed 1-Pencil interposed between the thumb and index finger can be kept in place, but not against a slight tug 2-Pencil is held firmly against a tug
	Grasp IV - The patient should grasp a can by opposing the volar surfaces of the 1st and 2nd digits.			0-Function cannot be performed 1-A can interposed between the thumb and index finger can be kept in place, but not against a slight tug 2-Can is held firmly against a tug
	Grasp V - The patient grasps a tennis ball with a spherical grip or is instructed to place his/her fingers in a position with abduction position of the thumb and abduction flexion of the 2nd, 3rd, 4th & 5th fingers			0-Function cannot be performed 1-A tennis ball can be kept in place with a spherical grasp but not against a slight tug 2-Tennis ball is held firmly against a tug
IX.Coordination/ Speed- Finger from knee to nose (5 repetitions in rapid succession)	Tremor			0-Marked tremor 1-Slight tremor 2-No tremor
	Dysmetria			0-Pronounced or unsystematic dysmetria 1-Slight or systematic dysmetria 2-No dysmetria
	Speed			0-Activity is more than 6 seconds longer than unaffected hand 1-(2-5.9) seconds longer than unaffected hand 2-Less than 2 seconds difference
Upper Extremity Total				Maximum = 66

Table A.2. FMA scoresheet (this and last page)

## Outlier Quartiles

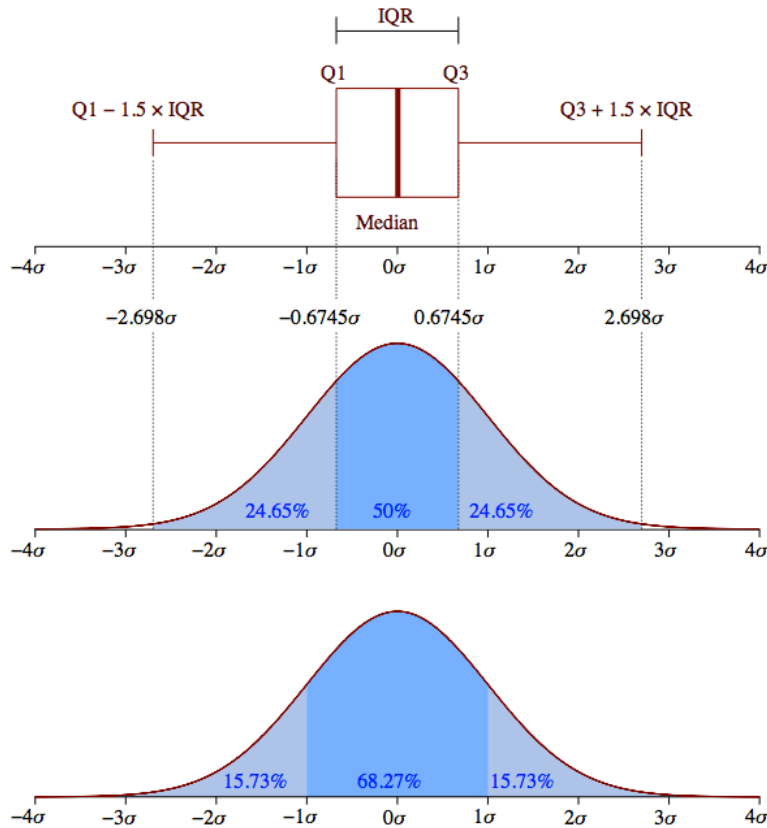


Figure A.5. Boxplot with with quartiles and an interquartile range and a probability density function of a normal  $N(0, 1\sigma^2)$  population

## Cross Correlation Demonstration

The cross correlation equation is partially sourced from Mathworks, at <http://www.mathworks.com/help/signal/ref/xcorr.html>. There are two functions to calculate cross correlation in Matlab, one with Signal Processing Toolbox named `xcorr` and another with Econometrics Toolbox named `crosscorr`. One difference between them at first glance is what they return and what options they allow. Signal Processing Toolbox just returns the correlations and

lags while allowing the user to specify a normalization method, while Econometrics Toolbox returns correlations, lags, and confidence bounds but does not allow the user to specify a normalization method. The other difference is about its internal processing with the input mean,  $c = \text{xcorr}(x - \text{mean}(x), y - \text{mean}(y), N, \text{'coeff'})$ ; while  $C = \text{crosscorr}(y, x, N)$ . Some may say, for `crosscorr`, the scaling does not ensure the XCF at lag zero is one. See more at: Box, G. E. P., Jenkins G. M., and Reinsel G. C. “Time Series Analysis: Forecasting and Control.” *3rd ed.* Upper Saddle River, NJ: Prentice-Hall (1994).

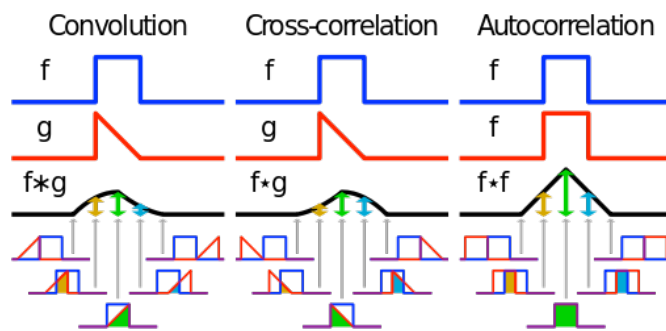


Figure A.6. Cross correlation demonstration [http://en.wikipedia.org/wiki/Cross\\_correlation](http://en.wikipedia.org/wiki/Cross_correlation)

## Fourier Transform

The mathematical Fourier transform is a to transform signals between time (or spatial) domain and frequency domain, which has vast applications in physics and engineering. The new function is then known as the Fourier transform and/or the frequency spectrum of the function  $f$ . The Fourier transform operation is reversible. Hence, given the function  $\hat{f}$  one can determine the original function,  $f$ . Most often,  $f$  is a real-valued function, and  $\hat{f}$  is complex valued, where a

complex number describes both the amplitude and phase of a corresponding frequency component. If  $f$  is a periodic function (for example, a continuous but not necessarily sinusoidal musical sound), the Fourier transform can be simplified to the calculation of a discrete set of complex amplitudes, called Fourier series coefficients. When a time-domain function is sampled to ease storage or computer-processing, it is possible to recreate a version of the original Fourier transform according to the Poisson summation formula, also known as discrete-time Fourier transform.

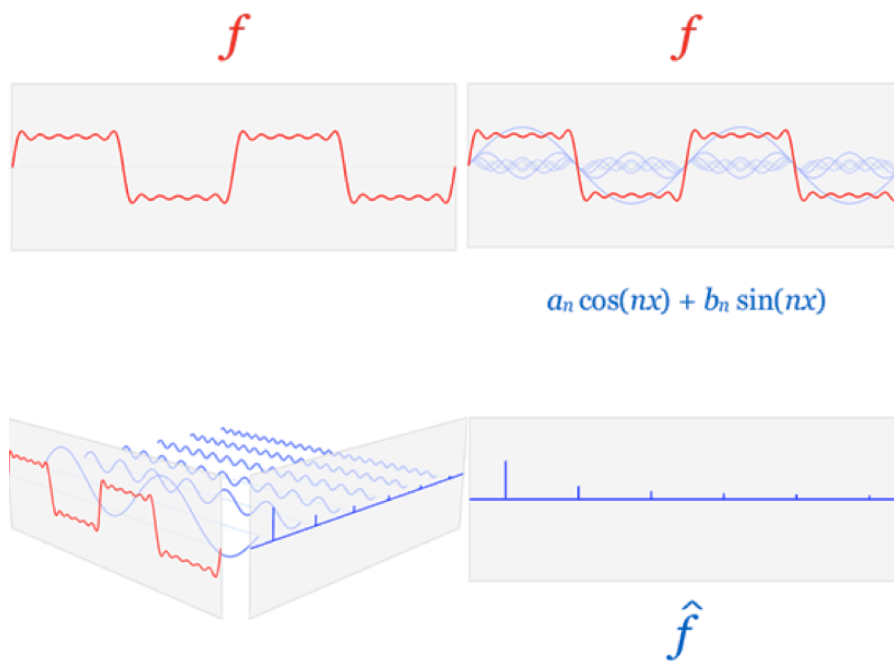


Figure A.7. Fourier transform relates the function  $f$  in time domain (red) to the one  $\hat{f}$  in frequency domain (blue). The component frequencies, spread across the frequency spectrum, are represented as peaks in the frequency domain.



The motivation for the Fourier transform comes from the study of Fourier series. In the study of the series, complicated but periodic functions are written as the sum of simple waves mathematically represented by sines and cosines. The Fourier transform is an extension of the Fourier series that results when the period of the represented function is lengthened and allowed to approach infinity (textbook of Taneja, 2008). With the properties of sine and cosine, it is feasible to restore the amplitude of each wave in a Fourier series using an integral. It is convenient to use Euler's formula which states that  $e^{2\pi i\theta} = \cos(2\pi\theta) + i\sin(2\pi\theta)$ , to write Fourier series in terms of the basic waves  $e^{2\pi i\theta}$ . This can help to simplify plural formulas involved. Re-writing sines and cosines as complex exponentials makes it necessary for the Fourier coefficients to be complex valued. The usual interpretation of this complex number is that it gives both the amplitude of the wave present in the function and the phase (or the initial angle) of the wave.

### ***Calibration of Tekscan Force Sensors***

The four thin-film Tekscan FlexiForce sensors were separately calibrated, each at three times (then the average was used), at our laboratory temperature of 25°C. The sensor was recommended by its manufacturer to be used at the same temperature as it was calibrated. We conducted all the experiments in the same lab at the same temperature setting.

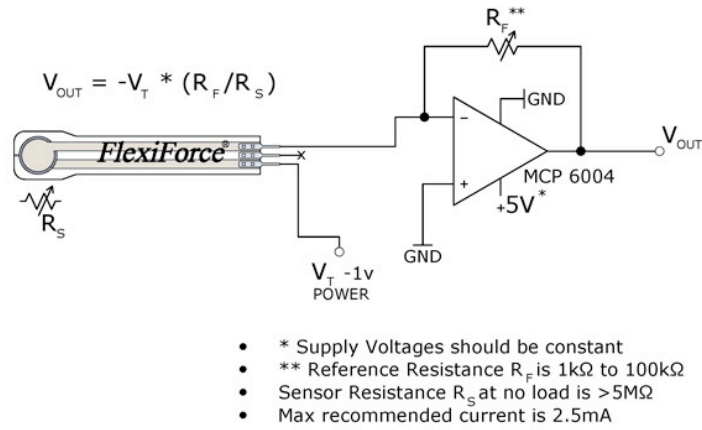


Figure A.8. Flexiforce A401, max load 110 N, ultra-thin .008 in., sensing area of 1 inch diameter, response time less than 5 microseconds and the sensor Phidget adaptor with its circuit diagram.

For the elbow extension task, the left-push and right-push sensors both had the same calibration equation:

$$N = (25.00)V + 3.00 \quad (A.3)$$

For the elbow flexor task, the left-pull and right-pull sensors both had the same calibration equation:

$$N = (24.2874)V + 3.1688 \quad (A.4)$$

Table A.3. FlexiForce calibration

Peak force apply (recorded by microFET2) (N)	Output Volts from the above amplifier circuit (recorded via Matlab and NI DAQ USB 6210)	Newton = $24.2874 \cdot \text{volt} + 3.1688$ verified
0	0.02	3.16
20.9	0.7375	21.08
39.5	1.333	35.54
60	2.193	56.4
82.3	3.385	85.38
100.9	4.049	101.5

### *The Coherence Apparatus Mechanical Parts*

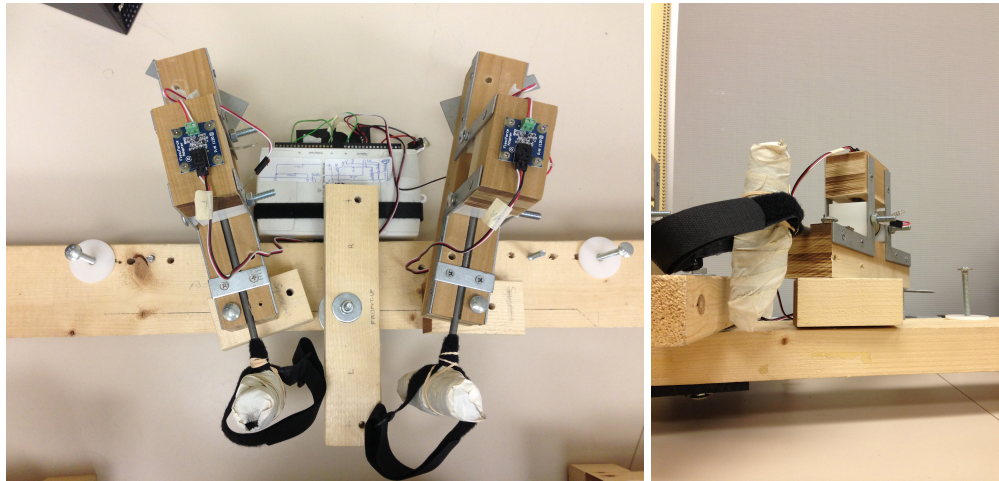


Figure A.9. Mechanical parts of the coherence apparatus

## Noraxon EMG Unit and Accessories

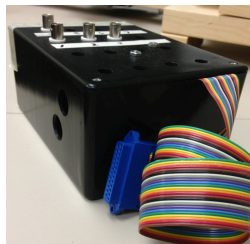


Figure A.10. Myosystem 1400A with standard 2/3-pin leads and dual/single electrodes, and the connection box was built to connect Myosystem 25-pin D-sub port to Optotrak ODAU BNC .

## National Instruments Data Acquisition

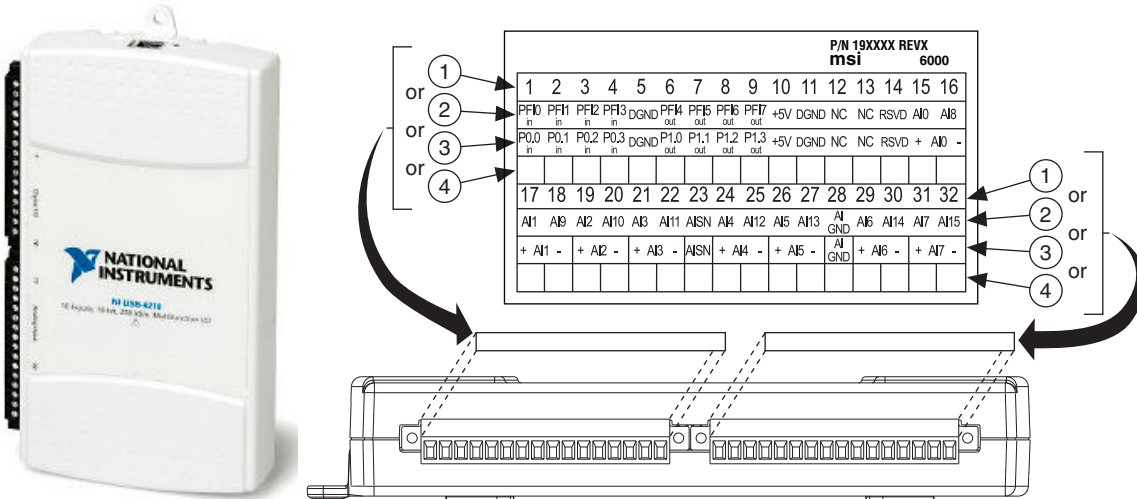


Figure A.11. National Instruments DAQ USB 6210 (up to 250 KHz) outlook with its I/O terminals. It is connected with PC via USB, works at sampling rate of 100 Hz for realtime processing of force, it triggers the Optotrak unit through a serial cable.

## Motor Unit

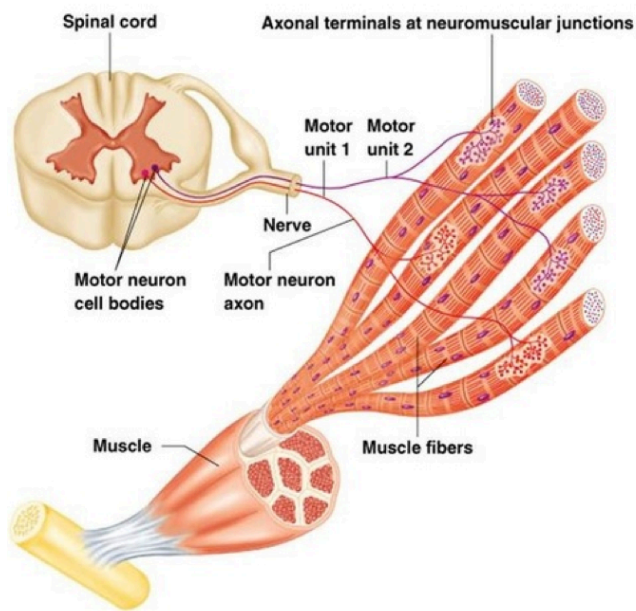


Figure A.12. Motor unit (Ronald et al., 2002).

### ► Properties of Motor Units

<b>Muscle Fibers</b>			
Twitch properties	Slow	Fast	
Metabolic properties	Oxidative	Oxidative/ glycolytic	Glycolytic
Name based on twitch and metabolic properties	SO	FOG	FG
Other nomenclature	ST, Type I	FTa, FTA, Type IIA	FTb, FTB, Type IIB
<b>Motor Neurons</b>			
Neuron type	$\alpha_2$	$\alpha_1$	$\alpha_1$
Neuron size	Small	Large	Large
Conduction velocity	Slow	Fast	Fast
Recruitment threshold	Low	High	High

Figure A.13. Properties of motor units (McArdle et al., 2000, Plowman et al., 1998)

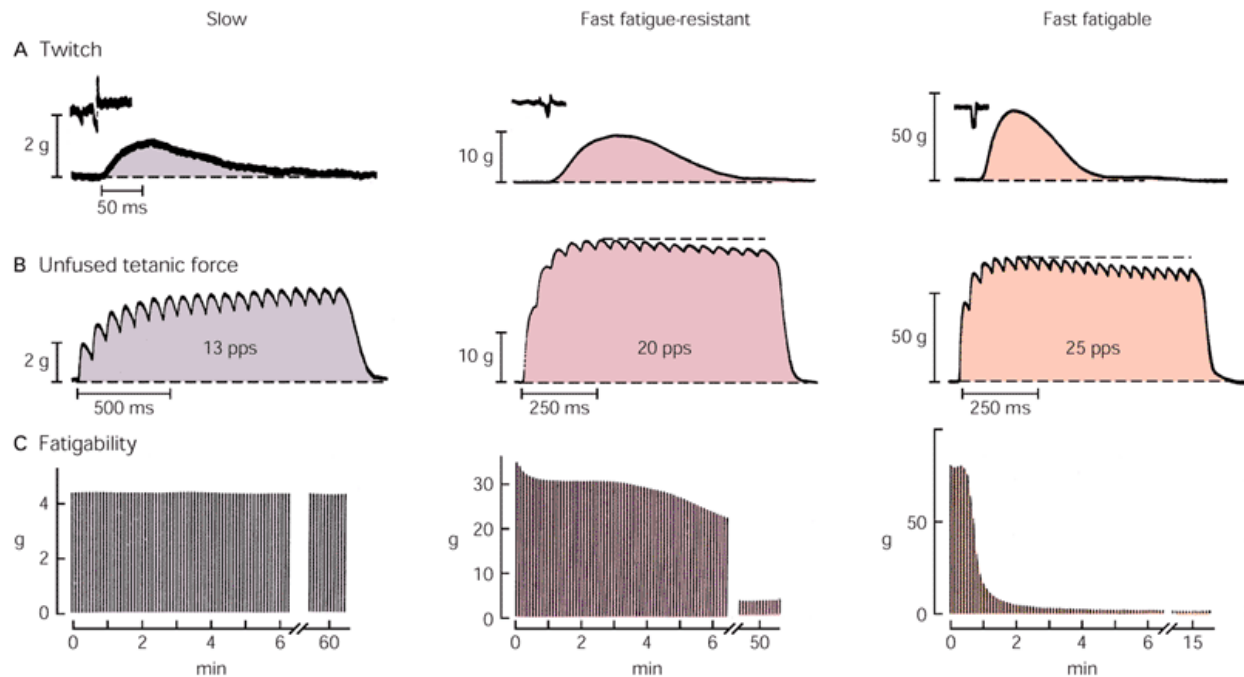


Figure A.14. Slow, fast fatigue-resistant, and fast fatigable motor units vary in twitch, tetanic force, and fatigability. (Burke et al., 1973.) A. Traces show the twitches of the three motor units. B. Unfused contractions produced by a train of stimuli at a rate typical for each type of motor unit. Fast units produce much larger twitch and tetanic forces than do slow units (vertical scale changed for each). C. Fatigability can be seen in records of the force produced by sustained activation. The motor units were activated by stimulus trains (40 pps) lasting 0.33 s and repeated every second. In the records shown here a single vertical line represents the force produced by one contraction, recorded at slow speed. In the slow unit the force remained essentially constant for over an hour of repeated stimulation. In the fast fatigable unit the force dropped abruptly after only a minute. The fast fatigue-resistant unit had substantial resistance to fatigue and the force declined slowly over many minutes; some residual force remained after 50 min.

### *The Coherence GUI Code*

```
function varargout = CohFgui(varargin)
% COHFGUI M-file for CohFgui.fig
% COHFGUI, by itself, creates a new COHFGUI or raises the existing
% singleton*.
%
% H = COHFGUI returns the handle to a new COHFGUI or the handle to
% the existing singleton*.
%
```

```
% COHFGUI('Property','Value',...) creates a new COHFGUI using the
% given property value pairs. Unrecognized properties are passed via
% varargin to CohFgui_OpeningFcn. This calling syntax produces a
% warning when there is an existing singleton*.
%
% COHFGUI('CALLBACK') and COHFGUI('CALLBACK',hObject,...) call the
% local function named CALLBACK in COHFGUI.M with the given input
% arguments.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

```
% Edit the above text to modify the response to help CohFgui
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',    mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @CohFgui_OpeningFcn, ...
    'gui_OutputFcn', @CohFgui_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before CohFgui is made visible.
function CohFgui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
```

```
% hObject   handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)
```

```
% Choose default command line output for CohFgui
handles.output = hObject;
```

```
global dist_UL; % between upper & lower lines of the guidance
dist_UL = 2.5;
```

```
global leftpush_p1;
global leftpush_p2;
global rightpush_p1;
global rightpush_p2;
leftpush_p1=25; %24.2874; % newton = p1*volt + p2;
leftpush_p2=3; %3.1688;
rightpush_p1=25; %31
rightpush_p2=3; % 1
```

```
global leftpull_p1;
global leftpull_p2;
global rightpull_p1;
global rightpull_p2;
leftpull_p1=24.2874;
leftpull_p2=3.1688;
rightpull_p1=24.2874;
rightpull_p2=3.1688;
```

```
global random_task_array;
global cond_line;
global preset_newton;
global newton;
global newton_right;
global flevel_max;
```

```
global flevel ;
global flevel_right;
```



```

global left_position;
global left_start_position;
global left_target_position;
global right_position;
% global right_start_position;
% global right_target_position;

global myclk;
myclk=1;

global ball_roll_ratio;
ball_roll_ratio=.3; % assigning smaller value will slow down ball rolling (easier)
global guidance_pos_reach_out_left;
global guidance_pos_reach_back_left;
global guidance_pos_reach_out_right;
global guidance_pos_reach_back_right;
global guidance_sq_count_reaching_out;
guidance_sq_count_reaching_out=0;
global guidance_sq_count_reaching_back;
guidance_sq_count_reaching_back=0;
guidance_pos_reach_out_left=[8+dist_UL:.15:56+dist_UL] ;%[23:.05:41];
% guidance_pos_reach_out_left(1:100)=[8+dist_UL:.05:8+4.95+dist_UL]
% guidance_pos_reach_out_left(101:101+286)=[8+5+dist_UL:.15:56+dist_UL];
guidance_pos_reach_back_left=[56+dist_UL:-3:8+dist_UL] ;% [ 41:-1:23];

global distance;
global ball_position;
global guidance_left_position;
% global guidance_right_position;
global guidance_pos_reach_out_left_pull;
global guidance_pos_reach_back_left_pull;

global trial_preset;
% global trial_preset_rest;
trial_preset = 21; %pilot: use 3; number of trials per task
% trial_preset_rest = trial_preset-2;
global rep_period;
rep_period = 7; % seconds per repetition (ext or flex)

```

```

rep_flag = 1;
global rest_count_flag;
rest_count_flag = 0;
global recording_timer_period;
recording_timer_period = .01; % sampling rate of 100 Hz
global inst_period; % must be a multiply of rep_period
inst_period = 5; % seconds, or other multiply of 2.5
trial_preset = trial_preset*2+1; %+inst_period/2.5+1; % trials per task * out/back
% +inst_period to include time of instruction display in each task
% +1 to be matched with TasksExecuted
% trial_preset_rest = trial_preset_rest*2+inst_period/2.5+1;
global flag_inst;
global delay; % delay in .01s from target changed to the go of guidance
delay=40;

% create a random series of tasks
% way 1 using randperm: not successful
% way 2 using a lookup table (SELECTED), each line for each condition
% each run: operator types in a subject number
% and a condition number which is used as line
% number to pick up corresponding task condition
Master_two_matrix=[ 1 0;
                    2 0;
                    3 0;
                    4 0;
                    5 0;
                    6 0 ];
cond_numb=input('Type condition number (within 1 to 6)');
while isempty(cond_numb)||((cond_numb<1)||((cond_numb>6)
    cond_numb=input('Type condition number (within 1 to 6)');
end
cond_line=round(cond_numb);
Master_two=Master_two_matrix(cond_line,:);

% input the force (Newton) level
preset_newton=input('Type preset force in Newton (within 20 to 110)');
while isempty(preset_newton)||((preset_newton<20)||((preset_newton>110)
    preset_newton=input('Type preset force in Newton (within 20 to 110)');
end

```

```

random_task_array=Master_two;

% calculate the conversion rate between force level
% and the distance of movement on screen
% y-position of target_bar & start_bar: 56 & 8
flevel_max=(56-8);

% record
cond_order=strcat('C',num2str(cond_line),'_name.mat');
save(cond_order,'random_task_array');
% in readable format for everyone
char cond_order_string;
for cond_count=1:length(random_task_array)-1
    if random_task_array(cond_count)==1
        cond_order_string(cond_count,1:10)='bbSyExHi ';
    elseif random_task_array(cond_count)==4
        cond_order_string(cond_count,1:10)='bbSyFlHi ';

    elseif random_task_array(cond_count)==2
        cond_order_string(cond_count,1:10)='SyExHi ';
    elseif random_task_array(cond_count)==5
        cond_order_string(cond_count,1:10)='SyFlHi ';

    elseif random_task_array(cond_count)==3
        cond_order_string(cond_count,1:10)='ExFreeHi ';
    elseif random_task_array(cond_count)==6
        cond_order_string(cond_count,1:10)='FlFreeHi ';
    end
end
cond_order_intuit=strcat('C',num2str(cond_line),'_name_intuitive.mat');
save(cond_order_intuit,'cond_order_string');

%% Initialize the nidaq
% out = daqhwinfo
% out.InstalledAdaptors
% out = daqhwinfo('nidaq')
% out.ObjectConstructorName(:)

```

```

handles.flexiforce = analoginput('nidaq','Dev2')
out = daqhwinfo(handles.flexiforce)
% whos handles.flexiforce
% get(handles.flexiforce,{'Name','Type'})
chans = addchannel(handles.flexiforce,0:1)
% whos chans
% get(chans,{'HwChannel','Index','Parent','Type'})
% ActualRate = get(handles.flexiforce,'SampleRate')

handles.dio = digitalio('nidaq','Dev2');
addline(handles.dio,0:3,1,'out'); % 0:1,1, for port 1

start(handles.flexiforce);
start(handles.dio);
pause(1);

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes CohFgui wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = CohFgui_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
% GO box
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
handles = guidata(hObject);

global dist_UL;

global leftpush_p1;
global leftpush_p2;
global rightpush_p1;
global rightpush_p2;
global leftpull_p1;
global leftpull_p2;
global rightpull_p1;
global rightpull_p2;

global random_task_array;
global cond_line;
global preset_newton;
global newton;
global newton_right;
global flevel_max;

set(handles.pushbutton1, 'enable', 'off');
% set(handles.pushbutton6, 'BackgroundColor',[.941 .941 .941]) ;
% set(handles.pushbutton6, 'ForegroundColor',[.941 .941 .941]) ;

%%

for task_seq_index = 1:length(random_task_array)

    CondValue = random_task_array(task_seq_index);

    if CondValue == 1

        % START OF A CONDITION
        global flevel ;
        global flevel_old;
        global left_position;

```

```

left_position=[]; % to avoid rewrite previous var
flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
global left_start_position;
left_start_position=[]; % to avoid rewrite previous var
global left_target_position;
left_target_position=[]; % to avoid rewrite previous var
global flevel_right;
global flevel_right_old
% change initials here
global right_position;
right_position=[]; % to avoid rewrite previous var
flevel_right_init = 0;
global myclk;
myclk=1;
global trial_preset;
global rep_period;
global recording_timer_period;
global inst_period;
global flag_inst;
flag_inst=0;

global myclk_old;
myclk_old=0;
global distance;
global ball_position;
global ball_roll_ratio;
global guidance_left_position;
%      global guidance_right_position;

% set the target bars for the condition
L_target_pos = get(handles.edit3,'Position');
set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
L_target_pos(4)]);

%      R_target_pos = get(handles.edit7,'Position');
%      set(handles.edit7,'Position',[R_target_pos(1) R_target_pos(2) R_target_pos(3)
R_target_pos(4)]); % 36=2*(41-23)

%--

```

```

% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr , 'TimerFcn',...
    ['if rep_flag == 1;',...
    'rep_flag = 0;',...
    'else',...
    'rep_flag = 1;',...
    'end;']);

%--
% Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;
handles.LR_tmr = timer('TimerFcn',{@LRTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
guidata(handles.guifig,handles);

%--
% Third Timer used for instruction
handles.guifig = gcf;
handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',inst_period);
guidata(handles.guifig,handles);
start(handles.inst_tmr);
guidata(hObject, handles);
%--

%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;

```

```

else
    % flevel = (newton-flevel_init)/flevel_max; % adjustment based on the scale of GUI
bar movement
    flevel=(flevel_max/(preset_newton-6))*(newton-6);
end

newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    %flevel_right = (newton_right - flevel_right_init)/flevel_max; % adjustment based on
the scale of GUI bar movement
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6);
end

%%% Set homing depending on actual force at the beginning
% Left
left_pos = get(handles.text6,'Position');
% start-bar position is being used as homing
start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.001);

% homing the guidance cursors too

```



```

left_pos_cursor = get(handles.text22,'Position');
set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

```

```

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');
set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
'DrawMode','fast','Visible','off');
% 'Layer','top'); % 109 23+1 axes_pos(3) 20.23
% ball roll on the bar
% parameters: mass; gravity; alpha angle of inclined bar with
% horizontal surface; distance moved on inclined bar
% mass=1; % kg, not included in motion equation now
gravity=9.8; % m/s2
distance=0; % initials, in m (=points), measured compared to center of the bar
% negative - left side of center, positive - right side of center
distance_old=0;
sinalpha=0; % initials, ranged from -1 to 1
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

```

```

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;
while 1
    % Display instruction here
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {
                '
                ' EXTEND LEFT/RIGHT FORWARD WITH BALL BALANCE',...
                '
                HIGH LEVEL
            },...

```

```

        'When you hear a beep',...
        'MOVE the cursor bar within the two target lines by EXTENDING your
forearms',...
        'Keep ball remaining on the bar'});
set(handles.text12,'FontSize',14);
set(handles.text12,'HorizontalAlignment', 'center');
set(handles.text12, 'BackgroundColor',[.314 .318 .314]) ;
set(handles.text12, 'ForegroundColor',[ 1 1 1]) ;
set(handles.text12,'Position',...
    [9.8 4.538 64 42.385]);
% Update handles structure
guidata(hObject, handles);
else
if flag_inst==0
    % remove instruction
    set(handles.text12,'String',' ');
    set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
    set(handles.text12,'Position',...
        [9.8 14.385 32.6 24.846]);
    % start two timers
    start(handles.rep_tmr);
    start(handles.LR_tmr);
    guidata(hObject, handles);
    %--
    flag_inst=1;
    % start of triggering the Optotrak
    putvalue(handles.dio,1);
    %pause(.001)
    putvalue(handles.dio,0);
    % end of triggering
end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6

```

```

        flevel=0;
    else
        % flevel = (newton-flevel_init)/flevel_max; % adjustment based on the scale of GUI
bar movement
        flevel=(flevel_max/(preset_newton-6))*(newton-6);
    end

    newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)
    if newton_right<6
        flevel_right=0;
    else
        % flevel_right = (newton_right - flevel_right_init)/flevel_max; % adjustment based
on the scale of GUI bar movement
        flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6);
    end

    %--
    % if flevel is greater then flevel_old, make the bar closer to
    % target-bar, otherwise, lower the bar, close to start-bar
    % zero flevel produce a still bar
    % to make sure that at beginning, the movement starts from start-bar
    if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0
        handles.flevel = 0;
        handles.flevel_right = 0;
    else
        handles.flevel = flevel-flevel_old;
        handles.flevel_right = flevel_right-flevel_right_old;
    end
    % hand position
    left_pos = get(handles.text6,'Position');
    left_pos_new = left_pos(2) + handles.flevel;
    % update new hand position
    set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
    % Update handles structure
    % hand position
    right_pos = get(handles.text15,'Position');
    right_pos_new = right_pos(2) + handles.flevel_right;
    % update new hand position
    set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);

```

```

% Update handles structure
guidata(hObject, handles);

pause(.0001) % .025

%--
% based on odd/even rep count to appear/disappear the start-/target-bars
if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN
    % Left
    % start-bar appears
    set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit2,'ForegroundColor',[0 0 0]) ;
    % target-bar disappears
    set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
else % ODD
    % Left
    % start-bar disappears
    set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
    % target-bar appears
    set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit3,'ForegroundColor',[0 0 0]) ;
end
% Update handles structure
guidata(hObject, handles);
%--

%--
% set color change
% if sensor (in this case, text6) reach start (in this case, edit2)
%   sensor changes its color to green
% else if sensor (text6) reach target (edit3)
%   sensor changes its color to red
% otherwise
%   sensor remains its gray color
% end
sensor_pos = get(handles.text6,'Position'); % sensor-bar
depart_pos = get(handles.edit2,'Position'); % start-bar

```

```

target_pos = get(handles.edit3,'Position'); % target-bar
%-----
if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
    set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
%-----
elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL)% sensor reaches target
    set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
%-----
else
    set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
end

sensor_right_pos = get(handles.text15,'Position');
if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
    set(handles.text15,'BackgroundColor','g'); % sensor changes color to green
elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor','r'); % sensor changes color to red
else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end
guidata(hObject, handles);
%--

% ball roll on hill
sinalpha=(flevel-flevel_right)/sqrt((flevel-flevel_right)^2+(bar_b-bar_a)^2); % no unit
distance=ball_roll_ratio*gravity/2*sinalpha*(myclk-myclk_old)+distance_old;
% despite where it was, bring the ball back to middle on bar when returning to start-bar
if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % -1.2 & +2.3
    distance=0;
end
% (note: tried both set(h,'Rotation',deg) and rotate. neither of them work
h_bar1=plot([bar_a bar_b],[flevel flevel_right],'LineWidth',8,'Color',[.314 .318 .314]);
hold on;

```

```

        plot(bar_c+distance,ball_r/16+((flevel_right-flevel)/(bar_b-bar_a))*(bar_c+distance)
+ flevel-((flevel_right-flevel)/(bar_b-bar_a))*bar_a,'o',...
        'MarkerSize',ball_r,'MarkerFaceColor','c');
    hold off;
    axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
    % gca here stands for the plot
    set(gca, 'XTick',[], 'YTick',[], 'Color',[.925 .914 .847], 'Visible','off');
    % set(h_bar1,'Erasemode','xor');
    myclk_old = myclk;
    distance_old=distance;

    %--
    flevel_old = flevel; % store the old of flevel
    flevel_right_old = flevel_right; % store the old of flevel
    %--

    iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
    if get(handles.rep_tmr,'TasksExecuted') == trial_preset
        % old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
        % Destroy 1st timer
        stop(handles.rep_tmr); % stop timer before exit each task
        delete(handles.rep_tmr); % delete timer to start fresh in a new task

        % Destroy 2nd timer
        stop(handles.LR_tmr);
        delete(handles.LR_tmr);

        % Destroy 3rd timer
        stop(handles.inst_tmr);
        delete(handles.inst_tmr);

        % save position data
        left_position_file=strcat('C',num2str(cond_line),'_left.mat');
        save(left_position_file,'left_position');
        left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
        save(left_start_position_file,'left_start_position');
        left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
        save(left_target_position_file,'left_target_position');

```

```

% save position data
right_position_file=strcat('C',num2str(cond_line),'_right.mat');
save(right_position_file,'right_position');

% save ball position to detect a successful trial
ball_position_file=strcat('C',num2str(cond_line),'_ball.mat');
save(ball_position_file,'ball_position');
% save guidance cursor position
left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
save(left_guidance_position_file,'guidance_left_position');

% exit the infinite While
break;
% exit Left/Right loop
break;
end
%--
end % of infinite While, meaning, END OF A CONDITION

```

```

elseif CondValue == 4
% START OF A CONDITION
global flevel ;
global flevel_old;
global left_position;
left_position=[]; % to avoid rewrite previous var
flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
global left_start_position;
left_start_position=[]; % to avoid rewrite previous var
global left_target_position;
left_target_position=[]; % to avoid rewrite previous var
global flevel_right;
global flevel_right_old
% change initials here
global right_position;
right_position=[]; % to avoid rewrite previous var
flevel_right_init = 0;

```

```

global myclk;
myclk=1;
global trial_preset;
global rep_period;
global recording_timer_period;
global inst_period;
global flag_inst;
flag_inst=0;

global myclk_old;
myclk_old=0;
global distance;
global ball_position;
global ball_roll_ratio;
global guidance_left_position;

% set the target bars for the condition
L_target_pos = get(handles.edit3,'Position');
set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
L_target_pos(4)]);

%--
% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr, 'TimerFcn',...
    ['if rep_flag == 1;...',
    'rep_flag = 0;...',
    'else;...',
    'rep_flag = 1;...',
    'end;']);

%--
% Two Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;

```



```

handles.LR_tmr = timer('TimerFcn',{@LRTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
guidata(handles.guifig,handles);

%--
% Third Timer used for instruction
handles.guifig = gcf;
handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',inst_period);
guidata(handles.guifig,handles);
start(handles.inst_tmr);
guidata(hObject, handles);
%--

%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the scale
of GUI bar movement
end

newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
end

%% Set homing depending on actual force at the beginning
% Left
left_pos = get(handles.text6,'Position');
% start-bar position is being used as homing

```

```

start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.001);

% homing the guidance cursors too
left_pos_cursor = get(handles.text22,'Position');
set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');
set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
'DrawMode','fast','Visible','off');
% 'Layer','top'); % 109 23+1 axes_pos(3) 20.23
% ball roll on the bar
% parameters: mass; gravity; alpha angle of inclined bar with
% horizontal surface; distance moved on inclined bar
% mass=1; % kg, not included in motion equation now
gravity=9.8; % m/s2
distance=0; % initials, in m (=points), measured compared to center of the bar
% negative - left side of center, positive - right side of center
distance_old=0;

```

```

sinalpha=0; % initials, ranged from -1 to 1
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;
while 1
    % Display instruction here
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {'', 'FLEX LEFT/RIGHT BACKWARD WITH BALL BALANCE',...
            'HIGH LEVEL', 'When you hear a beep',...
            'MOVE the cursor bar within the two target lines by FLEXING your forearms',...
            'Keep ball remaining on the bar'});
        set(handles.text12,'FontSize',14);
        set(handles.text12,'HorizontalAlignment', 'center');
        set(handles.text12, 'BackgroundColor',[.314 .318 .314]) ;
        set(handles.text12, 'ForegroundColor',[ 1 1 1]) ;
        set(handles.text12,'Position',...
            [9.8 4.538 64 42.385]);
        % Update handles structure
        guidata(hObject, handles);
    else
        if flag_inst==0
            % remove instruction
            set(handles.text12,'String',' ');
            set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;
            set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
            set(handles.text12,'Position',...
                [9.8 14.385 32.6 24.846]);

```

```

        % start two timers
        start(handles.rep_tmr);
        start(handles.LR_tmr);
        guidata(hObject, handles);
        %--
        flag_inst=1;
        % start of triggering the Optotrak
        putvalue(handles.dio,1);
        %pause(.001)
        putvalue(handles.dio,0);
        % end of triggering
    end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the
scale of GUI bar movement
end

newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment
based on the scale of GUI bar movement
end

%--
% if flevel is greater then flevel_old, make the bar closer to
% target-bar, otherwise, lower the bar, close to start-bar
% zero flevel produce a still bar
% to make sure that at beginning, the movement starts from start-bar
if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0
    handles.flevel = 0;

```

```

        handles.flevel_right = 0;
    else
        handles.flevel = flevel-flevel_old;
        handles.flevel_right = flevel_right-flevel_right_old;
    end
    % hand position
    left_pos = get(handles.text6,'Position');
    left_pos_new = left_pos(2) + handles.flevel;
    % update new hand position
    set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
    % Update handles structure

    % hand position
    right_pos = get(handles.text15,'Position');
    right_pos_new = right_pos(2) + handles.flevel_right;
    % update new hand position
    set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
    % Update handles structure
    guidata(hObject, handles);

    pause(.0001) % .025

    %--
    % based on odd/even rep count to appear/disappear the start-/target-bars
    if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN
        % Left
        % start-bar appears
        set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.edit2,'ForegroundColor',[0 0 0]) ;
        % target-bar disappears
        set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
    else % ODD
        % Left
        % start-bar disappears
        set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
        % target-bar appears

```

```

        set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.edit3,'ForegroundColor',[0 0 0]) ;
    end
    % Update handles structure
    guidata(hObject, handles);
    %--

    %--
    % set color change
    % if sensor (in this case, text6) reach start (in this case, edit2)
    %   sensor changes its color to green
    % else if sensor (text6) reach target (edit3)
    %   sensor changes its color to red
    % otherwise
    %   sensor remains its gray color
    % end
    sensor_pos = get(handles.text6,'Position'); % sensor-bar
    depart_pos = get(handles.edit2,'Position'); % start-bar
    target_pos = get(handles.edit3,'Position'); % target-bar
    %-----
    if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
        set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
    %-----
    elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL)% sensor reaches target
        set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
    %-----
    else
        set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
    end

    sensor_right_pos = get(handles.text15,'Position');
    if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
        set(handles.text15,'BackgroundColor','g'); % sensor changes color to green
    elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
        set(handles.text15, 'BackgroundColor','r'); % sensor changes color to red
    end

```

```

else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end

guidata(hObject, handles);
%--

% ball roll on hill
sinalpha=(flevel-flevel_right)/sqrt((flevel-flevel_right)^2+(bar_b-bar_a)^2); % no unit
distance=ball_roll_ratio*gravity/2*sinalpha*(myclk-myclk_old)+distance_old;
% despite where it was, bring the ball back to middle on bar when returning to start-bar
if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % -1.2 & +2.3
    distance=0;
end
% note: tried both set(h,'Rotation',deg) and rotate.
% neither of them work
h_bar1=plot([bar_a bar_b],[flevel flevel_right],'LineWidth',8,'Color',[.314 .318 .314]);
hold on;
plot(bar_c+distance,ball_r/16+((flevel_right-flevel)/(bar_b-bar_a))*(bar_c+distance)
+flevel-((flevel_right-flevel)/(bar_b-bar_a))*bar_a,'o',...
'MarkerSize',ball_r,'MarkerFaceColor','c');
hold off;
axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
% gca here stands for the plot
set(gca, 'XTick',[], 'YTick',[], 'Color',[.925 .914 .847], 'Visible','off');

% set(h_bar1,'Erasemode','xor');
myclk_old = myclk;
distance_old=distance;

%--
flevel_old = flevel; % store the old of flevel
flevel_right_old = flevel_right; % store the old of flevel
%--

iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
if get(handles.rep_tmr,'TasksExecuted') == trial_preset

```

```

% old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
% Destroy 1st timer
stop(handles.rep_tmr); % stop timer before exit each task
delete(handles.rep_tmr); % delete timer to start fresh in a new task

% Destroy 2nd timer
stop(handles.LR_tmr);
delete(handles.LR_tmr);

% Destroy 3rd timer
stop(handles.inst_tmr);
delete(handles.inst_tmr);

% save position data
left_position_file=strcat('C',num2str(cond_line),'_left.mat');
save(left_position_file,'left_position');
left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
save(left_start_position_file,'left_start_position');
left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
save(left_target_position_file,'left_target_position');

% save position data
right_position_file=strcat('C',num2str(cond_line),'_right.mat');
save(right_position_file,'right_position');

% save ball position to detect a successful trial
ball_position_file=strcat('C',num2str(cond_line),'_ball.mat');
save(ball_position_file,'ball_position');
% save guidance cursor position
left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
save(left_guidance_position_file,'guidance_left_position');

% exit the infinite While
break;
% exit Left/Right loop
break;
end
%--
end % of infinite While, meaning, END OF A CONDITION

```



```

elseif CondValue == 2
    % START OF A CONDITION
    global flevel ;
    global flevel_old;
    global left_position;
    left_position=[]; % to avoid rewrite previous var
    flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
    global left_start_position;
    left_start_position=[]; % to avoid rewrite previous var
    global left_target_position;
    left_target_position=[]; % to avoid rewrite previous var
    global flevel_right;
    global flevel_right_old
    % change initials here
    global right_position;
    right_position=[]; % to avoid rewrite previous var
    flevel_right_init = 0;
    global myclk;
    myclk=1;
    global trial_preset;
    global rep_period;
    global recording_timer_period;
    global inst_period;
    global flag_inst;
    flag_inst=0;

    global myclk_old;
    myclk_old=0;
    global guidance_left_position;

    % set the target bars for the condition
    L_target_pos = get(handles.edit3,'Position');

```

```

        set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
L_target_pos(4)]);

%--
% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr , 'TimerFcn',...
    ['if rep_flag == 1;',...
    'rep_flag = 0;',...
    'else',...
    'rep_flag = 1;',...
    'end;']);

%--
% Two Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;
handles.LR_tmr = timer('TimerFcn',
{@NoBB_LRTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
guidata(handles.guifig,handles);

%--
% Third Timer used for instruction
handles.guifig = gcf;
handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',inst_period);
guidata(handles.guifig,handles);
start(handles.inst_tmr);
guidata(hObject, handles);
%--

```

```

%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the scale
of GUI bar movement
end

newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
end

%% Set homing depending on actual force at the beginning
% Left
left_pos = get(handles.text6,'Position');
% start-bar position is being used as homing
start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

```

```

pause(.001);

% homing the guidance cursors too
left_pos_cursor = get(handles.text22,'Position');
set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');
set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
'DrawMode','fast','Visible','off');
% 'Layer','top'); % 109 23+1 axes_pos(3) 20.23
% ball roll on the bar
% parameters: mass; gravity; alpha angle of inclined bar with
% horizontal surface; distance moved on inclined bar
% mass=1; % kg, not included in motion equation now
gravity=9.8; % m/s2
% distance=0; % initials, in m (=points), measured compared to center of the bar
% % negative - left side of center, positive - right side of center
% distance_old=0;
sinalpha=0; % initials, ranged from -1 to 1
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;
while 1
    % Display instruction here
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {
                '

```

```

        ' EXTEND LEFT/RIGHT FORWARD ',...
        '          HIGH LEVEL          ',...
        '          ',...
        'When you hear a beep',...
        'MOVE the cursor bar within the two target lines by EXTENDING your
forearms'});
    set(handles.text12,'FontSize',14);
    set(handles.text12,'HorizontalAlignment', 'center');
    set(handles.text12, 'BackgroundColor',[.314 .318 .314]) ;
    set(handles.text12, 'ForegroundColor',[ 1 1 1]) ;
    set(handles.text12,'Position',...
        [9.8 4.538 64 42.385]);
    % Update handles structure
    guidata(hObject, handles);
else
    if flag_inst==0
        % remove instruction
        set(handles.text12,'String',' ');
        set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
        set(handles.text12,'Position',...
            [9.8 14.385 32.6 24.846]);
        % start two timers
        start(handles.rep_tmr);
        start(handles.LR_tmr);
        guidata(hObject, handles);
        %--
        flag_inst=1;
        % start of triggering the Optotrak
        putvalue(handles.dio,1);
        %pause(.001)
        putvalue(handles.dio,0);
        % end of triggering
    end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)

```

```

% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the
scale of GUI bar movement
end

newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
end

%--
% if flevel is greater then flevel_old, make the bar closer to
% target-bar, otherwise, lower the bar, close to start-bar
% zero flevel produce a still bar
% to make sure that at beginning, the movement starts from start-bar
if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0
    handles.flevel = 0;
    handles.flevel_right = 0;
else
    handles.flevel = flevel-flevel_old;
    handles.flevel_right = flevel_right-flevel_right_old;
end
% hand position
left_pos = get(handles.text6,'Position');
left_pos_new = left_pos(2) + handles.flevel;
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% Update handles structure

%         guidata(hObject, handles);

% hand position
right_pos = get(handles.text15,'Position');

```

```

right_pos_new = right_pos(2) + handles.flevel_right;
% update new hand position
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.0001) %.025

%--
% based on odd/even rep count to appear/disappear the start-/target-bars
if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN
    % Left
    % start-bar appears
    set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit2,'ForegroundColor',[0 0 0]) ;
    % target-bar disappears
    set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
else % ODD
    % Left
    % start-bar disappears
    set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
    % target-bar appears
    set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit3,'ForegroundColor',[0 0 0]) ;
end
% Update handles structure
guidata(hObject, handles);
%--
% set color change
% if sensor (in this case, text6) reach start (in this case, edit2)
%   sensor changes its color to green
% else if sensor (text6) reach target (edit3)
%   sensor changes its color to red
% otherwise
%   sensor remains its gray color
% end

```

```

sensor_pos = get(handles.text6,'Position'); % sensor-bar
depart_pos = get(handles.edit2,'Position'); % start-bar
target_pos = get(handles.edit3,'Position'); % target-bar
%-----
if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
    set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
%-----
elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL)% sensor reaches target
    set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
%-----
else
    set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
end

sensor_right_pos = get(handles.text15,'Position');
if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
    set(handles.text15,'BackgroundColor','g'); % sensor changes color to green
elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor','r'); % sensor changes color to red
else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end

guidata(hObject, handles);
%--

h_bar1=plot([bar_a bar_b],[flevel flevel_right],'LineWidth',8,'Color',[.314 .318 .314]);
axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
% gca here stands for the plot
set(gca, 'XTick',[], 'YTick',[], 'Color',[.925 .914 .847], 'Visible','off');
% set(h_bar1,'Erasemode','xor');
myclk_old = myclk;

%--

```



```

flevel_old = flevel; % store the old of flevel
flevel_right_old = flevel_right; % store the old of flevel
%--

iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
if get(handles.rep_tmr,'TasksExecuted') == trial_preset
    % old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
    % Destroy 1st timer
    stop(handles.rep_tmr); % stop timer before exit each task
    delete(handles.rep_tmr); % delete timer to start fresh in a new task

    % Destroy 2nd timer
    stop(handles.LR_tmr);
    delete(handles.LR_tmr);

    % Destroy 3rd timer
    stop(handles.inst_tmr);
    delete(handles.inst_tmr);

    % save position data
    left_position_file=strcat('C',num2str(cond_line),'_left.mat');
    save(left_position_file,'left_position');
    left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
    save(left_start_position_file,'left_start_position');
    left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
    save(left_target_position_file,'left_target_position');

    % save position data
    right_position_file=strcat('C',num2str(cond_line),'_right.mat');
    save(right_position_file,'right_position');

    left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
    save(left_guidance_position_file,'guidance_left_position');
    % exit the infinite While
    break;
    % exit Left/Right loop
    break;
end
%--

```

end % of infinite While, meaning, END OF A CONDITION

```
elseif CondValue == 5
    % START OF A CONDITION
    global flevel ;
    global flevel_old;
    global left_position;
    left_position=[]; % to avoid rewrite previous var
    flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
    global left_start_position;
    left_start_position=[]; % to avoid rewrite previous var
    global left_target_position;
    left_target_position=[]; % to avoid rewrite previous var
    global flevel_right;
    global flevel_right_old
    % change initials here
    global right_position;
    right_position=[]; % to avoid rewrite previous var
    flevel_right_init = 0;
    global myclk;
    myclk=1;
    global trial_preset;
    global rep_period;
    global recording_timer_period;
    global inst_period;
    global flag_inst;
    flag_inst=0;

    global myclk_old;
    myclk_old=0;
    global guidance_left_position;

    % set the target bars for the condition
    L_target_pos = get(handles.edit3,'Position');
    set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
    L_target_pos(4)]);
```

```

%--
% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr , 'TimerFcn',...
    ['if rep_flag == 1;',...
    'rep_flag = 0;',...
    'else',...
    'rep_flag = 1;',...
    'end;']);

%--
% Two Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;
handles.LR_tmr = timer('TimerFcn',
{@NoBB_LRTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
guidata(handles.guifig,handles);

%--
% Third Timer used for instruction
handles.guifig = gcf;
handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',inst_period);
guidata(handles.guifig,handles);
start(handles.inst_tmr);
guidata(hObject, handles);
%--

%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
% this to avoid a little press on sensor to be counted

```

```

if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the scale
of GUI bar movement
end

newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
end

%% Set homing depending on actual force at the beginning
% Left
left_pos = get(handles.text6,'Position');
% start-bar position is being used as homing
start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.001);

% homing the guidance cursors too
left_pos_cursor = get(handles.text22,'Position');

```

```

        set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
        set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

```

```

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');
set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
'DrawMode','fast','Visible','off');

```

```

gravity=9.8; % m/s2
sinalpha=0; % initials, ranged from -1 to 1
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

```

```

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;
while 1
    % Display instruction here
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {
                'FLEX LEFT/RIGHT BACKWARD ',...
                'HIGH LEVEL ',...
                'When you hear a beep,',...
                'MOVE the cursor bar within the two target lines by FLEXING your forearms'});
        set(handles.text12,'FontSize',14);
        set(handles.text12,'HorizontalAlignment','center');
        set(handles.text12,'BackgroundColor',[.314 .318 .314]);
        set(handles.text12,'ForegroundColor',[ 1 1 1]);
        set(handles.text12,'Position',...

```

```

        [9.8 4.538 64 42.385]);
    % Update handles structure
    guidata(hObject, handles);
else
    if flag_inst==0
        % remove instruction
        set(handles.text12,'String',' ');
        set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
        set(handles.text12,'Position',...
            [9.8 14.385 32.6 24.846]);
        % start two timers
        start(handles.rep_tmr);
        start(handles.LR_tmr);
        guidata(hObject, handles);
        %--
        flag_inst=1;
        % start of triggering the Optotrak
        putvalue(handles.dio,1);
        %pause(.001)
        putvalue(handles.dio,0);
        % end of triggering
    end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the
scale of GUI bar movement
end

newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else

```

```

        flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment
based on the scale of GUI bar movement

```

```

end

```

```

%--

```

```

% if flevel is greater then flevel_old, make the bar closer to

```

```

% target-bar, otherwise, lower the bar, close to start-bar

```

```

% zero flevel produce a still bar

```

```

% to make sure that at beginning, the movement starts from start-bar

```

```

if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0

```

```

    handles.flevel = 0;

```

```

    handles.flevel_right = 0;

```

```

else

```

```

    handles.flevel = flevel-flevel_old;

```

```

    handles.flevel_right = flevel_right-flevel_right_old;

```

```

end

```

```

% hand position

```

```

left_pos = get(handles.text6,'Position');

```

```

left_pos_new = left_pos(2) + handles.flevel;

```

```

% update new hand position

```

```

set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);

```

```

% Update handles structure

```

```

% hand position

```

```

right_pos = get(handles.text15,'Position');

```

```

right_pos_new = right_pos(2) + handles.flevel_right;

```

```

% update new hand position

```

```

set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);

```

```

% Update handles structure

```

```

guidata(hObject, handles);

```

```

pause(.0001) %.025

```

```

%--

```

```

% based on odd/even rep count to appear/disappear the start-/target-bars

```

```

if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN

```

```

    % Left

```

```

    % start-bar appears

```

```

        set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.edit2,'ForegroundColor',[0 0 0]) ;
        % target-bar disappears
        set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
    else % ODD
        % Left
        % start-bar disappears
        set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
        % target-bar appears
        set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.edit3,'ForegroundColor',[0 0 0]) ;
    end
    % Update handles structure
    guidata(hObject, handles);
    %--

    %--
    % set color change
    % if sensor (in this case, text6) reach start (in this case, edit2)
    %   sensor changes its color to green
    % else if sensor (text6) reach target (edit3)
    %   sensor changes its color to red
    % otherwise
    %   sensor remains its gray color
    % end
    sensor_pos = get(handles.text6,'Position'); % sensor-bar
    depart_pos = get(handles.edit2,'Position'); % start-bar
    target_pos = get(handles.edit3,'Position'); % target-bar
    %-----
    if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
        set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
    %-----
    elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL) % sensor reaches target
        set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
    %-----

```



```

else
    set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
end

sensor_right_pos = get(handles.text15, 'Position');
if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor', 'g'); % sensor changes color to green
elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor', 'r'); % sensor changes color to red
else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end

guidata(hObject, handles);
%--

% note: tried both set(h, 'Rotation', deg) and rotate. neither of them work
h_bar1=plot([bar_a bar_b],[flevel flevel_right], 'LineWidth', 8, 'Color', [.314 .318 .314]);
axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
% gca here stands for the plot
set(gca, 'XTick', [], 'YTick', [], 'Color', [.925 .914 .847], 'Visible', 'off');
% set(h_bar1, 'Erasemode', 'xor');
myclk_old = myclk;

%--
flevel_old = flevel; % store the old of flevel
flevel_right_old = flevel_right; % store the old of flevel
%--

iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
if get(handles.rep_tmr, 'TasksExecuted') == trial_preset
    % old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
    % Destroy 1st timer
    stop(handles.rep_tmr); % stop timer before exit each task
    delete(handles.rep_tmr); % delete timer to start fresh in a new task

```

```

    % Destroy 2nd timer
    stop(handles.LR_tmr);
    delete(handles.LR_tmr);

    % Destroy 3rd timer
    stop(handles.inst_tmr);
    delete(handles.inst_tmr);

    % save position data
    left_position_file=strcat('C',num2str(cond_line),'_left.mat');
    save(left_position_file,'left_position');
    left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
    save(left_start_position_file,'left_start_position');
    left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
    save(left_target_position_file,'left_target_position');

    % save position data
    right_position_file=strcat('C',num2str(cond_line),'_right.mat');
    save(right_position_file,'right_position');
    % save guidance cursor position
    left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
    save(left_guidance_position_file,'guidance_left_position');

    % exit the infinite While
    break;
    % exit Left/Right loop
    break;
end
%--
end % of infinite While, meaning, END OF A CONDITION

elseif CondValue == 3
    % START OF A CONDITION
    global flevel ;
    global flevel_old;
    global left_position;
    left_position=[]; % to avoid rewrite previous var
    flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
    global left_start_position;

```

```

left_start_position=[]; % to avoid rewrite previous var
global left_target_position;
left_target_position=[]; % to avoid rewrite previous var
global flevel_right;
global flevel_right_old
% change initials here
global right_position;
right_position=[]; % to avoid rewrite previous var
flevel_right_init = 0;
global myclk;
myclk=1;
global trial_preset;
global rep_period;
global recording_timer_period;
global inst_period;
global flag_inst;
flag_inst=0;

global myclk_old;
myclk_old=0;
global guidance_left_position;

% set the target bars for the condition
L_target_pos = get(handles.edit3,'Position');
set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
L_target_pos(4)]);

%      R_target_pos = get(handles.edit7,'Position');
%      set(handles.edit7,'Position',[R_target_pos(1) R_target_pos(2) R_target_pos(3)
R_target_pos(4)]); % 36=2*(41-23)

%--
% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr , 'TimerFcn',...

```

```

        ['if rep_flag == 1;',...
        'rep_flag = 0;',...
        'else',...
        'rep_flag = 1;',...
        'end;']);

%--
% Two Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;
handles.LR_tmr = timer('TimerFcn',
{@NoBB_LRTmrFcn,handles.guifig},'BusyMode','Queue',...
'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
guidata(handles.guifig,handles);

%--
% Third Timer used for instruction
handles.guifig = gcf;
handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
'ExecutionMode','FixedRate','Period',inst_period);
guidata(handles.guifig,handles);
start(handles.inst_tmr);
guidata(hObject, handles);
%--

%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the scale
of GUI bar movement
end

newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)

```

```

if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
end

%%% Set homing depending on actual force at the beginning
% Left
left_pos = get(handles.text6,'Position');
% start-bar position is being used as homing
start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.001);

% homing the guidance cursors too
left_pos_cursor = get(handles.text22,'Position');
set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');

```

```

set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
    'DrawMode','fast','Visible','off');
sinalpha=0; % initials, ranged from -1 to 1
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;
while 1
    % Display instruction here
    % display instruction
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {
                '
                ' EXTEND ' ,...
                '
                ' HIGH LEVEL ' ,...
            },...
            'When you hear a beep',...
            'MOVE the cursor bar within the two target lines by EXTENDING your
forearms'});
        set(handles.text12,'FontSize',14);
        set(handles.text12,'HorizontalAlignment', 'center');
        set(handles.text12, 'BackgroundColor',[.314 .318 .314]) ;
        set(handles.text12, 'ForegroundColor',[ 1 1 1]) ;
        set(handles.text12,'Position',...
            [9.8 4.538 64 42.385]);
        % Update handles structure
        guidata(hObject, handles);
    else
        if flag_inst==0
            % remove instruction
            set(handles.text12,'String',' ');
            set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;

```

```

        set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
        set(handles.text12,'Position',...
            [9.8 14.385 32.6 24.846]);
        % start two timers
        start(handles.rep_tmr);
        start(handles.LR_tmr);
        guidata(hObject, handles);
        %--
        flag_inst=1;
        % start of triggering the Optotrak
        putvalue(handles.dio,1);
        %pause(.001)
        putvalue(handles.dio,0);
        % end of triggering
    end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpush_p1*volt(1)+leftpush_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the
scale of GUI bar movement
end

newton_right=rightpush_p1*volt(2)+rightpush_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment
based on the scale of GUI bar movement
end

%--
% if flevel is greater then flevel_old, make the bar closer to
% target-bar, otherwise, lower the bar, close to start-bar
% zero flevel produce a still bar

```

```

% to make sure that at beginning, the movement starts from start-bar
if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0
    handles.flevel = 0;
    handles.flevel_right = 0;
else
    handles.flevel = flevel-flevel_old;
    handles.flevel_right = flevel_right-flevel_right_old;
end
% hand position
left_pos = get(handles.text6,'Position');
left_pos_new = left_pos(2) + handles.flevel;
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% Update handles structure

% hand position
right_pos = get(handles.text15,'Position');
right_pos_new = right_pos(2) + handles.flevel_right;
% update new hand position
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.0001) %.025

%--
% based on odd/even rep count to appear/disappear the start-/target-bars
if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN
    % Left
    % start-bar appears
    set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit2,'ForegroundColor',[0 0 0]) ;
    % target-bar disappears
    set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
else % ODD
    % Left
    % start-bar disappears

```



```

        set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
        set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
        % target-bar appears
        set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.edit3,'ForegroundColor',[0 0 0]) ;
    end
    % Update handles structure
    guidata(hObject, handles);
    %--

    %--
    % set color change
    % if sensor (in this case, text6) reach start (in this case, edit2)
    %   sensor changes its color to green
    % else if sensor (text6) reach target (edit3)
    %   sensor changes its color to red
    % otherwise
    %   sensor remains its gray color
    % end
    sensor_pos = get(handles.text6,'Position'); % sensor-bar
    depart_pos = get(handles.edit2,'Position'); % start-bar
    target_pos = get(handles.edit3,'Position'); % target-bar
    %-----
    if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
        set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
    %-----
    elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL)% sensor reaches target
        set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
    %-----
    else
        set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
    end

    sensor_right_pos = get(handles.text15,'Position');
    if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
        set(handles.text15,'BackgroundColor','g'); % sensor changes color to green

```

```

elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor','r'); % sensor changes color to red
else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end

guidata(hObject, handles);
%--

% note: tried both set(h,'Rotation',deg) and rotate. neither of them work
h_bar1=plot([bar_a bar_b],[(flevel+flevel_right)/2 (flevel+flevel_right)/2],'LineWidth',
8,'Color',[.314 .318 .314]);
axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
% gca here stands for the plot
set(gca, 'XTick',[], 'YTick',[], 'Color',[.925 .914 .847], 'Visible','off');
% set(h_bar1,'Erasemode','xor');
myclk_old = myclk;

%--
flevel_old = flevel; % store the old of flevel
flevel_right_old = flevel_right; % store the old of flevel
%--

iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
if get(handles.rep_tmr,'TasksExecuted') == trial_preset
    % old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
    % Destroy 1st timer
    stop(handles.rep_tmr); % stop timer before exit each task
    delete(handles.rep_tmr); % delete timer to start fresh in a new task

    % Destroy 2nd timer
    stop(handles.LR_tmr);
    delete(handles.LR_tmr);

    % Destroy 3rd timer
    stop(handles.inst_tmr);
    delete(handles.inst_tmr);

```

```

    % save position data
    left_position_file=strcat('C',num2str(cond_line),'_left.mat');
    save(left_position_file,'left_position');
    left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
    save(left_start_position_file,'left_start_position');
    left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
    save(left_target_position_file,'left_target_position');

    % save position data
    right_position_file=strcat('C',num2str(cond_line),'_right.mat');
    save(right_position_file,'right_position');
    % save guidance cursor position
    left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
    save(left_guidance_position_file,'guidance_left_position');
    % exit the infinite While
    break;
    % exit Left/Right loop
    break;
end
%--
end % of infinite While, meaning, END OF A CONDITION

```

```

elseif CondValue == 6
    % START OF A CONDITION
    global flevel ;
    global flevel_old;
    global left_position;
    left_position=[]; % to avoid rewrite previous var
    flevel_init = 0; % flevel = (flevel - flevel_init)/flevel_max;
    global left_start_position;
    left_start_position=[]; % to avoid rewrite previous var
    global left_target_position;
    left_target_position=[]; % to avoid rewrite previous var
    global flevel_right;
    global flevel_right_old
    % change initials here
    global right_position;

```

```

right_position=[]; % to avoid rewrite previous var
flevel_right_init = 0;
global myclk;
myclk=1;
global trial_preset;
global rep_period;
global recording_timer_period;
global inst_period;
global flag_inst;
flag_inst=0;

global myclk_old;
myclk_old=0;
global guidance_left_position;

% set the target bars for the condition
L_target_pos = get(handles.edit3,'Position');
set(handles.edit3,'Position',[L_target_pos(1) L_target_pos(2) L_target_pos(3)
L_target_pos(4)]);

%--
% First Timer
% Fcn the timer calls when it stops
% Period indicates the delay (seconds) between execution of TimerFcn
handles.rep_tmr = timer('executionMode', 'fixedRate',...
    'TasksToExecute', trial_preset, 'Period', rep_period);
set(handles.rep_tmr,'StartFcn','rep_flag=1;');
set(handles.rep_tmr , 'TimerFcn',...
    ['if rep_flag == 1;',...
    'rep_flag = 0;',...
    'else',...
    'rep_flag = 1;',...
    'end;']);

%--
% Two Second Timers used to create time mark for position
% recording Left and Right hand
handles.guifig = gcf;

```

```

        handles.LR_tmr = timer('TimerFcn',
{@NoBB_LRTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',recording_timer_period); % timer recording both
hands
    guidata(handles.guifig,handles);

    %%--
    % Third Timer used for instruction
    handles.guifig = gcf;
    handles.inst_tmr = timer('TimerFcn',
{@InstTmrFcn,handles.guifig},'BusyMode','Queue',...
    'ExecutionMode','FixedRate','Period',inst_period);
    guidata(handles.guifig,handles);
    start(handles.inst_tmr);
    guidata(hObject, handles);
    %%--

    %% Receive data from sensor through nidaq
    volt=getsample(handles.flexiforce);
    newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
    % this to avoid a little press on sensor to be counted
    if newton<6
        flevel=0;
    else
        flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the scale
of GUI bar movement
    end

    newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
    if newton_right<6
        flevel_right=0;
    else
        flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment based
on the scale of GUI bar movement
    end

    %% Set homing depending on actual force at the beginning
    % Left
    left_pos = get(handles.text6,'Position');

```

```

% start-bar position is being used as homing
start_pos = get(handles.edit2,'Position');
left_pos_new = start_pos(2)+flevel; % +.25
% update new hand position
set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% set(handles.display,'CurrentAxes',handles.displayaxis)
% Update handles structure
guidata(hObject, handles);

% Right
right_pos = get(handles.text15,'Position');
right_pos_new = left_pos_new;
% update new
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.001);

% homing the guidance cursors too
left_pos_cursor = get(handles.text22,'Position');
set(handles.text22,'Position',[left_pos_cursor(1) left_pos_new+dist_UL
left_pos_cursor(3) left_pos_cursor(4)]);
set(handles.text24,'Position',[left_pos_cursor(1) left_pos_new-dist_UL left_pos_cursor(3)
left_pos_cursor(4)]);

% draw space for bar and ball
axes_pos=get(handles.axes8,'Position');
set(handles.axes8, 'Position',[axes_pos(1) start_pos(2) axes_pos(3) 60 ],...
'DrawMode','fast','Visible','off'); %
bar_a=0; % left loc of the bar (hill)
bar_b=axes_pos(3); % was 29, right loc, in points
bar_c=bar_a+(bar_b-bar_a)/2; % center of the bar
ball_r=20; % radius of the ball (not in points, but MarkerSize units)

%--
%% Tracking the change of flevel and move the hand bar
iter_loop = 0;

```

```

while 1
    % Display instruction here
    % display instruction
    if get(handles.inst_tmr,'TasksExecuted')<=1
        % display instruction
        set(handles.text12,'String',...
            {'
            '
            ' FLEX ' ,...
            '
            '          HIGH LEVEL          ' ,...
            '
            'When you hear a beep',...
            'MOVE the cursor bar within the two target lines by FLEXING your forearms'});
        set(handles.text12,'FontSize',14);
        set(handles.text12,'HorizontalAlignment','center');
        set(handles.text12,'BackgroundColor',[.314 .318 .314]) ;
        set(handles.text12,'ForegroundColor',[ 1 1 1]) ;
        set(handles.text12,'Position',...
            [9.8 4.538 64 42.385]);
        % Update handles structure
        guidata(hObject, handles);
    else
        if flag_inst==0
            % remove instruction
            set(handles.text12,'String',' ');
            set(handles.text12,'BackgroundColor',[.925 .914 .847]) ;
            set(handles.text12,'ForegroundColor',[.941 .941 .941]) ;
            set(handles.text12,'Position',...
                [9.8 14.385 32.6 24.846]);
            % start two timers
            start(handles.rep_tmr);
            start(handles.LR_tmr);
            guidata(hObject, handles);
            %--
            flag_inst=1;
            % start of triggering the Optotrak
            putvalue(handles.dio,1);
            %pause(.001)
        end
    end
end

```

```

        putvalue(handles.dio,0);
        % end of triggering
    end
end
%% Receive data from sensor through nidaq
volt=getsample(handles.flexiforce);
newton=leftpull_p1*volt(1)+leftpull_p2; % (Newtons)
% this to avoid a little press on sensor to be counted
if newton<6
    flevel=0;
else
    flevel=(flevel_max/(preset_newton-6))*(newton-6); % adjustment based on the
scale of GUI bar movement
end

newton_right=rightpull_p1*volt(2)+rightpull_p2; % (Newtons)
if newton_right<6
    flevel_right=0;
else
    flevel_right=(flevel_max/(preset_newton-6))*(newton_right-6); % adjustment
based on the scale of GUI bar movement
end

%--
% if flevel is greater then flevel_old, make the bar closer to
% target-bar, otherwise, lower the bar, close to start-bar
% zero flevel produce a still bar
% to make sure that at beginning, the movement starts from start-bar
if iter_loop == 0 % get(handles.rep_tmr,'TasksExecuted') == 0
    handles.flevel = 0;
    handles.flevel_right = 0;
else
    handles.flevel = flevel-flevel_old;
    handles.flevel_right = flevel_right-flevel_right_old;
end
% hand position
left_pos = get(handles.text6,'Position');
left_pos_new = left_pos(2) + handles.flevel;
% update new hand position

```



```

set(handles.text6,'Position',[left_pos(1) left_pos_new left_pos(3) left_pos(4)]);
% Update handles structure

% hand position
right_pos = get(handles.text15,'Position');
right_pos_new = right_pos(2) + handles.flevel_right;
% update new hand position
set(handles.text15,'Position',[right_pos(1) right_pos_new right_pos(3) right_pos(4)]);
% Update handles structure
guidata(hObject, handles);

pause(.0001) %.025

%--
% based on odd/even rep count to appear/disappear the start-/target-bars
if mod(get(handles.rep_tmr,'TasksExecuted'),2) == 0 % EVEN
    % Left
    % start-bar appears
    set(handles.edit2,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit2,'ForegroundColor',[0 0 0]) ;
    % target-bar disappears
    set(handles.edit3,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit3,'ForegroundColor',[.925 .914 .847]) ;
else % ODD
    % Left
    % start-bar disappears
    set(handles.edit2,'BackgroundColor',[.925 .914 .847]) ;
    set(handles.edit2,'ForegroundColor',[.925 .914 .847]) ;
    % target-bar appears
    set(handles.edit3,'BackgroundColor',[.314 .318 .314]) ;
    set(handles.edit3,'ForegroundColor',[0 0 0]) ;
end
% Update handles structure
guidata(hObject, handles);
%--

%--
% set color change

```

```

% if sensor (in this case, text6) reach start (in this case, edit2)
%   sensor changes its color to green
% else if sensor (text6) reach target (edit3)
%   sensor changes its color to red
% otherwise
%   sensor remains its gray color
% end
sensor_pos = get(handles.text6,'Position'); % sensor-bar
depart_pos = get(handles.edit2,'Position'); % start-bar
target_pos = get(handles.edit3,'Position'); % target-bar
%-----
if (sensor_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL) % sensor reaches start, based on text6 and edit2 distance
    set(handles.text6,'BackgroundColor','g'); % sensor changes color to green
%-----
elseif (sensor_pos(2) >= target_pos(2)-dist_UL) && (sensor_pos(2) <=
target_pos(2)+dist_UL)% sensor reaches target
    set(handles.text6, 'BackgroundColor','r'); % sensor changes color to red
%-----
else
    set(handles.text6, 'BackgroundColor', [.314 .318 .314]); % sensor remains gray
end

sensor_right_pos = get(handles.text15,'Position');
if (sensor_right_pos(2) >= depart_pos(2)-dist_UL) && (sensor_pos(2) <=
depart_pos(2)+dist_UL)
    set(handles.text15,'BackgroundColor','g'); % sensor changes color to green
elseif (sensor_right_pos(2) >= target_pos(2)-dist_UL) && (sensor_right_pos(2) <=
target_pos(2)+dist_UL)
    set(handles.text15, 'BackgroundColor','r'); % sensor changes color to red
else
    set(handles.text15, 'BackgroundColor', [.314 .318 .314]); % sensor remains dark-
gray
end

guidata(hObject, handles);
%--

```

```

h_bar1=plot([bar_a bar_b],[(flevel+flevel_right)/2 (flevel+flevel_right)/2],'LineWidth',
8,'Color',[.314 .318 .314]);
axis([0 axes_pos(3) 0 60+1]); % H was 20.23; L was 60+1
% gca here stands for the plot
set(gca, 'XTick',[], 'YTick',[], 'Color',[.925 .914 .847], 'Visible','off');
% set(h_bar1,'Erasemode','xor');
myclk_old = myclk;

%--
flevel_old = flevel; % store the old of flevel
flevel_right_old = flevel_right; % store the old of flevel
%--

iter_loop = iter_loop + 1; % just count how many loop ran inside the infinite while
if get(handles.rep_tmr,'TasksExecuted') == trial_preset
    % old: iter_loop == 400 % equivalent to 400*pause(.1/2) = 20s
    % Destroy 1st timer
    stop(handles.rep_tmr); % stop timer before exit each task
    delete(handles.rep_tmr); % delete timer to start fresh in a new task

    % Destroy 2nd timer
    stop(handles.LR_tmr);
    delete(handles.LR_tmr);

    % Destroy 3rd timer
    stop(handles.inst_tmr);
    delete(handles.inst_tmr);

    % save position data
    left_position_file=strcat('C',num2str(cond_line),'_left.mat');
    save(left_position_file,'left_position');
    left_start_position_file=strcat('C',num2str(cond_line),'_left_start.mat');
    save(left_start_position_file,'left_start_position');
    left_target_position_file=strcat('C',num2str(cond_line),'_left_target.mat');
    save(left_target_position_file,'left_target_position');

    % save position data
    right_position_file=strcat('C',num2str(cond_line),'_right.mat');
    save(right_position_file,'right_position');

```

```

    % save guidance cursor position
    left_guidance_position_file=strcat('C',num2str(cond_line),'_guidance_left.mat');
    save(left_guidance_position_file,'guidance_left_position');

    % exit the infinite While
    break;
    % exit Left/Right loop
    break;
end
%--
end % of infinite While, meaning, END OF A CONDITION

else
    % STOP AND EXIT
    % nidaq
    stop(handles.flexiforce)
    delete(handles.flexiforce)
    stop(handles.dio)
    delete(handles.dio)
    % clear all old timers if missed in clear above
    allprevioustimer = timerfindall;
    clear allprevioustimer
    clear global
    % exit
    close(gcf)
    % note: don't use clear all
    % END OF STOP AND EXIT

    end % END OF FLAG CONDITION (CondValue)
end % END OF ALL TASKS

% start bar
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit2 as text
%      str2double(get(hObject,'String')) returns contents of edit2 as a double

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% target bar
function edit3_Callback(hObject, eventdata, handles)

% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%      str2double(get(hObject,'String')) returns contents of edit3 as a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```
% --- Executes during object creation, after setting all properties.
% cursor box
function text6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% --- Executes during object creation, after setting all properties.
% second box
function text10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% --- Executes during object creation, after setting all properties.
% s unit box
function text11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
function LRTmrFcn(src,event,handles)
```

```
handles = guidata(handles);
```

```
global dist_UL;
```

```
global newton;
global newton_right;
```

```
global left_position;
global left_start_position;
global left_target_position;
global right_position;
% global right_start_position;
% global right_target_position;
```

```

global myclk;

global guidance_pos_reach_out_left;
global guidance_pos_reach_back_left;
% global guidance_pos_reach_out_right;
% global guidance_pos_reach_back_right;
global guidance_sq_count_reaching_out;
global guidance_sq_count_reaching_back;
global distance;
global ball_position;
global guidance_left_position;
% global guidance_right_position;

global recording_timer_period

global rep_period;

global delay;

depart_pos=get(handles.edit2,'Position');
target_pos=get(handles.edit3,'Position');

% depart_right_pos=get(handles.edit5,'Position');
% target_right_pos=get(handles.edit7,'Position');

myclk = myclk + 1;
% display time, in seconds, on GUI
if mod((myclk*recording_timer_period),1)==0 % check if this multiply is a natural number
    set(handles.text10,'String', num2str(myclk*recording_timer_period));
    % old: datestr(clock,14) ) w 14 indicates format of HH:MM:SS PM
end
% record sensor position of the movements
myposition = get(handles.text6,'Position'); % get sensor position
left_position(1,myclk) = myclk*recording_timer_period;
left_position(2,myclk) = myposition(2);
left_position(3,myclk) = newton;

myposition_right = get(handles.text15,'Position'); % get sensor position

```

```

right_position(1,myclk) = left_position(1,myclk); % myclk*recording_timer_period;
right_position(2,myclk) = myposition_right(2);
right_position(3,myclk) = newton_right;

% record ball position
ball_position(1,myclk)=distance;

% record position of cursors
myguidance_left = get(handles.text22,'Position'); % get sensor position
guidance_left_position(1,myclk) = myguidance_left(2)-dist_UL; % get btw text22 and text24

% myguidance_right = get(handles.text23,'Position'); % get sensor position
% guidance_right_position(1,myclk) = myguidance_right(2);

% record targets for movements
if (get(handles.edit2,'BackgroundColor')==[.314 .318 .314]) % edit2 is start-bar
    left_start_position(myclk) = depart_pos(2); % start-bar appears
    left_target_position(myclk) = 0; % target-bar disappears
else
    left_start_position(myclk) = 0; % start-bar disappears
    left_target_position(myclk) = target_pos(2); % target-bar appears
end

% guidance cursors
if (get(handles.edit2,'BackgroundColor')==[.314 .318 .314]) % edit2 is start-bar
    % add the guidance-square part
    guidance_sq_count_reaching_back = guidance_sq_count_reaching_back+1;
    guidance_sq_pos_left = get(handles.text22,'Position');
    %   guidance_sq_pos_right = get(handles.text23,'Position');
    if guidance_sq_count_reaching_back<=delay % in this period, just keep guidance-sq at target-
bar
        set(handles.text22,'Position',[guidance_sq_pos_left(1) guidance_pos_reach_back_left(1)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(1)-2*dist_UL guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
        %   set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(1) guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);

```



```

elseif guidance_sq_count_reaching_back>(delay+length(guidance_pos_reach_back_left)) %
in this period, just keep guidance-sq at start-bar
    set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(length(guidance_pos_reach_back_left)) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
    set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(length(guidance_pos_reach_back_left))-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
    %    set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(length(guidance_pos_reach_back_right))
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
    else % in this period, update position of guidance-sq every pulse of 100Hz
        set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(guidance_sq_count_reaching_back-delay)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(guidance_sq_count_reaching_back-delay)-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        %    set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(guidance_sq_count_reaching_back-20)
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
    end
    guidance_sq_count_reaching_out = 0; % reset to start new rep
else
    % add the guidance-square part
    guidance_sq_count_reaching_out = guidance_sq_count_reaching_out+1;
    guidance_sq_pos_left = get(handles.text22,'Position');
    %    guidance_sq_pos_right = get(handles.text23,'Position');
    if guidance_sq_count_reaching_out<=delay % in this period, just keep guidance-sq at start-bar
        set(handles.text22,'Position',[guidance_sq_pos_left(1) guidance_pos_reach_out_left(1)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(1)-2*dist_UL guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        if guidance_sq_count_reaching_out==1
            sound(sin(linspace(0, .2*400*2*pi, round(.2*800))),800);
        end
    %    set(handles.text23,'Position',[guidance_sq_pos_right(1) guidance_pos_reach_out_right(1)
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);

```

```

elseif guidance_sq_count_reaching_out>(delay+length(guidance_pos_reach_out_left)) % in
this period, just keep guidance-sq at target-bar
    set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(length(guidance_pos_reach_out_left)) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
    set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(length(guidance_pos_reach_out_left))-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
%    set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_out_right(length(guidance_pos_reach_out_right))
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
    else % in this period, update position of guidance-sq every pulse of 100Hz
        set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(guidance_sq_count_reaching_out-delay) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(guidance_sq_count_reaching_out-delay)-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
%    set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_out_right(guidance_sq_count_reaching_out-20) guidance_sq_pos_right(3)
guidance_sq_pos_right(4)]);
    end
    guidance_sq_count_reaching_back = 0;
end

guidata(handles.guifig, handles);

```

```

function NoBB_LRTmrFcn(src,event,handles)

```

```

handles = guidata(handles);

```

```

global dist_UL;

```

```

global newton;

```

```

global newton_right;

```

```

global left_position;
global left_start_position;
global left_target_position;
global right_position;
% global right_start_position;
% global right_target_position;

global myclk;

global guidance_pos_reach_out_left;
global guidance_pos_reach_back_left;
% global guidance_pos_reach_out_right;
% global guidance_pos_reach_back_right;
global guidance_sq_count_reaching_out;
global guidance_sq_count_reaching_back;
% global distance;
% global ball_position;
global guidance_left_position;
% global guidance_right_position;

global recording_timer_period

global delay;

depart_pos=get(handles.edit2,'Position');
target_pos=get(handles.edit3,'Position');

% depart_right_pos=get(handles.edit5,'Position');
% target_right_pos=get(handles.edit7,'Position');

myclk = myclk + 1;
% display time, in seconds, on GUI
if mod((myclk*recording_timer_period),1)==0 % check if this multiply is a natural number
    set(handles.text10,'String', num2str(myclk*recording_timer_period));
    % old: datestr(clock,14) ) w 14 indicates format of HH:MM:SS PM
end
% record sensor position of the movements
myposition = get(handles.text6,'Position'); % get sensor position
left_position(1,myclk) = myclk*recording_timer_period;

```

```

left_position(2,myclk) = myposition(2);
left_position(3,myclk) = newton;

myposition_right = get(handles.text15,'Position'); % get sensor position
right_position(1,myclk) = myclk*recording_timer_period;
right_position(2,myclk) = myposition_right(2);
right_position(3,myclk) = newton_right;

% % record ball position
% ball_position(1,myclk)=distance;

% record position of cursors
myguidance_left = get(handles.text22,'Position'); % get sensor position
guidance_left_position(1,myclk) = myguidance_left(2)-dist_UL; % get btw text22 and text24

% myguidance_right = get(handles.text23,'Position'); % get sensor position
% guidance_right_position(1,myclk) = myguidance_right(2);

% record targets for movements
if (get(handles.edit2,'BackgroundColor')==[.314 .318 .314]) % edit2 is start-bar
    left_start_position(myclk) = depart_pos(2); % start-bar appears
    left_target_position(myclk) = 0; % target-bar disappears
else
    left_start_position(myclk) = 0; % start-bar disappears
    left_target_position(myclk) = target_pos(2); % target-bar appears
end

% guidance cursors
if (get(handles.edit2,'BackgroundColor')==[.314 .318 .314]) % edit2 is start-bar
    % add the guidance-square part
    guidance_sq_count_reaching_back = guidance_sq_count_reaching_back+1;
    guidance_sq_pos_left = get(handles.text22,'Position');
    % guidance_sq_pos_right = get(handles.text23,'Position');
    if guidance_sq_count_reaching_back<=delay % in this period, just keep guidance-sq at target-
bar
        set(handles.text22,'Position',[guidance_sq_pos_left(1) guidance_sq_pos_reach_back_left(1)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);

```

```

        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(1)-2*dist_UL guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
%      set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(1) guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
        elseif guidance_sq_count_reaching_back>(delay+length(guidance_pos_reach_back_left)) %
in this period, just keep guidance-sq at start-bar
            set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(length(guidance_pos_reach_back_left)) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
            set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(length(guidance_pos_reach_back_left))-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
%      set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(length(guidance_pos_reach_back_right))
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
        else % in this period, update position of guidance-sq every pulse of 100Hz
            set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(guidance_sq_count_reaching_back-delay)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
            set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_back_left(guidance_sq_count_reaching_back-delay)-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
%      set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_back_right(guidance_sq_count_reaching_back-20)
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
        end
        guidance_sq_count_reaching_out = 0; % reset to start new rep
    else
        % add the guidance-square part
        guidance_sq_count_reaching_out = guidance_sq_count_reaching_out+1;
        guidance_sq_pos_left = get(handles.text22,'Position');
%      guidance_sq_pos_right = get(handles.text23,'Position');
        if guidance_sq_count_reaching_out<=delay % in this period, just keep guidance-sq at start-bar
            set(handles.text22,'Position',[guidance_sq_pos_left(1) guidance_pos_reach_out_left(1)
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
            set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(1)-2*dist_UL guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
            if guidance_sq_count_reaching_out==1

```

```

        sound(sin(linspace(0, .2*400*2*pi, round(.2*800))),800);
    end
%     set(handles.text23,'Position',[guidance_sq_pos_right(1) guidance_pos_reach_out_right(1)
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
    elseif guidance_sq_count_reaching_out>(delay+length(guidance_pos_reach_out_left)) % in
this period, just keep guidance-sq at target-bar
        set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(length(guidance_pos_reach_out_left)) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(length(guidance_pos_reach_out_left))-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
%     set(handles.text23,'Position',[guidance_sq_pos_right(1)
guidance_pos_reach_out_right(length(guidance_pos_reach_out_right))
guidance_sq_pos_right(3) guidance_sq_pos_right(4)]);
    else % in this period, update position of guidance-sq every pulse of 100Hz
        set(handles.text22,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(guidance_sq_count_reaching_out-delay) guidance_sq_pos_left(3)
guidance_sq_pos_left(4)]);
        set(handles.text24,'Position',[guidance_sq_pos_left(1)
guidance_pos_reach_out_left(guidance_sq_count_reaching_out-delay)-2*dist_UL
guidance_sq_pos_left(3) guidance_sq_pos_left(4)]);
        guidance_pos_reach_out_right(guidance_sq_count_reaching_out-20) guidance_sq_pos_right(3)
guidance_sq_pos_right(4)]);
    end
    guidance_sq_count_reaching_back = 0;
end

```

```

guidata(handles.guifig, handles);

```

```

function InstTmrFcn(src,event,handles)

```

```

handles = guidata(handles);

```

```

guidata(handles.guifig, handles);

```

## Bibliography

1. Ablitt, Nicholas A., Jianxin Gao, Jennifer Keegan, Lars Stegger, David N. Firmin, and Guang-Zhong Yang. "Predictive cardiac motion modeling and correction with partial least squares regression." *Medical Imaging, IEEE Transactions on* 23, no. 10 (2004): 1315-1324.
2. Baker, S. N., E. Olivier, and R. N. Lemon. "Coherent oscillations in monkey motor cortex and hand muscle EMG show task-dependent modulation." *The Journal of physiology* 501, no. 1 (1997): 225-241.
3. Baker, S. N., J. M. Kilner, E. M. Pinches, and R. N. Lemon. "The role of synchrony and oscillations in the motor output." *Experimental Brain Research* 128, no. 1-2 (1999): 109-117.
4. Baldissera, Fausto, Viviana Rota, and Roberto Esposti. "Anticipatory postural adjustments in arm muscles associated with movements of the contralateral limb and their possible role in interlimb coordination." *Experimental brain research* 185, no. 1 (2008): 63-74.
5. Bartlett, M. S. "The spectral analysis of point processes." *Journal of the Royal Statistical Society. Series B (Methodological)* (1963): 264-296.
6. Boonstra, T. W., A. Daffertshofer, E. van As, S. van der Vlugt, and P. J. Beek. "Bilateral motor unit synchronization is functionally organized." *Experimental brain research* 178, no. 1 (2007): 79-88.
7. Boonstra, T. W., A. Daffertshofer, J. C. Van Ditschuijzen, M. R. C. Van den Heuvel, C. Hofman, N. W. Willigenburg, and P. J. Beek. "Fatigue-related changes in motor-unit

synchronization of quadriceps muscles within and across legs." *Journal of Electromyography and Kinesiology* 18, no. 5 (2008): 717-731.

8. Boonstra, Tjeerd W., Bernadette van Wijk, Peter Praamstra, and Andreas Daffertshofer. "Corticomuscular and bilateral EMG coherence reflect distinct aspects of neural synchronization." *Neuroscience letters* 463, no. 1 (2009): 17-21.

9. Bosecker, Caitlyn, Laura Dipietro, Bruce Volpe, and Hermano Igo Krebs. "Kinematic robot-based evaluation scales and clinical counterparts to measure upper limb motor performance in patients with chronic stroke." *Neurorehabilitation and neural repair* 24, no. 1 (2010): 62-69.

10. Buchthal, F., and H. Schmalbruch. "Motor unit of mammalian muscle." *Physiological Reviews* 60, no. 1 (1980): 90-142.

11. Campolo, Domenico, Dino Accoto, Domenico Formica, and Eugenio Guglielmelli. "Intrinsic constraints of neural origin: assessment and application to rehabilitation robotics." *Robotics, IEEE Transactions on* 25, no. 3 (2009): 492-501.

12. Cauraugh, James H., Stephen A. Coombes, Neha Lodha, Sagar K. Naik, and Jeffery J. Summers. "Upper extremity improvements in chronic stroke: coupled bilateral load training." *Restorative neurology and neuroscience* 27, no. 1 (2009): 17-25.

13. Chang, Jyh-Jong, Wen-Lin Tung, Wen-Lan Wu, and Fong-Chin Su. "Effect of bilateral reaching on affected arm motor control in stroke-with and without loading on unaffected arm." *Disability & Rehabilitation* 28, no. 24 (2006): 1507-1516.



14. Chapra, S., and Canale R. "Numerical methods for engineers." *5th ed. McGraw-Hill* (2005): 425-455.
15. Classen, Joseph, Joachim Liepert, Steven P. Wise, Mark Hallett, and Leonardo G. Cohen. "Rapid plasticity of human cortical movement representation induced by practice." *Journal of neurophysiology* 79, no. 2 (1998): 1117-1123.
16. Conway, B. A., D. M. Halliday, S. F. Farmer, U. Shahani, P. Maas, A. I. Weir, and J. R. Rosenberg. "Synchronization between motor cortex and spinal motoneuronal pool during the performance of a maintained motor task in man." *The Journal of Physiology* 489, no. Pt 3 (1995): 917-924.
17. Cunningham, C. L., M. E. Stoykov, and C. B. Walter. "Bilateral facilitation of motor control in chronic hemiplegia." *Acta psychologica* 110, no. 2 (2002): 321-337.
18. Datta, A. K., and J. A. Stephens. "Synchronization of motor unit activity during voluntary contraction in man." *The Journal of physiology* 422, no. 1 (1990): 397-419.
19. De Luca, C. J., R. S. LeFever, M. P. McCue, and A. P. Xenakis. "Behaviour of human motor units in different muscles during linearly varying contractions." *The Journal of physiology* 329, no. 1 (1982): 113-128.
20. de Poel, Harjo J., C. Lieke E. Peper, and Peter J. Beek. "Handedness-related asymmetry in coupling strength in bimanual coordination: furthering theory and evidence." *Acta Psychologica* 124, no. 2 (2007): 209-237.

21. de Xivry, Jean-Jacques Orban, Mohammad Ali Ahmadi-Pajouh, Michelle D. Harran, Yousef Salimpour, and Reza Shadmehr. "Changes in corticospinal excitability during reach adaptation in force fields." *Journal of Neurophysiology* 109, no. 1 (2013): 124-136.

22. Diolaiti, Nicola, Claudio Melchiorri, and Stefano Stramigioli. "Contact impedance estimation for robotic systems." *Robotics, IEEE Transactions on* 21, no. 5 (2005): 925-935.

23. Dodds, G., and N. Glover. "Detailed modelling and estimation of practical robotic parameters for precision control." *Control engineering practice* 3, no. 10 (1995): 1401-1408.

24. Doebelin, E. "System Dynamics: modeling, analysis, simulation, design." *CRC Press* (1998): 527-528.

25. Dowling, James J., Jennifer L. Durkin, and David M. Andrews. "The uncertainty of the pendulum method for the determination of the moment of inertia." *Medical engineering & physics* 28, no. 8 (2006): 837-841.

26. Elali, T. S. "Discrete systems and digital signal processing with Matlab." *CRC Press* (2004): 500-503.

27. Elbert, Thomas, Christo Pantev, Christian Wienbruch, Brigitte Rockstroh, and Edward Taub. "Increased cortical representation of the fingers of the left hand in string players." *Science* 270, no. 5234 (1995): 305-307.

28. Engel, Yaakov, Shie Mannor, and Ron Meir. "The kernel recursive least-squares algorithm." *Signal Processing, IEEE Transactions on* 52, no. 8 (2004): 2275-2285.

29. Evans, Clary, and Stuart N. Baker. "Task-dependent intermanual coupling of 8-Hz discontinuities during slow finger movements." *European Journal of Neuroscience* 18, no. 2 (2003): 453-456.
30. Farmer, S. F., F. D. Bremner, D. M. Halliday, J. R. Rosenberg, and J. A. Stephens. "The frequency content of common synaptic inputs to motoneurons studied during voluntary isometric contraction in man." *The Journal of Physiology* 470, no. 1 (1993): 127-155.
31. Farmer, Simon F., John Gibbs, David M. Halliday, Linda M. Harrison, Leon M. James, Margaret J. Mayston, and John A. Stephens. "Changes in EMG coherence between long and short thumb abductor muscles during human development." *The Journal of physiology* 579, no. 2 (2007): 389-402.
32. Finley, Margaret A., Laura Dipietro, Jill Ohlhoff, Jill Whittall, Hermano I. Krebs, and Christopher T. Bever. "The effect of repeated measurements using an upper extremity robot on healthy adults." *Journal of applied biomechanics* 25, no. 2 (2009): 103-10.
33. Flash, Tamar, and Neville Hogan. "The coordination of arm movements: an experimentally confirmed mathematical model." *The journal of Neuroscience* 5, no. 7 (1985): 1688-1703.
34. Garry, M. I., R. E. Van Steenis, and J. J. Summers. "Interlimb coordination following stroke." *Human movement science* 24, no. 5 (2005): 849-864.

35. Goodman, D., R. B. Kobayashi, and J. A. Kelso. "Maintenance of symmetry as a constraint in motor control." *Canadian journal of applied sport sciences. Journal canadien des sciences appliquées au sport* 8, no. 4 (1983): 238.
36. Goto, Satoru, Tatsumi Usui, Nobuhiro Kyura, and Masatoshi Nakamura. "Forcefree control with independent compensation for industrial articulated robot arms." *Control engineering practice* 15, no. 6 (2007): 627-638.
37. Gross, Joachim, P. A. Tass, S. Salenius, R. Hari, H-J. Freund, and A. Schnitzler. "Cortico-muscular synchronization during isometric muscle contraction in humans as revealed by magnetoencephalography." *The Journal of physiology* 527, no. 3 (2000): 623-631.
38. Guyton and Hall, *Textbook of Medical Physiology: Enhanced E-book*. Elsevier Health Sciences, 2010.
39. Halliday, D. M., B. A. Conway, L. O. D. Christensen, N. L. Hansen, N. P. Petersen, and Jens Bo Nielsen. "Functional coupling of motor units is modulated during walking in human subjects." *Journal of neurophysiology* 89, no. 2 (2003): 960-968.
40. Halliday, D. M., J. R. Rosenberg, A. M. Amjad, P. Breeze, B. A. Conway, and S. F. Farmer. "A framework for the analysis of mixed time series/point process data—theory and application to the study of physiological tremor, single motor unit discharges and electromyograms." *Progress in biophysics and molecular biology* 64, no. 2 (1996): 237-278.
41. Halliday, David M. "NeuroSpec 2.0 User Guide." (2008).

42. Hari, Riitta, and Stephan Salenius. "Rhythmical corticomotor communication." *Neuroreport* 10, no. 2 (1999): R1.
43. Harris-Love, Michelle L., Sandy McCombe Waller, and Jill Whitall. "Exploiting interlimb coupling to improve paretic arm reaching performance in people with chronic stroke." *Archives of physical medicine and rehabilitation* 86, no. 11 (2005): 2131-2137.
44. Hatzitaki, Vassilia, and Patricia McKinley. "Effect of single-limb inertial loading on bilateral reaching: interlimb interactions." *Experimental brain research* 140, no. 1 (2001): 34-45.
45. Heather Mudie, Thomas A. Matyas, M. "Can simultaneous bilateral movement involve the undamaged hemisphere in reconstruction of neural networks damaged by stroke?." *Disability & Rehabilitation* 22, no. 1-2 (2000): 23-37.
46. Huang, Felix C., James L. Patton, and Ferdinando A. Mussa-Ivaldi. "Manual skill generalization enhanced by negative viscosity." *Journal of neurophysiology* 104, no. 4 (2010): 2008-2019.
47. Hwang, Eun Jung, Maurice A. Smith, and Reza Shadmehr. "Adaptation and generalization in acceleration-dependent force fields." *Experimental brain research* 169, no. 4 (2006): 496-506.
48. Inoue, Ryotaro, Kazuhiko Miwa, Haruhisa Kitano, Atsutaka Maeda, Yasuhiko Odate, and Eiji Tanabe. "Highly accurate and real-time determination of resonant characteristics: Complex linear regression of the transmission coefficient." *Microwave Theory and Techniques, IEEE Transactions on* 52, no. 9 (2004): 2163-2168.

49. Jackson, G. M., S. R. Jackson, M. Husain, M. Harvey, T. Kramer, and L. Dow. "The coordination of bimanual prehension movements in a centrally deafferented patient." *Brain* 123, no. 2 (2000): 380-393.

50. Jensen, O., P. Goel, N. Kopell, Marjatta Pohja, Riitta Hari, and B. Ermentrout. "On the human sensorimotor-cortex beta rhythm: sources and modeling." *Neuroimage* 26, no. 2 (2005): 347-355.

51. Kar, M., J. Hornkohl, and W. Farmer. "A new approach to Fourier analysis of randomly sampled data using linear regression technique." In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'81.*, vol. 6, pp. 89-93. IEEE, 1981.

52. Kelso, J. Scott, Dan L. Southard, and David Goodman. "On the coordination of two-handed movements." *Journal of Experimental Psychology: Human Perception and Performance* 5, no. 2 (1979): 229.

53. Kilbreath, S. L., J. Crosbie, C. G. Canning, and M-J. Lee. "Inter-limb coordination in bimanual reach-to-grasp following stroke." *Disability & Rehabilitation* 28, no. 23 (2006): 1435-1443.

54. Kilner, J. M., S. N. Baker, S. Salenius, V. Jousmäki, R. Hari, and R. N. Lemon. "Task-dependent modulation of 15-30 Hz coherence between rectified EMGs from human hand and forearm muscles." *The Journal of physiology* 516, no. 2 (1999): 559-570.

55. Kilner, James M., Stuart N. Baker, Stephan Salenius, Riitta Hari, and Roger N. Lemon. "Human cortical muscle coherence is directly related to specific motor parameters." *The Journal of Neuroscience* 20, no. 23 (2000): 8838-8845.

56. Krebs, Hermano Igo, Neville Hogan, Mindy L. Aisen, and Bruce T. Volpe. "Robot-aided neurorehabilitation." *Rehabilitation Engineering, IEEE Transactions on* 6, no. 1 (1998): 75-87.
57. Krieger, Don, Seun Onodipe, and Robert J. Sclabassi. "Uses of regression in real time processing of neurophysiological signals." In *Engineering in Medicine and Biology Society, 1996. Bridging Disciplines for Biomedicine. Proceedings of the 18th Annual International Conference of the IEEE*, vol. 4, pp. 1758-1759. IEEE.
58. Lemon, R. N. "Mapping the output functions of the motor cortex." *Signal and Sense New York: Wiley-Liss* (1990): 315-355.
59. Lewis, Gwyn N., and Winston D. Byblow. "Bimanual coordination dynamics in poststroke hemiparetics." *Journal of motor behavior* 36, no. 2 (2004): 174-188.
60. Lewis, Gwyn N., and Winston D. Byblow. "Neurophysiological and behavioural adaptations to a bilateral training intervention in individuals following stroke." *Clinical rehabilitation* 18, no. 1 (2004): 48-59.
61. Li, Yong, Oron Levin, Arturo Forner-Cordero, and Stephan P. Swinnen. "Effects of interlimb and intralimb constraints on bimanual shoulder–elbow and shoulder–wrist coordination patterns." *Journal of neurophysiology* 94, no. 3 (2005): 2139-2149.
62. Li, Yong, Oron Levin, Arturo Forner-Cordero, and Stephan P. Swinnen. "Interactions between interlimb and intralimb coordination during the performance of bimanual multijoint movements." *Experimental brain research* 163, no. 4 (2005): 515-526.

63. Lo, Albert C., Peter D. Guarino, Lorie G. Richards, Jodie K. Haselkorn, George F. Wittenberg, Daniel G. Federman, Robert J. Ringer et al. "Robot-assisted therapy for long-term upper-limb impairment after stroke." *New England Journal of Medicine* 362, no. 19 (2010): 1772-1783.

64. Lotze, Martin, Christoph Braun, Niels Birbaumer, Silke Anders, and Leonardo G. Cohen. "Motor learning elicited by voluntary drive." *Brain* 126, no. 4 (2003): 866-872.

65. Lum, Peter S., Charles G. Burgar, Peggy C. Shor, Matra Majmundar, and Machiel Van der Loos. "Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper-limb motor function after stroke." *Archives of physical medicine and rehabilitation* 83, no. 7 (2002): 952-959.

66. McArdle, William D., Franck I. Katch, and Victor L. Katch. *Essentials of exercise physiology*. Lippincott Williams & Wilkins, 2006.

67. McAuley, J. H., S. F. Farmer, J. C. Rothwell, and C. D. Marsden. "Common 3 and 10 Hz oscillations modulate human eye and finger movements while they simultaneously track a visual target." *The Journal of physiology* 515, no. 3 (1999): 905-917.

68. McQuade, Kevin, Michelle L. Harris-Love, and Jill Whitall. "Maximal voluntary isometric elbow flexion force during unilateral versus bilateral contractions in individuals with chronic stroke." *Journal of applied biomechanics* 24, no. 1 (2008): 69-74.

69. Mercier, Catherine, Anne Martine Bertrand, and Daniel Bourbonnais. "Differences in the magnitude and direction of forces during a submaximal matching task in hemiparetic subjects." *Experimental brain research* 157, no. 1 (2004): 32-42.



70. Messier, Sylvie, Daniel Bourbonnais, Johanne Desrosiers, and Yves Roy. "Kinematic analysis of upper limbs and trunk movement during bilateral movement after stroke." *Archives of physical medicine and rehabilitation* 87, no. 11 (2006): 1463-1470.
71. Milner-Brown, H. S., R. B. Stein, and R. Yemm. "The orderly recruitment of human motor units during voluntary isometric contractions." *The Journal of physiology* 230, no. 2 (1973): 359.
72. Mima, Tatsuya, and Mark Hallett. "Corticomuscular coherence: a review." *Journal of Clinical Neurophysiology* 16, no. 6 (1999): 501.
73. Mitra, S. K., and Kaiser J. F. "Handbook for Digital Signal Processing." *John Wiley & Sons* (1993): 932-936.
74. Muellbacher, Wolf, Ulf Ziemann, Babak Boroojerdi, Leonardo Cohen, and Mark Hallett. "Role of the human motor cortex in rapid motor learning." *Experimental Brain Research* 136, no. 4 (2001): 431-438.
75. Muellbacher, Wolf, Ulf Ziemann, Joerg Wissel, Nguyet Dang, Markus Kofler, Stefano Facchini, Babak Boroojerdi, Werner Poewe, and Mark Hallett. "Early consolidation in human primary motor cortex." *Nature* 415, no. 6872 (2002): 640-644.
76. Nef, Tobias, Marco Guidali, and Robert Riener. "ARMin III—arm therapy exoskeleton with an ergonomic shoulder actuation." *Applied Bionics and Biomechanics* 6, no. 2 (2009): 127-142.

77. Nef, Tobias, Matjaz Mihelj, and Robert Riener. "ARMin: a robot for patient-cooperative arm therapy." *Medical & biological engineering & computing* 45, no. 9 (2007): 887-900.
78. Neto, Osmar Pinto, and Evangelos A. Christou. "Rectification of the EMG signal impairs the identification of oscillatory input to the muscle." *Journal of neurophysiology* 103, no. 2 (2010): 1093-1103.
79. Nguyen, Hoi B., and Peter S. Lum. "Compensation for the intrinsic dynamics of the InMotion2 robot." *Journal of neuroscience methods* 214, no. 1 (2013): 15-20.
80. Ohara, Shinji, Takashi Nagamine, Akio Ikeda, Takeharu Kunieda, Riki Matsumoto, Waro Taki, Nobuo Hashimoto et al. "Electrocorticogram–electromyogram coherence during isometric contraction of hand muscle in human." *Clinical neurophysiology* 111, no. 11 (2000): 2014-2024.
81. Omlor, Wolfgang, Luis Patino, Marie-Claude Hepp-Reymond, and Rumyana Kristeva. "Gamma-range corticomuscular coherence during dynamic force output." *Neuroimage* 34, no. 3 (2007): 1191-1198.
82. Oppenheim, A. V., and Schafer R. W. "Digital signal processing." *Englewood Cliffs, NJ Prentice-Hall* (1975): 16-18.
83. Palazzolo, Jerome J., Mark Ferraro, Hermano Igo Krebs, Daniel Lynch, Bruce T. Volpe, and Neville Hogan. "Stochastic estimation of arm mechanical impedance during robotic stroke rehabilitation." *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* 15, no. 1 (2007): 94-103.

84. Pascual-Leone, Alvaro, D. Nguyet, Leonardo G. Cohen, Joaquim P. Brasil-Neto, Angel Cammarota, and Mark Hallett. "Modulation of muscle responses evoked by transcranial magnetic stimulation during the acquisition of new fine motor skills." *Journal of Neurophysiology* 74, no. 3 (1995): 1037-1045.

85. Pascual-Leone, Alvaro, Josep Valls-Solé, Eric M. Wassermann, and Mark Hallett. "Responses to rapid-rate transcranial magnetic stimulation of the human motor cortex." *Brain* 117, no. 4 (1994): 847-858.

86. Penfield, W., and T. Rasmussen. "The cerebral cortex of man. 2 e ed." *New York: The Mac Millan Co* (1952).

87. Perez, Monica A., Bjarke KS Lungholt, Kathinka Nyborg, and Jens B. Nielsen. "Motor skill training induces changes in the excitability of the leg cortical area in healthy humans." *Experimental Brain Research* 159, no. 2 (2004): 197-205.

88. Perez, Monica A., Jesper Lundbye-Jensen, and Jens B. Nielsen. "Changes in corticospinal drive to spinal motoneurons following visuo-motor skill learning in humans." *The Journal of physiology* 573, no. 3 (2006): 843-855.

89. Plowman, Sharon A., and Denise L. Smith. "Digital Image Archive for Exercise Physiology." *Allyn & Bacon* (1998).

90. Poston, Brach, Alessander Danna-Dos Santos, Mark Jesunathadas, Thomas M. Hamm, and Marco Santello. "Force-independent distribution of correlated neural inputs to hand muscles during three-digit grasping." *Journal of neurophysiology* 104, no. 2 (2010): 1141-1154.

91. Qiu, Peng, Warren D D'Souza, Thomas J. McAvoy, and KJ Ray Liu. "Inferential modeling and predictive feedback control in real-time motion compensation using the treatment couch during radiotherapy." *Physics in medicine and biology* 52, no. 19 (2007): 5831.
92. Rice, M. S., and K. M. Newell. "Interlimb coupling and left hemiplegia because of right cerebral vascular accident." *Occupational Therapy Journal of Research* 21, no. 1 (2001): 12-28.
93. Rice, Martin S., and Karl M. Newell. "Upper-extremity interlimb coupling in persons with left hemiplegia due to stroke." *Archives of physical medicine and rehabilitation* 85, no. 4 (2004): 629-634.
94. Riener, Robert, Tobias Nef, and Gery Colombo. "Robot-aided neurorehabilitation of the upper extremities." *Medical and Biological Engineering and Computing* 43, no. 1 (2005): 2-10.
95. Robinson, Richard. "In Mammalian Muscle, Axonal Wiring Takes Surprising Paths." *PLoS biology* 7, no. 2 (2009): e1000050.
96. Rosenberg, J. R., A. M. Amjad, P. Breeze, D. R. Brillinger, and D. M. Halliday. "The Fourier approach to the identification of functional coupling between neuronal spike trains." *Progress in biophysics and molecular biology* 53, no. 1 (1989): 1-31.
97. Salenius, Stephan, Karin Portin, Matti Kajola, Riitta Salmelin, and Riitta Hari. "Cortical control of human motoneuron firing during isometric contraction." *Journal of Neurophysiology* 77, no. 6 (1997): 3401-3405.

98. Sanchis-Moysi, J., F. Idoate, H. Olmedillas, A. Guadalupe-Grau, S. Alayon, A. Carreras, C. Dorado, and J. A. L. Calbet. "The upper extremity of the professional tennis player: muscle volumes, fiber-type distribution and muscle strength." *Scandinavian journal of medicine & science in sports* 20, no. 3 (2010): 524-534.
99. Schabowsky, Christopher N., Alexander W. Dromerick, Rahsaan J. Holley, Brian Monroe, and Peter S. Lum. "Trans-radial upper extremity amputees are capable of adapting to a novel dynamic environment." *Experimental Brain Research* 188, no. 4 (2008): 589-601.
100. Schabowsky, Christopher N., Joseph M. Hidler, and Peter S. Lum. "Greater reliance on impedance control in the nondominant arm compared with the dominant arm when adapting to a novel dynamic environment." *Experimental Brain Research* 182, no. 4 (2007): 567-577.
101. Scheidt, Robert A., and Tina Stoeckmann. "Reach adaptation and final position control amid environmental uncertainty after stroke." *Journal of neurophysiology* 97, no. 4 (2007): 2824-2836.
102. Serrien, Deborah J., and Stephan P. Swinnen. "Intentional switching between behavioral patterns of homologous and nonhomologous effector combinations." *Journal of Experimental Psychology: Human Perception and Performance* 25, no. 5 (1999): 1253.
103. Serrien, Deborah J., Lucy HA Strens, Antonio Oliviero, and Peter Brown. "Repetitive transcranial magnetic stimulation of the supplementary motor area (SMA) degrades bimanual movement control in humans." *Neuroscience letters* 328, no. 2 (2002): 89-92.

104. Smith, Hazel C., Nick J. Davey, Gordana Savic, David W. Maskill, Peter H. Ellaway, and Hans L. Frankel. "Motor unit discharge characteristics during voluntary contraction in patients with incomplete spinal cord injury." *Experimental physiology* 84, no. 6 (1999): 1151-1160.

105. Swinnen, Stephan P., Kris Jardin, Ruud Meulenbroek, Natalia Dounskaia, and Myriam Hofkens-Van Den Brandt. "Egocentric and allocentric constraints in the expression of patterns of interlimb coordination." *Journal of Cognitive Neuroscience* 9, no. 3 (1997): 348-377.

106. Swinnen, Stephan P., Kris Jardin, Sabine Verschueren, Ruud Meulenbroek, Liz Franz, N. Dounskaia, and C. B. Walter. "Exploring interlimb constraints during bimanual graphic performance: effects of muscle grouping and direction." *Behavioural brain research* 90, no. 1 (1998): 79-87.

107. Tagliamonte, Nevio Luigi, Domenico Formica, Domenico Campolo, and Eugenio Guglielmelli. "Coping with intrinsic constraints of neural origin in the design of rehabilitation robots: a preliminary study." In *Neural Engineering, 2009. NER'09. 4th International IEEE/EMBS Conference on*, pp. 124-127. IEEE, 2009.

108. Tagliamonte, Nevio Luigi, Maria Scorcio, Domenico Formica, Domenico Campolo, and Eugenio Guglielmelli. "Effects of Impedance Reduction of a Robot for Wrist Rehabilitation on Human Motor Strategies in Healthy Subjects during Pointing Tasks." *Advanced Robotics* 25, no. 5 (2011): 537-562.

109. Tsumugiwa, Toru, Ryuichi Yokogawa, and Kazunobu Yoshida. "Stability analysis for impedance control of robot for human-robot cooperative task system." In *Intelligent Robots and*

*Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 4, pp. 3883-3888. IEEE, 2004.

110. Ustinova, K. I., J. Fung, and M. F. Levin. "Disruption of bilateral temporal coordination during arm swinging in patients with hemiparesis." *Experimental brain research* 169, no. 2 (2006): 194-207.

111. Utley, A., B. Steenbergen, and D. A. Sugden. "The influence of object size on discrete bimanual co-ordination in children with hemiplegic cerebral palsy." *Disability & Rehabilitation* 26, no. 10 (2004): 603-613.

112. Vallbo, A. B., and J. Wessberg. "Organization of motor output in slow finger movements in man." *The Journal of physiology* 469, no. 1 (1993): 673-691.

113. Vaseghi, S. V. "Advanced digital signal processing and noise reduction." *2nd ed. John Wiley & Sons* (2000): 308-309.

114. Waller, Sandy McCombe, Michelle Harris-Love, Wei Liu, and Jill Whittall. "Temporal coordination of the arms during bilateral simultaneous and sequential movements in patients with chronic hemiparesis." *Experimental brain research* 168, no. 3 (2006): 450-454.

115. Wang, Xinyu, Guilin Zhang, and Ying Chu. "A robust approach to the detection and tracking of small targets with low contrast." In *VLSI Design and Video Technology, 2005. Proceedings of 2005 IEEE International Workshop on*, pp. 296-299. IEEE, 2005.

116. Wang, Z., and P. C. Ching. "Constrained Least Squares Estimation for Position Tracking." In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 4, pp. IV-IV. IEEE, 2006.