

ABSTRACT

Title of dissertation: INERTIAL MOTION CAPTURE SYSTEM
 FOR BIOMECHANICAL ANALYSIS
 IN PRESSURE SUITS

Massimiliano Di Capua, Doctor of Philosophy, 2012

Dissertation directed by: Professor David L. Akin
 Department of Aerospace Engineering

A non-invasive system has been developed at the University of Maryland Space System Laboratory with the goal of providing a new capability for quantifying the motion of the human inside a space suit. Based on an array of six microprocessors and eighteen microelectromechanical (MEMS) inertial measurement units (IMUs), the Body Pose Measurement System (BPMS) allows the monitoring of the kinematics of the suit occupant in an unobtrusive, self-contained, lightweight and compact fashion, without requiring any external equipment such as those necessary with modern optical motion capture systems. BPMS measures and stores the accelerations, angular rates and magnetic fields acting upon each IMU, which are mounted on the head, torso, and each segment of each limb. In order to convert the raw data into a more useful form, such as a set of body segment angles quantifying pose and motion, a series of geometrical models and a non-linear complimentary filter were implemented. The first portion of this work focuses on assessing system performance, which was measured by comparing the BPMS filtered data against rigid body angles measured through an external VICON optical motion capture system. This type of system is the industry standard, and is used here for independent measurement of body pose angles. By comparing the two sets of data, performance metrics such

as BPMS system operational conditions, accuracy, and drift were evaluated and correlated against VICON data. After the system and models were verified and their capabilities and limitations assessed, a series of pressure suit evaluations were conducted. Three different pressure suits were used to identify the relationship between usable range of motion and internal suit pressure. In addition to addressing range of motion, a series of exploration tasks were also performed, recorded, and analysed in order to identify different motion patterns and trajectories as suit pressure is increased and overall suit mobility is reduced. The focus of these evaluations was to quantify the reduction in mobility when operating in any of the evaluated pressure suits. This data should be of value in defining new low cost alternatives for pressure suit performance verification and evaluation. This work demonstrates that the BPMS technology is a viable alternative or companion to optical motion capture; while BPMS is the first motion capture system that has been designed specifically to measure the kinematics of a human in a pressure suit, its capabilities are not constrained to just being a measurement tool. The last section of the manuscript is devoted to future possible uses for the system, with a specific focus on pressure suit applications such in the use of BPMS as a master control interface for robot teleoperation, as well as an input interface for future robotically augmented pressure suits.

INERTIAL MOTION CAPTURE SYSTEM
FOR BIOMECHANICAL ANALYSIS IN PRESSURE SUITS

by

Massimiliano Di Capua

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2012

Advisory Committee:

David L. Akin, Ph.D., Advisor and Chair

Arthur T. Johnson, Ph.D., Dean's Representative

Norman M. Wereley, Ph.D.

Ray Sedwick, Ph.D.

Robert M. Sanner, Ph.D.

© Copyright by
Massimiliano Di Capua
2012

Dedication

To my dearest love Katharina,
for her unconditional support and affection and for enduring three long years apart.

To my family,
for giving me all the love and opportunities a man could only dream of.

Acknowledgments

Many years have passed since i first set foot at the University of Maryland, I remember i was scared, eager to learn and to prove myself but at the same time confused on what to focus on. Now as the conclusion of this wonderful voyage comes to a close, I began to think back at the endless list of challenges and the wonderful people that guided and accompanied me through it. Where to start... nothing better than my first day of classes. It feels like yesterday, my English was very rusty, I knew no one and most of all I didn't know yet but I was about to meet my future mentor. Shane Jacobs was the quiet, serious and concentrated type of student sitting at the back of the room. As soon as the first class was over, still half dizzy from the math overload of that first lecture, I had to run to the next class, and for my own surprise, as soon as I walked in I saw... again... Shane, only this time he was behind the lecturer desk. The lecture began and by the time it was over, my confusion on my future plans was gone. I wanted to focus on Space Human Factors and I wanted to join the SSL! The plan was simple.. or at least so I thought... I had to meet Dr. Akin and demonstrate to him that I was worthy to be part of his team. The hunt began. Dr. Akin is one of the busiest and hardest man to get a hold of I have ever met (later I learned his habits and it became much easier....). In brief, the hunt took a while... The first time I met Dr. Akin it was a huge surprise. He was welcoming, listened to my wishes and was genuinely interested. That day he immediately put me to work. He gave me a small LCD screen and told me to make it become a display interface for MX-2. I couldn't disappoint him so I got to work that same day. The first project lasted about a month in which time I got to meet the rest of the crew. Martin Stollen was the student in charge of advanced controls and displays but most importantly, he became a great friend and mentor. Martin thought me how to properly design and

implement experimental protocols involving human test subjects and spent endless hours training and inspiring me. I will never forget his Fitt's law tests and his white board full of dots and lines where he would try any new ideas he had. Shane was the MX-2 guru, and since I was working on an MX-2 subsystem I followed his lead. Shane is an extremely professional person, very organized and always up to something. Today I can say that he is perhaps the person who has shown me what a leader should be, and I took his example. As my first year and a half passed, and my Master's adventure was coming to a conclusion, sad news arrived. Martin was leaving. During my masters thesis I finally set foot in the GSO (Graduate Student's Office). I finally had my own desk, computer and a really interesting project on Augmented Reality. Sharon Singer sat behind me, she was quiet, and very concentrated. I was the opposite... my desk was full of electronics components, tools and I was frenetically trying to get my code to compile, I was running around and making a "touch" of noise, and I am pretty sure this was upsetting her... especially when I started tapping on my desk at the rhythm of the music in my headphones or loudly chatting with the rest of the crew. That was also the time when I first had the pleasure to meet Barrett Dillow. Barrett, the grumpy, software guru who occasionally slept in his office, and that I continuously bothered for guidance. Possibly one of the most honest and open minded people I ever met as well as a great friend and colleague. Kate McBrian, Nick D'Amore, Connie Ciarlegio and Nick Limparis were the rest of the crew at the time, always complaining (except for Neo... he never spoke!) but always up for fun times. We all came a long way since then, and although we never really worked on a project together, they were the key people that made my SSL experience as great as it has been. Thank you all! When Martin left, and with Shane getting close to graduating, It was my time to take over. A new project had been assigned to us, it was time to look at space habitats. Dr.

Akin called an initial meeting and announced that Adam Mirvis and me would be taking charge of the project. I was in my first year as a PhD student while Adam was in his first year of his Masters, so I was given my first lead role. I was excited and couldn't wait to get started. This was the time when my relationship with Dr. Akin ended and a new figure appeared, Dave was my new advisor and inspiration. When I think of those times I think of our endless and very (very) frequent brainstorming meetings where nothing was too crazy or impossible. The multitude of prototypes and concepts that were developed and the... tons (literally) of ice that had to be chipped off the bottom of the old NBF tank outside in the boneyard as well as the sand to be moved in the new Moonyard... As every day passed I felt I was growing professionally and becoming more and more like my example... and I was and still am very proud of it!! The minimum functionality habitat project was a great success, the team worked wonders and the atmosphere was truly unbelievable! Dave, Adam, Kevin, Omar and William were the best team of engineers I have ever had the honour of working with! Thank you for that unforgettable time and extreme dedication! Since then things picked up even more for the human factors research group, first the LASER grant, then the 2011 and 2012 X-Hab challenges and finally my dissertation work. In this time, I was pleased to have the chance to work with great people like Nick D'Amore, Justin Brannan, Amanda Salmoiraghi, Srikanth Saripalli, Kip Hodges and many others. I will never forget the three insane adventures in the Arizona Desert especially when Kevin and me decided it was a good idea to drive from College Park, MD to Tempe, AZ in just under 44 hours...

The past three years have passed in a flash and I wish to thank all the people that participated in all those activities. First and foremost, Kevin Davis, a great friend and colleague, I wish you good luck for all your future quests, never give up and never stop

dreaming! I also wish to thank Dr. Derek Paley, Dr. Humbert, Greg Gremillion the entire CDCL (Collective Dynamics and Control Laboratory) and AVL (Autonomous Vehicle Laboratory) for their availability and support with their optical motion capture system, the David Clark Company and ILC Dover for allowing us to evaluate BPMS on their suits.

Six years of endless and exciting times are very hard to summarize especially when trying to give justice to all the wonderful people that made it all possible, but I am still missing a few of them, perhaps the most important of all...

To my Dad and my Mom, there are no words to describe my gratitude to you. No matter what, you were always there for me especially when I needed you most even though you were so far away. Now more than ever I think I finally understood how hard it has been for you. Thank you Dad for your...."kind" words of comfort when the times were grey and I was losing my motivation. Thank you Mom for reminding me EVERY day that I had to finish my dissertation! You know me too well and for as hard as it has been, without you I could have never done it. To Luca my little brother, my grandparents and all my family, thank you for making me feel close and loved and not alone even if I was very far away. Finally to my love Kathi. You are the strongest person I know. It is mostly for you that all this has been possible. You give me the strength and love to keep pushing and the desire to be back in your arms was the strongest motivation for me to complete this journey. After those three long years apart and after feeling on my skin what it means not to have you close, as soon as I will hold you again in my arms I will never let you go!

Today more than any other day, even with it's ups and downs, I am proud to have been part of it all. The United States, the University of Maryland and most of all the SSL have left a huge mark in my heart, hopefully I have also left a small one in the hearts of

the people that travelled with me in this fantastic journey. To you all, Thank You.

”Aut inveniam viam aut faciam”

Massimiliano Di Capua

Table of Contents

List of Tables	xii
List of Figures	xiii
List of Abbreviations	xvi
1 Introduction and Literature Review	1
1.1 Introduction	1
1.2 Outline of Dissertation	3
1.3 Significant Contributions	4
1.4 Literature Review	6
1.4.1 Introduction to Pressure Suits	7
1.4.1.1 Mobility System	8
1.4.2 Pressure Suit Analysis	11
1.4.2.1 Range of Motion	11
1.4.2.2 Activity Analysis	13
1.4.2.3 Workload	13
1.4.2.4 Forces and Torques	15
1.4.2.5 Human Integration	16
1.4.3 Human Motion Capture Technologies	17
1.4.3.1 Video and Photographic Analysis	17
1.4.3.2 Optical Motion Capture	18
1.4.3.3 Inertial Motion Capture	20
1.4.3.4 Robot Assisted and Mechanical Measurement Systems	21
1.4.4 Measurement Comparison Between Motion Capture Systems	22
1.4.5 Environmental Simulations	23
1.4.6 MEMS Sensors	26
1.4.6.1 Accelerometers	27
1.4.6.2 Magnetometers	28
1.4.6.3 Gyroscopes	30
1.4.7 The human system and pressure suits	32
2 BPMS Design, Implementation, and Performance Assessment	36
2.1 BPMS Garment	39
2.1.1 Garment Selection and Sizing	39
2.1.2 Sensor Placement and Cable Routing	40
2.2 Electronics	43
2.2.1 System Definition and Requirements	44
2.2.2 Sensor Selection	48
2.2.2.1 Accelerometer	50
2.2.2.2 Magnetometer	50
2.2.2.3 Gyroscope	51
2.2.3 Wireless interface	52
2.2.4 Power distribution	57
2.2.5 System architecture	60
2.3 Software	65

2.3.1	Embedded software	65
2.3.1.1	Flashing the Arduino Bootloader	66
2.3.1.2	Sensors cluster logic	68
2.3.1.3	Arbiter logic	69
2.3.2	Main software package	70
2.3.2.1	Standard Data View	73
2.3.2.2	Custom plots generation	74
2.3.2.3	3D human model views	74
2.3.2.4	File operations	76
2.3.2.5	Reset Zero function	76
2.3.2.6	Realtime monitoring of errors and filter timing	76
2.3.2.7	Sensor mode with UDP server capabilities	77
2.3.2.8	UDP client mode	78
2.3.2.9	File mode	78
2.3.3	Additional tools developed	79
2.3.3.1	IMU class	79
2.3.3.2	3D visualization subroutines	80
2.3.3.3	Translator scripts	80
2.4	System calibration	81
2.4.1	Initial system calibration, determination of sensor offsets	82
2.4.1.1	Accelerometer calibration	82
2.4.1.2	Magnetometer calibration	84
2.4.1.3	Gyroscope calibration	85
2.4.1.4	BPMS prototype calibration	86
2.4.2	Initial pose calibration for sensor alignment	87
2.4.2.1	Initial rotation of the sensor readings	89
2.4.2.2	Pose calibration	90
2.5	Complimentary filter	91
2.6	System verification and performance assessment	100
2.6.1	Experimental setup	100
2.6.2	Results	103
2.6.2.1	Attitude estimate performance and IMU filter gains	103
2.6.2.2	Position estimate performance	122
2.6.2.3	Comparison between VICON and the BPMS suit attitude estimates	125
2.7	Ancillary systems	125
3	Experimental evaluations	130
3.1	Converting IMU attitude to body joint angles	131
3.2	Pressure suits	137
3.2.1	David Clark prototype testbed suit	137
3.2.2	David Clark CHAPS (Contingency Hypobaric Astronaut Protective Suit)	137
3.2.3	ILC Dover Launch and reentry I-Suit	139
3.3	Kinematic evaluation of three pressure suits: Range of motion	140
3.3.1	Data plots	140
3.3.2	Discussion	150
3.4	Geological sortie evaluation	153

3.4.1	Data acquisition	153
3.4.2	Data analysis	154
3.4.2.1	Discussion	159
4	Future Work and Conclusions	162
4.1	Future Work and Potential Applications for BPMS	162
4.1.1	Attitude Estimation Filters	163
4.1.2	Electronics upgrades	163
4.1.3	Operations in Microgravity Environments	166
4.1.4	Operations in Partial-Gravity Environments in the Absence of a Planetary Magnetic Field	167
4.1.5	Human factors systems integration evaluations	168
4.1.6	Exoskeletal suits feedback interface	168
4.1.7	Pressure suit based robot teleoperations	169
4.1.8	BPMS as an interactive interface	171
4.1.9	Non pressure suit related applications	172
4.2	Conclusions	172
A	Early BPMS Prototypes	175
A.1	Instrumented Space Suit Glove	175
A.1.1	Background	176
A.1.1.1	The human hand, and the space suit glove	176
A.1.1.2	State of the art in hand performance measuring systems	179
A.1.2	Design Phase: Trade Studies and System Specifications	181
A.1.3	Manufacturing Process and Software Development	185
A.1.4	Preliminary Testing and System Performance Evaluation	187
A.1.5	Conclusions	188
A.1.6	Future Work	189
A.2	Concepts for advanced pressure suit controls and displays	191
A.2.1	MX-A/B Unpressurized Space Suit Simulators	191
A.2.1.1	Pressure Garment Simulators	193
A.2.1.2	On-Board Computer and Audio Loop	195
A.2.1.3	Helmet Assembly	196
A.2.1.4	Backpack Assembly	197
A.2.2	Advanced Controls	198
A.2.2.1	Multi-Functional Wrist Pad	199
A.2.2.2	Head Tracking	201
A.2.2.3	Gestural Control Interface	202
A.2.2.4	Speech Recognition	203
A.2.3	Advanced Displays	203
A.2.3.1	Night Operations and Lighting	205
A.2.4	Rover Control Interface Evaluation	206
A.2.5	Conclusion and Future Work	210
A.2.6	Acknowledgments	212

B	Concepts for advanced high mobility pressure suits	213
B.1	Introduction	213
B.2	MX-3 Architecture	215
	B.2.0.1 Soft Versus Hard Joints	215
	B.2.0.2 Joint Degrees of Freedom	216
	B.2.0.3 MX-3 Advanced Mobility System	216
	B.2.1 Integrated Advanced Controls and Displays	226
	B.2.2 Entry Type	228
B.3	Operational Environment and Scenarios	229
B.4	Testing and Performance Evaluation	231
	B.4.0.1 Destructive Hydrostatic Testing	233
B.5	Conclusions	236
B.6	Acknowledgments	237
C	IMU Class source code	238
D	Mathematical Background	249
	D.0.1 Attitude in 3D space	250
	D.0.1.1 Rotation matrices	250
	D.0.2 Single axis rotations	252
	D.0.3 Arbitrary rotations in 3D space	252
	D.0.3.1 Euler Angles and Gimbal Lock	253
	Bibliography	255

List of Tables

2.1	Battery technology comparison table	58
2.2	BPMS prototype sensor calibration values	86
2.3	BPMS theoretical axial position errors on 50th percentile American male .	123
2.4	50th percentile American male body segments lengths	124
3.1	IMU angles to body angles for multi-DOF joints	135
A.1	Finger Torque on a phase VI IMU glove at 4.3psi table	178

List of Figures

1.1	Buzz Aldrin on the lunar surface, Apollo 11, source:NASA	7
1.2	The A7L suit without the protective garments (TMG), source:NASA	9
1.3	Suited/unsuited range of motion on the Mark III suit comparison between different techniques[21]	12
1.4	Metabolic workload measurement rig [67]	14
1.5	Torque profile for single axis joint. Phase VI glove at 4.3 psid during wrist adduction/abduction. [59]	15
1.6	Strobe based photographic analysis [21]	18
1.7	Robotic Space Suit Tester wearing the EMU [22]	22
1.8	Inertial motion capture of the lower limbs and Vicon markers during a comparison study [41]	23
1.9	MEMS single axis accelerometer working principle, [42]	27
1.10	MEMS 3 axis accelerometer sample schematic, [42]	28
1.11	MEMS magnetometer working principle, [92]	29
1.12	MEMS gyroscope working principle, [42]	31
1.13	Human skeletal system (left) and simplified model (right), source: http://paulkrohtherapy.com	33
1.14	Human motion standard nomenclature [21]	34
1.15	Human motion standard nomenclature (continued) [21]	35
2.1	Vicon motion capture setup in the AVL laboratory	37
2.2	BPMS during the 2012 desert FLEAS trials	38
2.3	BPMS sensor locations and cables routing	41
2.4	Lines on non extension on the human body [4]	43
2.5	Instrumented arm during the 2011 Desert FLEAS trials	45
2.6	BPMS Electronics Architecture	47
2.7	Sparkfun SEN-10724 IMU sensor stick. Source:Sparkfun.com	49
2.8	LiFePO4 18650 Rechargeable Cell: 3.2V 1500 mAh [100]	59
2.9	BPMS final iteration main board schematic	61
2.10	BPMS final iteration main board layout	62
2.11	BPMS final iteration sensors board schematic	63
2.12	BPMS final iteration sensors board layout	64
2.13	BPMS main software architecture diagram	65
2.14	BPMS start-up user interface	72
2.15	BPMS standard data view user interface	73
2.16	BPMS selective data plot user interface	74
2.17	BPMS 3D human model view user interface	75
2.18	BPMS Calibration Pose	88
2.19	Sensor test rig	101
2.20	Sensor test bed GUI	102
2.21	BPMS Vs potentiometer Roll mean error as a function of filter gains	104
2.22	BPMS Vs potentiometer Roll mean error as a function of filter gains side views	105
2.23	BPMS Vs potentiometer Roll standard deviation as a function of filter gains	106

2.24	BPMS Vs potentiometer Roll standard deviation as a function of filter gains side views	107
2.25	BPMS Vs potentiometer Pitch mean error as a function of filter gains . . .	108
2.26	BPMS Vs potentiometer Pitch mean error as a function of filter gains side views	109
2.27	BPMS Vs potentiometer Pitch standard deviation as a function of filter gains	110
2.28	BPMS Vs potentiometer Pitch standard deviation as a function of filter gains side views	111
2.29	BPMS Vs potentiometer Yaw mean error as a function of filter gains	112
2.30	BPMS Vs potentiometer Yaw mean error as a function of filter gains side views	113
2.31	BPMS Vs potentiometer Yaw standard deviation as a function of filter gains	114
2.32	BPMS Vs potentiometer Yaw standard deviation as a function of filter gains side views	115
2.33	Roll measurement comparison between potentiometer, Vicon and BPMS . .	117
2.34	Pitch measurement comparison between potentiometer, Vicon and BPMS .	118
2.35	Yaw measurement comparison between potentiometer, Vicon and BPMS . .	119
2.36	Error as a function of period	120
2.37	Standard deviation as a function of period	121
2.38	Tracking of the Head comparison between Vicon and BPMS	126
2.39	Tracking of the Right Leg comparison between Vicon and BPMS	127
2.40	BPMS ancillary systems (Zephyr HXM Bluetooth heart rate sensor and Nexus S)	128
3.1	Human body planes of reference Source:NASA STD-3000	134
3.2	David Clark prototype testbed suit	138
3.3	David Clark CHAPS suit	138
3.4	ILC Dover Launch and reentry I-Suit	139
3.5	Right Shoulder ROM	141
3.6	Left Shoulder ROM	142
3.7	Hip ROM	143
3.8	Right Ankle ROM	144
3.9	Left Ankle ROM	145
3.10	Torso ROM	146
3.11	Head ROM	147
3.12	Right Knee ROM	148
3.13	Left Knee ROM	148
3.14	Right Elbow ROM	149
3.15	Left Elbow ROM	149
3.16	Test subject during the unsuited geological sortie with BPMS, Arizona, 2012	154
3.17	Simple lower body model with heel strike detection snapshot (Asterisk size varies with the magnitude of the acceleration	155
3.18	Sample of the acceleration magnitude profiles for the right and left foot . .	157
3.19	Satellite map of the GPS log, Source:GoogleMaps	158
3.20	BPMS track and GPS track of a simulated geological sortie in the Arizona desert	158
3.21	BPMS 3D track and GPS 3D track of a simulated geological sortie in the Arizona desert	160

A.1	MX-A and Raven	175
A.2	The human hand in the robot perspective	177
A.3	Phase VI IMU glove and power-glove without TMG	177
A.4	Instrumented gloves technology in the years	180
A.5	Voltage divider and operational amplifier circuit	182
A.6	Sensitivity as a function of voltage divider resistance R2	183
A.7	Instrumented hand schematic	184
A.8	Instrumented glove GUI	186
A.9	Instrumented glove calibration curves	188
A.10	Instrumented glove sample hand poses with data view	189
A.11	MX-Alpha	192
A.12	MX-Bravo	193
A.13	Donning MX-B	194
A.14	MX-A and MX-B backpacks	195
A.15	Helmet Assembly	196
A.16	Avionics Architecture	198
A.17	MX-A and MX-B Wristpads	199
A.18	MX-A and MX-B head Tracking System	202
A.19	Arm pose measurement System	202
A.20	Modified z800 visor	204
A.21	Night Operations and lighting system	205
A.22	Course diagram	206
A.23	Experimental results: Execution times	207
A.24	Indoor test course and mini-RAVEN	208
A.25	Mean course performance	209
A.26	NASA TLX test results	209
A.27	Cooper Harper	210
A.28	The SSL 2011 Human Factors Team	212
B.1	Examples of Experimental Arm Sections	218
B.2	Convolute	219
B.3	All-Soft Toroidal joint	219
B.4	Thin wall pressurized Cylinder	222
B.5	Helicoidal Restraint Diagram	224
B.6	MX-2 AR visor and Chest Mounted Display	226
B.7	In-Helmet AR Projection System Concept	228
B.8	NBRF Neck Entry Concept Testing	229
B.9	Moonyard Operations and MX- α	230
B.10	Joint Torques Vs Angle Curves	231
B.11	Joint Torque Vs Angle Measurement Testbed	233
B.12	Water Filled Pressurized Section: Leaks	234
B.13	Burst Test Section	235
B.14	Hydrostatic Test Results	236

List of Abbreviations

AHRS	Attitude and Heading Reference System
AVL	Autonomous Vehicle Laboratory
BPMS	Body Pose Measurements System
CDCL	Collective Dynamics and Control Laboratory
CHAPS	Contingency Hypobaric Astronaut Protective Suit
D-FLEAS	Desert Field Lessons in Engineering And Science
DCM	Direction Cosine Matrix
DOF	Degree(s) of Freedom
EVA	Extra-Vehicular Activity
IDE	Integrated Development Environment
ILC	Intentional Latex Company
IMU	Inertial Measurement Unit
IVA	Intra-Vehicular Activity
I2C	Inter-Integrated Circuit
IR	InfraRed
LASER	Lunar Advanced Science and Exploration Research
MEMS	Micro Electro-Mechanical System
MX-A/B	Maryland eXperimental spacesuit simulator, Alpha/Bravo Variant
MX-1/2/3	Maryland eXperimental spacesuit, mark I/II/III
NBRF	Neutral Buoyancy Research Facility
NBV-1	Neutral Buoyancy Vehicle, mark I
PLSS	Portable Life Support System
psi	Pounds per square inch
psid	Pounds per square inch differential
SSL	Space Systems Laboratory
TMG	Thermal and Micrometeorite protection Garment
UDP	User Datagram Protocol
UMD	University of Maryland

Chapter 1

Introduction and Literature Review

1.1 Introduction

There is arguably no more seminal component of human space flight than extravehicular activity (EVA). The ability to perform useful work in space is largely driven by the restrictions on motion created by the pressure suit. The University of Maryland Space Systems Laboratory (SSL) has a long history of research in space human factors and pressure suits; this work builds upon this heritage with the objective of advancing the understanding of the kinematics of the pressure suit wearer.

In the fall of 2010, the SSL was awarded a grant from the NASA Lunar Advanced Science and Exploration Research (LASER) program. The objective of the grant, involving the collaboration of the SSL with Arizona State University, is to investigate human/robot cooperation in scientific lunar exploration tasks. In order to achieve this research objective, the two universities had to develop all the necessary equipment (rovers and pressure suit simulators, incorporating advanced controls and displays) and conduct a series of field trials in the Arizona desert, focusing on quantifying the effects of robotics on the scientific productivity of planetary surface exploration. As of the completion of this thesis, 3 out of 5 planned field trials have taken place; the results are fully documented in the following publications [29, 56, 61].

While the LASER field trials to date have used unpressurized suit simulators rather than actual pressure suits during the trials, it was clear that using a real pressure suit should be the ultimate goal for these studies. The SSL is not new to designing and

building pressure suits; the Maryland experimental series (MX-1/2) research suits are the proof [60, 63, 68, 89, 90]. Based on the prior knowledge on the subject, the SSL human factors research group began the development of a new pressure suit, MX-3. While MX-3 as of today is still a concept, this initial design phase was of great insight. The team did not initially want to compromise in the suit design; before attempting to build a sub-optimal suit which was most likely going to be based on current or past designs [38, 48, 69, 70, 71], it decided to focus on understanding pressure suits in a more rigorous and insightful way.

A series of prototype pressure suit joints were developed and tested at the SSL, each with positive and negative results. As of today, this research effort is still in progress, and is documented in Appendix B, but it soon became evident that, instead of attempting to build a novel suit from scratch (new joint designs, architecture, materials, etc...) with the hope of designing the ultimate pressure suit, it was more appropriate to take a step back and attempt to optimize the current technology. Instead of designing for all possible tasks, we could produce a huge benefit by optimizing joint placement and orientation and, in return, facilitate the most frequently executed tasks. A thorough literature search with this goal was conducted, and is provided below in the manuscript. Pressure suit joints are designed to be as low torque with as little hysteresis as possible, with a goal of providing an angular range of motion as close as possible to the unconstrained case of nude body range of motion. While this approach of replicating the unsuited performance of the suit wearer on a joint by joint basis is perfectly valid, it doesn't place any emphasis on the combined effect of the placement and orientation of aggregated joints and its effect on complex motions. To inform the design process, we had to understand and measure how the suit alters the wearer's overall mobility. Usually measurements on pressure suits are conducted through optical motion capture or photographic analysis, but they only

give the investigator the ability to measure the exterior motion of the suit, and not the interior motion of the test subject. These techniques are not compatible with routine use in extended field trials, and instead measure simple motions in a laboratory setting.

This gap in current suit instrumentation technology was the main inspiration of this work. Based on prior experience with inertial measurement units, it was immediately seen that an inertial motion capture system would be extremely useful in better understanding the kinematic effects of pressure suits on human motion. After an extensive search, it was clear that a commercial system that was compatible with pressure suit operations was not available and therefore had to be designed, built and tested before even attempting to answer the questions above. The following work will describe this process in detail, and will set the starting point for a more comprehensive investigation to be conducted in the future. This work focuses on developing and validating a noninvasive human kinematics measuring system compatible with extended pressure suit testing, along with analysis tools and methodologies to begin answering some of the multitude of questions that surround the space suit design field.

1.2 Outline of Dissertation

This manuscript is divided in four chapters; it is meant to guide the reader through the four main phases of this work. The first chapter introduces the subject and the relevant background material. This includes an extensive literature review on past and current space suit measurement techniques and their limitations, as well as the proposal of a novel inertial measurement system dubbed the body pose measurement system, or BPMS. The first chapter also includes a brief review of the analytical methods that are at the base of BPMS and introduces the nomenclature and symbolism that will be adopted

throughout this work. The second chapter covers the design and development of BPMS. It includes the design choices and lessons learned during the various iterations required to produce a usable working prototype of the system. This chapter also covers the validation and calibration of BPMS, where data was synchronously recorded with both BPMS and a Vicon system, the industry standard for optical motion. This data were later compared to assess BPMS's accuracy in measuring angular and position estimations of key body parts. Following the system description and its validation, a series of sample evaluations were conducted to demonstrate the ability of this new measurement system to serve its purpose. Chapter Three covers range of motion evaluations conducted on three different pressure suits in order to define their flexibility at various pressure levels. A sample of kinematic analysis from field testing is also presented in order to demonstrate the flexibility and ability of a system such as BPMS to record significant data in environments that up to now were considered extremely difficult if not impossible to study, such as a simulated geological sortie in the Arizona desert. The last chapter will summarize all the findings and lessons learned during the BPMS operations, as well as proposing a series of future applications, modifications to the system, and conceptual research directions for the future of this work. Following the core portion of the manuscript, several appendices have been included to augment the material previously described. These will cover the development and operation of BPMS, and document the parallel work that was done in space suit design that inspired this research.

1.3 Significant Contributions

This work presents a series of novel contributions to the field of space human factors, pressure suit design, and advanced human kinematics measurement systems. Those

contributions can be summarized as follows:

1. **BPMS inertial motion capture system:** A novel inertial motion capture system capable of measuring the pose of a person inside a pressure suit was designed, built, and tested. The system was then evaluated against the industry standard optical motion capture system to determine a series of performance metrics such as accuracy, resolution, drift and lag. In the process, all the embedded electronics and software to acquire, visualize and record the data were developed and tested.
2. **Multi-IMU arbitrary initial pose calibration algorithm:** In a multi-inertial measurement unit (IMU) system like BPMS, the various sensor attitudes must be measured in the appropriate frame. In this work, a single point calibration strategy has been implemented and evaluated to define the initial frames in which the sensor are referenced. This approach provides the system the ability to map, based on a given model, the IMU's orientation to a specific initial pose, thereby allowing the system to reconstruct the the body pose of a test subject.
3. **Kinematic evaluation of a human inside a pressure suit:** An initial range of motion study was conducted on three different NASA pressure suits at different pressures. The scope of the test was to address the effect of internal pressure on range of motion for each specific suit, and then compare the results against the unsuited reference case. In addition to a pure range of motion evaluation, the trajectories followed by the test subjects were also evaluated, and have been compared to address if the tasks were executed in the same way or if the suit altered the complex motion.
4. **Define a methodology to evaluate human joint angles from IMU angles:** A methodology to map human body joint angles from IMU sensor readings was

defined. By using a series of geometrical relationships between the inertial sensors and the constraints of the human body joints, we are able to translate the IMU sensors reading to the canonical set of human body joints, and thereby compare the measurements obtained with BPMS against the current literature.

5. **Kinematics measurement and analysis of a field geologist during a sim-**

ulated sortie: In order to demonstrate that the capabilities of BPMS are not constrained to pressure suits or to laboratory settings, as a target of opportunity, BPMS was used to record the kinematic motions of a field geologist during a sortie in the Arizona desert. The test was primarily aimed to demonstrate that significant data can be acquired by BPMS in extended tests, even in extreme environments, without affecting the mobility or capabilities of the test subject. By being able to acquire the kinematics of a test subject in both suited and unsuited configurations during simulated sorties in analog environments, this type of instrumentation can be used in the future to better inform the design of the next generation of planetary space suits.

1.4 Literature Review

In this section, the reader will be presented with the background information relevant to the present work. Following a general introduction to the subject of pressure suits, an extended literature search was conducted with the aim of better understanding the past and current analysis methodologies and technologies used for studying such systems. All the lessons learned were then used to identify how to contribute to the state of the art, and where to focus for the development of BPMS.

1.4.1 Introduction to Pressure Suits

Perhaps the most ubiquitous image representing human space flight is the picture of Buzz Aldrin standing on the surface of the moon during the Apollo 11 mission [74].

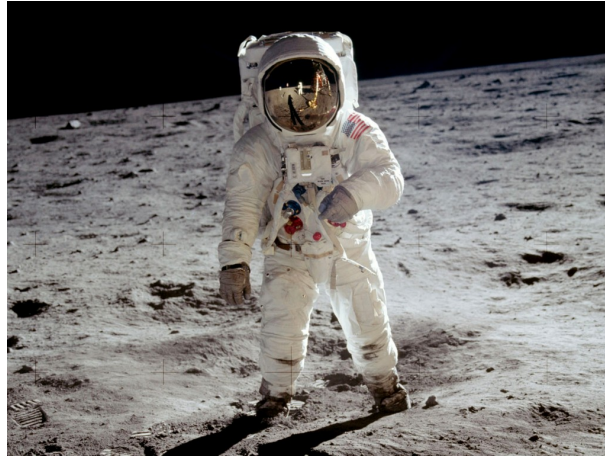


Figure 1.1: Buzz Aldrin on the lunar surface, Apollo 11, source:NASA

In this image, the centerpiece is the A7L space suit. A pressure suit is a fundamental, and extremely complex, system of systems that allows the wearer to survive the most extreme of environments while still retaining most of their mobility, and allowing him or her to conduct useful work in vacuum. A pressure suit can be divided into several functional groups, such as the mobility system, life support system, and command, control, and communications avionics. In this work we will exclusively focus on mobility systems and their analysis, but a few introductory comments should be made for the remaining systems.

In all existing (operational or experimental) pressure suits, we see some sort of life support system. A life support system can be closed, semi-closed or open loop; its main function is to provide the suit wearer with a comfortable environment in which to operate. The life support system needs to accomplish several functions simultaneously, such as

regulating the suit’s internal pressure, maintain a comfortable temperature and humidity level, and dispose of the by-products of human respiration and exercise including CO_2 , heat, and water vapor [75, 77, 78]. In order to do so, the portable life support system (PLSS) relies on a series of subsystems such as the liquid cooling garment (LCG), which is a network of small tubes mounted on a conformal garment worn by the suit wearer that removes excess heat from the user’s body and delivers it to the PLSS for disposal; and the ventilation system, which allows the PLSS to feed “new” air (or O_2 depending on the suit in question [79]) to the wearer through a network of ventilation ducts inside the suit, extracting the “spent” air and either recycling it or venting it overboard. Finally, the avionics system provides the user with a series of functions, such as the ability to communicate with the other parties involved in the operations, as well as controlling and monitoring the PLSS. The avionics in today’s suits are rather crude and simple; while this approach increases the simplicity and reliability of the system, it can greatly be improved. Currently a wide array of research efforts have been focusing on the upgrading and expansion of the avionics in pressure suits and the work in Appendix A and [9, 66] are just a few examples.

1.4.1.1 Mobility System

The mobility system is what allows the space suit user to accomplish meaningful work while in pressurized. It is usually composed of several layers of fabric combined with hard elements, each serving a specific purpose. Pressure suits can vary greatly in architecture; therefore, this portion of the work will be left relatively generic, and will instead focus on the impact and general definition of the mobility system.

The overarching goal of space suit design is to provide a system that does not

hinder human motion, while still ensuring survival in the harsh environment of space. Unfortunately, we are far from this objective, but progress is being made continuously as knowledge of pressure suits is gained. The mobility system of a pressure suit is a collection of single degree of freedom joints that are strategically positioned and oriented on the garment in order to allow the pressurized volume to bend and provide mobility to the wearer. On the other side, the human skeletal system incorporates several multi-degree of freedom joints; it is immediately apparent that a discrepancy in the mapping of the suit mobility systems on the human skeletal system will involve a performance impact. For all intents and purposes a pressure suit is an exoskeleton, which once pressurized becomes rigid and limited to motion at designed articulation points.



Figure 1.2: The A7L suit without the protective garments (TMG), source:NASA

Figure 1.4.1.1 shows the A7L suit without the exterior covering, so all the suit joints

are visible. While modern pressure suits provide comparable single axis ranges of motion on most enabled joints, the discrepancies in mapping between the suit and the wearer produce a series of effects, primary among which is the “programming” phenomenon. Programming refers to the suit-induced variation in the paths that are required in order to move a limb from an initial pose to a desired final pose. In current pressure suits, there are situations when in order to move from an initial pose to a final one or more intermediate steps are required, which would be unnecessary without the suit. While this phenomenon might seem of little significance, it becomes critical for motions such as walking, running, or instinctive body response during an emergency, such as falling down. Effects such as programming can destabilize gait, and alter significantly the energy expenditure required to accomplish a task.

A second critical effect of pressure suit use is damping. Pressure suits are highly damped systems, and are usually modeled as spring-damper systems. The obvious consequence of damping is the dissipation of energy put into kinematic motion, and therefore reduced ability to accomplish meaningful work.

So far, one of the major issues in the design phase of a pressure suit is the lack of a rigorous quantitative standard for requirement verification, as well as the provision of fully informed requirements. Today’s requirements are still based on the single axis testing, and do not take in consideration the previously mentioned issues. Perhaps the rationale behind this lies in the absence of a methodology to investigate how the human wants to move and from this define a more useful set of requirements. Future pressure suit requirements will potentially embrace the concept of requirements based on “use of motion” rather than the canonical “range of motion” approach.

1.4.2 Pressure Suit Analysis

In this section a brief overview of the current and past analysis on pressure suits is provided to identify the current efforts being conducted in the field. The list is by no means comprehensive, but it serves to better define the role of this work in the community.

1.4.2.1 Range of Motion

Range of motion (ROM) defines the extent to which one can move each joint of the body, and is frequently confined to primitive single axis motions. When this concept is introduced in pressure suit analysis, the focus becomes to compare the unsuited ROM to the suited. We know that the mobility system induces constraints in the ROM of the wearer; evaluating the variation between those two cases grants a metric of the quality and performance of the system. The plots below show an example of range of motion evaluation of a pressure suit and the comparison against the unsuited case. Those results will later be used and compared in the range of motion analysis with BPMS. In this study, the Mark III suit was used to conduct a range of motion evaluation with two different methods, photographic analysis and optical motion capture.

Aitchison concludes in her study[21] that, although statistically the results of the two methodologies appear equivalent, for all practical purposes they are vastly different. Based on this comparison, it seems that data must be compared only against data collected in the same manner. Comparing data within subjects does not necessarily improve the accuracy of comparisons across methods, across suit conditions, or within suit conditions. The range of expected values is entirely too variable to allow researchers to understand where in the span of accuracy they fall. Additionally, based on this comparison, one can conclude that statistical analysis is not the ideal means of comparing data; the percent

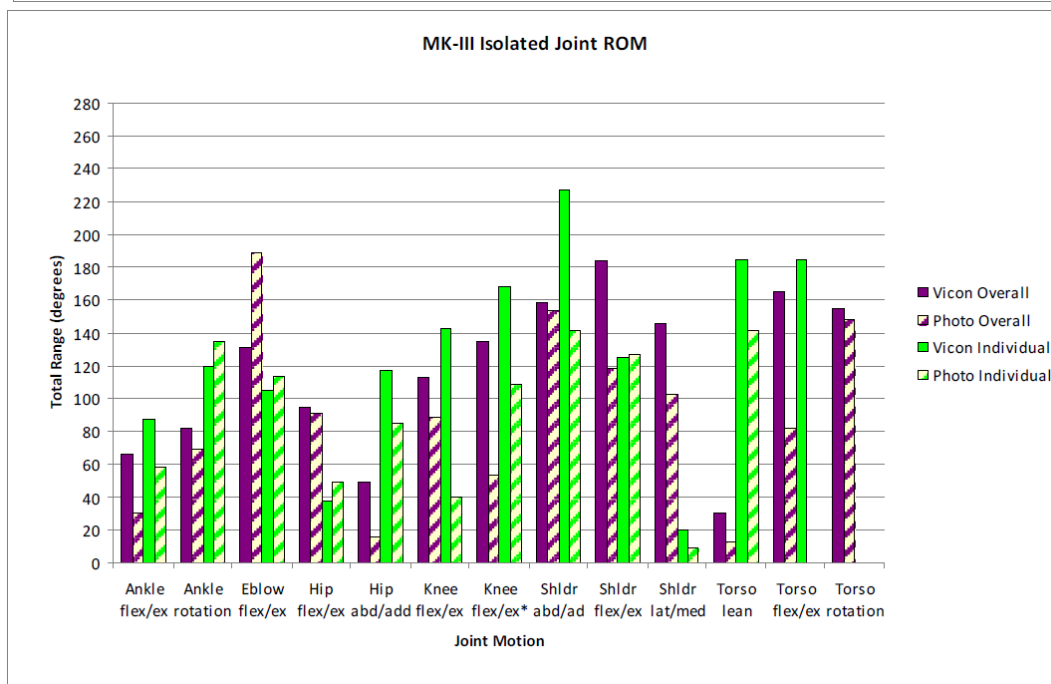
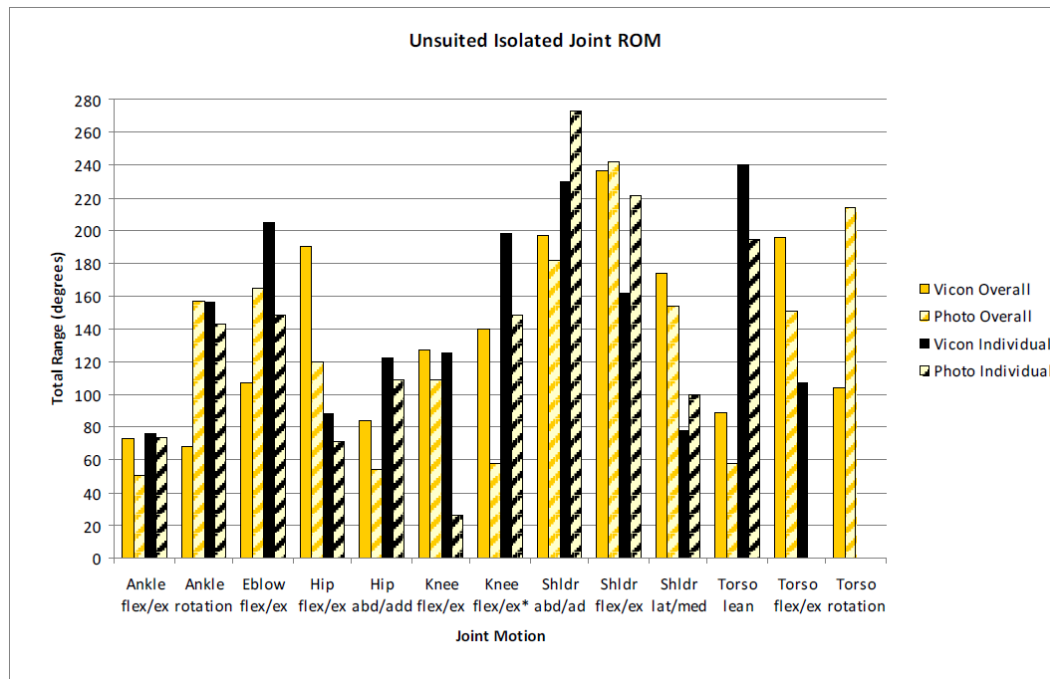


Figure 1.3: Suited/unsuited range of motion on the Mark III suit comparison between different techniques[21]

differences provide a more practical understanding that directly relates to acceptability of space suit mobility system performance [21]. Several other studies have been conducted on this topic[2, 19, 39, 65], which seems to be one of the most important items of interest in the community.

1.4.2.2 Activity Analysis

Activity analysis differs from the ROM studies, since the focus is to observe complex motion. The objective is to map and model how humans accomplish work when in a pressure suit, and again compare the results to the unsuited/unconstrained case. In the past, based on the objectives of the active space missions at that time, the studies have focused on different complex motions [52] such as walking, running, hammering, traversing slopes, entry and exit from a vehicle, etc. Activity analysis was originally used to measure the variation in motion due to a medical condition (illness or amputation)[24, 37, 32, 10], or for athletes monitoring and training[28, 31]; it since has been extended to a much wider array of applications. Activity analysis is usually implemented in conjunction with workload measurements to return a complete picture of how a certain task is executed, and how much effort is required. Additionally in the process, by observing the paths followed by the subjects and comparing them against the "normal/unsuited" case, it is possible to make suggestions and focus training in order to optimize the subjects' energy resources, or to alternatively drive the requirements and architecture of the pressure suit system.

1.4.2.3 Workload

As any physical system, humans can accomplish work at the expense of energy. The quantification of this parameter is of fundamental importance in space mission design,

since it drives what can be asked from the crew. The main metric used to quantify the effort required by the crew to accomplish a task is metabolic workload or oxygen consumption, which are closely related.



Figure 1.4: Metabolic workload measurement rig [67]

The metabolic workload, or the metabolic rate, is the amount of energy per unit time the human body is expending and is directly, but not constantly, proportional to the heat production by the body [67]. In order to accomplish such measurements, an adequate instrumentation system is required, usually consisting in measuring the composition of the air going in the suit and comparing it to the exhaust. By measuring the drop in O_2 and increase in CO_2 along with the airflow, it is possible to estimate the workload through several models [67]. In addition to the measurements above, other additional metrics usually acquired are body temperature, heart rate, and kinematic motion.

1.4.2.4 Forces and Torques

As previously mentioned, due to their nature pressure suits induce new dynamics to the suit bearer. The study of forces and torques required to actuate the pressure suit allows designer to better understand the mechanisms behind those dynamics, and introduces a series of concepts like hysteresis, volume variation torques, friction, and fit. If we attempt to actuate a pressure suit joint, we will immediately notice that a significant force is required. This force is usually proportional to the size of the joint, position, joint technology, and whether or not the joint is covered by the additional layers such as the thermal/micrometeoroid garment (TMG). We will also notice that the joint acts like a very damped spring, as there is an area in the proximity of the neutral position where, if we move the joint, it remains in that position. This phenomenon is due to the nonlinear nature of the torque curve of the joint as a function of bend angle [35, 46, 59].

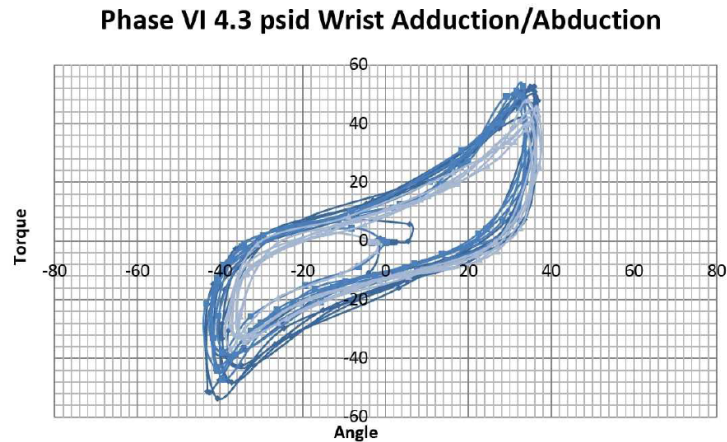


Figure 1.5: Torque profile for single axis joint. Phase VI glove at 4.3 psid during wrist adduction/abduction. [59]

Several models have been explored to map torques, but the models only go as far as describing a specific suit joint, and are hardly generalizable. Those models allow us

to understand the forces in play, but need to be used with caution since they only map quasistatic or static conditions, and rarely attempt to study the impact of dynamic motion. Granted that most EVA operations are conducted relatively slowly, we must also consider that the performance of the suit can vary significantly as its life cycle progresses, as well as due to the conditions in which it is deployed (e.g., in the presence of sand and dust such as on the lunar or Mars surface).

1.4.2.5 Human Integration

Human integration in the past was done very differently from today. Based on the use of either mockups or real systems, test subjects (frequently astronauts) were asked to evaluate the systems and respond to a series of questionnaires and debriefs. Today engineers and project managers search for a more rigorous approach[3]. While surveys and debriefs are still a very important portion of the process, other techniques have made their way in the evaluation of manned systems. Optical motion capture allows engineers to observe details that were previously impossible to visualize, especially when the item of the investigation is in a highly dynamic environment like the investigation of potential injuries [50] induced by improper positioning of equipment, or when operations are conducted in a very occluded and cluttered environment such as a spacecraft [53]. As of today, one of the major issues in human integration evaluations lies in the systems adopted for the measurements. Optical motion capture is far from the best choice for evaluating activities in confined environments, but due to the unavailability of a better suited system, other approaches had to be taken. Today in order to allow for optical motion capture data acquisition, engineers design specific low fidelity transparent mockups of the item of concern in order to minimize obstruction.

1.4.3 Human Motion Capture Technologies

Instrumentation has always played a fundamental role in scientific analysis and research, and human motion capture has been perhaps one of the most challenging applications. Human motion capture, as everything that has to do with humans, is plagued by variability and limitations. Variability is usually a result of individuality; humans move, act, think, behave, and feel differently not only between subjects but also within subjects. Humans also learn from previous experiences and adapt; therefore it is likely that by repeating an experiment twice with the same subjects, you might end up with two different results. From an engineering perspective, humans are the worst system to attempt to investigate, not only because of the previously mentioned variability, but also from a limitations perspective. The human system is very hard to measure, and any instrumentation must generally be non-invasive, and also must attempt not to alter the experiments themselves. Several techniques have been identified and used in the community: the following paragraphs aim to give a brief overview of the various methodologies along with their concept of operations, followed by a few pros and cons of each.

1.4.3.1 Video and Photographic Analysis

Photographic analysis was the the original approach for measuring the motion of human subjects, expanded by video analysis in recent years. The typical setup relies on a camera, a back plane with known subdivision marks, and a strobe. By overexposing the camera film in conjunction with a strobe, it is possible to capture multi-image still photos of the test subjects during their motion. Figure 1.4.3.1 is an example of what the typical overexposed still would be.

Once the image was acquired, it was possible to use a protractor and quantify

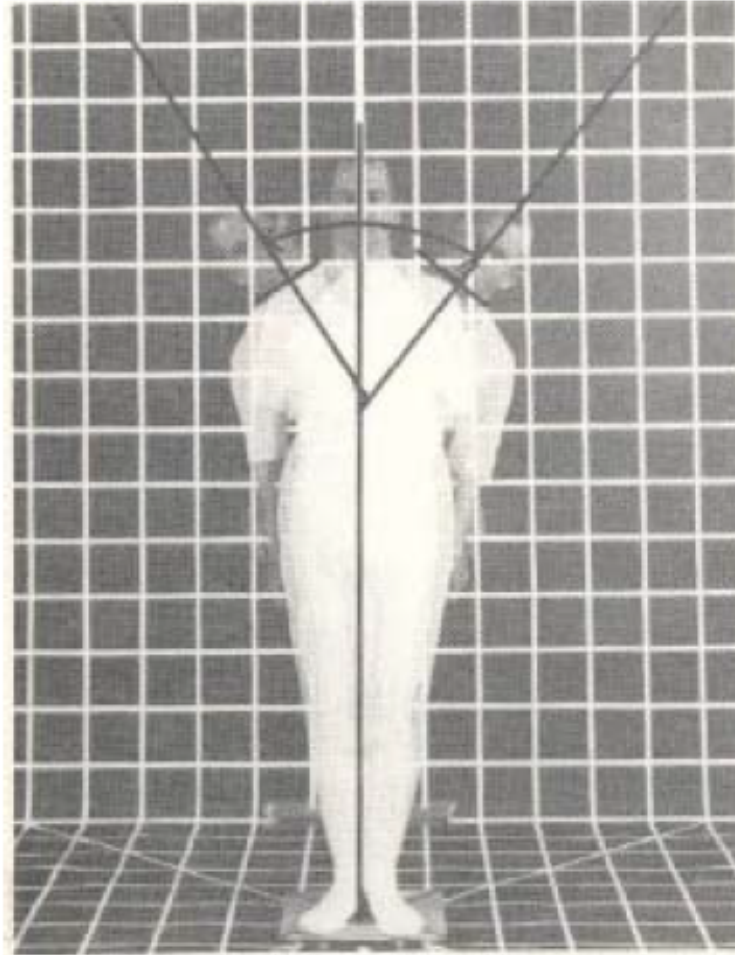


Figure 1.6: Strobe based photographic analysis [21]

motion. This crude process was the standard in both medicine and pressure suit analysis up until optical motion capture made its debut. A very similar technique that was also used was to record video of the motion of interest, and then to isolate the relevant frames in post processing, with analysis similar to photo analysis.

1.4.3.2 Optical Motion Capture

Optical motion capture systems consist of an array of cameras that "look" at a defined volume of interest. The cameras capture high speed video in either the visual or

infrared spectrum. The cameras also usually include a strobe light, which is tuned to the nominal light frequency detected by the cameras. The working principle for the system is relatively simple, and requires a series of markers (retroreflective-coated spheres). The system can detect the position in 3D space of single or multiple markers, as long as the marker is visible from at least 2 or more cameras, depending on the systems used and its current configuration. This type of system directly measures positions of the markers; additional analysis algorithms exist if we wish also to acquire attitude. The attitude of a rigid body can be derived from a series of markers strategically positioned about the body. One must be careful in the arrangement of those markers, and must make sure that during the entire acquisition, the markers remain:

- rigidly located on the body of interest, reducing as much as possible the possibility of changing distances between markers;
- in an asymmetric arrangement to reduce the possibility of detecting singular attitudes;
- as far away from each other as practical – the distance between the markers reduces the computational errors in the attitude and increase the sensitivity of the attitude measurement; and
- visible to the cameras at all times. When tracking a rigid body, there can be the issue of occlusion, and this is particularly true when tracking humans. There are no limitations to the maximum number of markers that define a rigid body (there is though a limitation on the minimum number, since in order to define an attitude we will need at least 3 markers) and usually more markers than the minimum required are used. Unfortunately, using more markers can bring additional analysis issues as

well, such as robustly tracking individual retroreflectors as they move into and out of occlusion.

From the above considerations some of the intrinsic limitations of optical motion capture technology are clear. In addition to the above, the remaining limitations include occlusion, lighting, overhead equipment and cost. Optical motion capture systems are also limited to established test volumes, most typically established inside laboratory spaces. Despite these drawbacks, optical motion capture for pressure suit analysis has generally become the “gold standard” in biomechanics tracking, mostly because of the lack of viable alternatives [64].

1.4.3.3 Inertial Motion Capture

Inertial motion capture is the most recent of the available instrumentation systems, primarily driven by the increasing spread and advancements in microelectromechanical systems (MEMS) technology. By allowing the investigators to collect data at high rates data from multiple, very sensitive, and low cost sensors, along with the implementation of advanced sensor fusion filters, MEMS inertial measurement units have opened the gates to a new frontier in human motion measurement and analysis [64, 40, 62]. Inertial motion capture differs from the the previous methodologies since it allows, through the use of multiple sensors, the measurement of the attitude (but not position) of the human limbs with comparable accuracy to the currently adopted systems [58]. It is still in the prototype stage, and the commercial availability of such systems is limited.

Inertial motion capture has several advantages over optical motion capture and photo analysis; foremost among these are that it does not constrain the users to a laboratory setting, and it does not suffer from occlusion or restrictions on lighting conditions[15].

Relieving these constraints results in extended possibilities for data acquisition applications. Rather than constraining the use of body measurement to laboratory simulations, the data can be acquired directly in the field during real operations. This is true in both the aerospace applications (e.g., measuring the kinematics of a person inside a pressure suit) as well as for medical, athletic, or other applications.

Unfortunately, though, inertial motion capture also has unique drawbacks. Depending on the sensor arrays used, it can suffer from drift or magnetic disturbances, and it also generally requires several layers of post processing in order to map the human body and return a canonical set of angular parameters[93].

1.4.3.4 Robot Assisted and Mechanical Measurement Systems

Several other approaches have been taken to date when it comes to measuring pressure suit kinematics or dynamics. While all the previous applications envisioned the human in the suit, there are several other test-bed approaches that are now commonly used. Force-torque sensors are commonly used to measure the dynamics of sections of pressure suits (mostly gloves, arms and leg sections) [26] but those tests do not take in account the effects of the human limb inside the suit. An extension to those methods can be seen in [23] where a full humanoid pneumatic robot is used to emulate the human in the suit, although only one side is actively articulated in this system. While this approach provides benefits by allowing for highly repetitive measurements, it is only as accurate as the modeled motions that were commanded to the robot. Unfortunately, aside from this work, the kinematics of a human inside a space suit have never been recorded, and therefore it is very hard to verify that the robotic system actually replicates the human motion. Other systems that are common in the literature include passive robotic systems



Figure 1.7: Robotic Space Suit Tester wearing the EMU [22]

equipped with linear potentiometers or encoders [12, 17].

1.4.4 Measurement Comparison Between Motion Capture Systems

As time goes on, several alternatives to optical motion capture have appeared [11, 41, 43, 49] which attempt to compare the estimates so obtained against what is usually considered as the industry standard. Vicon is the most widespread and accurate optical motion capture system, and it is generally accepted as “ground truth” in calibrating alternative systems. In this work, we will test this assumption and address Vicon accuracy and precision through a dedicated electro-mechanical system, which will be used to compare both optical and inertial motion capture.



Figure 1.8: Inertial motion capture of the lower limbs and Vicon markers during a comparison study [41]

1.4.5 Environmental Simulations

None of the above analysis methodologies were ever implemented during actual space missions; as valuable as flight measurement of EVA anthropometrics would be, the technical limitations and substantial cost of implementing flight-rated instrumentation deter efforts to obtain this data. In order to still be able to gain significant insight on pressure suits, most of the current testing is performed under more or less controlled environments that attempt to simulate the environmental conditions that the pressure suit in question will have to operate in for flight. The following list details the most common simulated environments currently used and their application scopes:

1. **Neutral Buoyancy:** Neutral buoyancy (NB) relies on Archimede's lift principle to offset all or some of the gravitational force acting on a body. By appropriately controlling the buoyancy, it is possible to reproduce the dynamic environment of the moon, Mars or free space fairly accurately. NB simulations are limited by hydrodynamic effects, including viscous drag and virtual mass, which become predominant when the object is required to accelerate and move within the medium. Luckily, most EVA operations can be considered quasi-static, and therefore dynamic inac-

curacies due to the hydrodynamic effects tend to be considered acceptable [27]. In addition to pure hydrodynamics, another item of concern in NB simulations is the increased mass. In order to properly ballast an object to reproduce a specific gravitational environment, additional masses are added; this induces changes in the dynamic response of the body. NB is usually used as a “first cut” simulation due to its affordability (in terms of operational costs) when compared to higher fidelity simulations.

2. **Parabolic Flight:** Parabolic flight offers subjects brief periods of actual, rather than simulated reduced gravity by flying a series of parabolic arcs in an airplane. Given the limitations on cabin size, available dynamic volume is usually highly constrained, and the size and number of free-floating components are limited for the safety of the flight crew. On the other hand, parabolic flight produces the most accurate simulation of microgravity in space: during the reduced gravity portions of the flight, the subject and experimental hardware have no predefined preferred orientation or position and exhibit full six degree-of-freedom motion limited by the fuselage, installed equipment, and flight crew. For transport category aircraft, microgravity periods of 20-30 seconds are separated by periods of increased gravity of typically 90 seconds duration. Although a human subject is weightless for short intervals, he or she does not have an opportunity to truly acclimate to the reduced gravity environment, and can be more accurately said to experience a variable-gravity environment.[27]
3. **Gravity Offset:** Gravity offset is the generic term for providing an upwards force to counter some or all of the subject’s weight on Earth. [1]. A number of ap-

proaches have been designed and built over the years, including inclined platforms with tension cables, counterweight rigs with offset ballast (with and without mechanical advantage designed in), and robotically driven active counterbalancing. Each methodology has its advantages and disadvantages, but overall it is considered a very effective technique to test partial gravity. Gravity offset must maintain a vertical overhead position to the test subject to prevent pendulum motions, resulting in relatively small simulation areas. This simulation approach is also limited by the suspension lines, preventing objects in the test environment from passing over the test subject, but gravity offset does eliminate the short time limitations of parabolic flight.

4. **Earth Analogues:** Earth analogue testing differs from the previously mentioned simulations since it does not simulate the dynamical environment of space. Accessible areas on Earth provide characteristics analogous to areas of interest on exploration targets such as the moon or Mars, and therefore are ideal locations to conduct operations testing, equipment evaluations and training [56, 61]. Earth analogues allow the simulation of entire planetary missions, and are especially useful and realistic when combined with the use of high fidelity functional mockups. Unfortunately, due to the scale of these simulations and the fact that they tend to be performed in remote regions of Earth, they tend to be very expensive and hard to organize, manage and conduct.
5. **Mockups:** Mockups are usually used to inform the iterative design process of a manned systems. Mockups allow the designers to experience the system during the design phase, and also to obtain fundamental feedback on their choices by enabling

simulated operations. [3, 50, 53] In return, they drive the development of the system to a higher standard. Mockups can be of various fidelity levels, and are also heavily used for requirements verification and manned systems integration.

In addition to the list above, other simulation approaches, such as hybrids of the listed methods, can be used to increase overall simulation fidelity. From the breadth of the approaches listed, it should be clear that there is no single ideal simulation approach; in general, the best approach will generally include a combination of different simulation media to verify any findings. As always, real flight experience will always be the best simulation, and return the best and most reliable data.

1.4.6 MEMS Sensors

Micro-electro-mechanical systems (MEMS) sensors usually include components sized between 1 – 100 micro-meters. They usually consist of a central unit that processes data (microprocessor) and several components that interact with the physical world. At these size scales, the standard constructs of classical physics do not always apply. The large surface area to volume ratio of MEMS devices makes surface effects such as electrostatics and wetting dominate over volumetric effects such as inertia or thermal mass, leading to a substantial gain in sensor sensitivity as compared with macroscopic sensors.

MEMS were first developed in the late 60s; once the fabrication technologies matured, the technology spread considerably, opening new frontiers. Common fabrication technologies include molding and plating, wet etching (KOH, TMAH) and dry etching (RIE and DRIE), electro discharge machining (EDM), etc. In this work we will refer to three specific sensors categories; accelerometers, gyroscopes and magnetometers. Those three sensors when used synchronously define what is called an inertial measurement unit

(IMU). IMUs allow us to measure the attitude of the sensor cluster in the Earth’s magnetic reference frame. In the following section, we will expand on the details of each sensor technology.

1.4.6.1 Accelerometers

A single axis accelerometer consists of a mass, suspended by a spring in a housing as shown in figure 1.4.6.1.

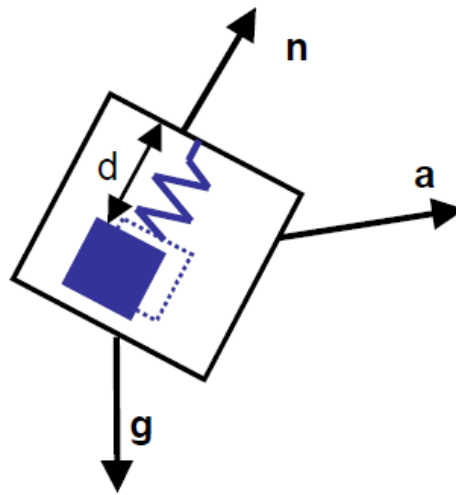


Figure 1.9: MEMS single axis accelerometer working principle, [42]

The mass is allowed to move in one direction, which is the sensitive direction of the accelerometer. The displacement of the mass is a measure of the difference of acceleration (\vec{a}) and gravity (\vec{g}) along the sensitive axis given by the unit vector \hat{n} . The electrical signal to be measured $s_{A,n}$ is related to these physical quantities according to:

$$s_{A,n} = k_{A,n}(\vec{a} - \vec{g})\hat{n} + o_{A,n} \quad (1.1)$$

with $k_{A,n}$ representing the scaling factor, and $o_{A,n}$ the offset. By mounting three such single axis accelerometers together, a 3 axis accelerometer can be constructed; the output vector ${}^S\vec{y}_A$ can thus be related to the original acceleration and gravity as follows:

$${}^S\vec{y}_A = {}^S\vec{a} - {}^S\vec{g} \quad (1.2)$$

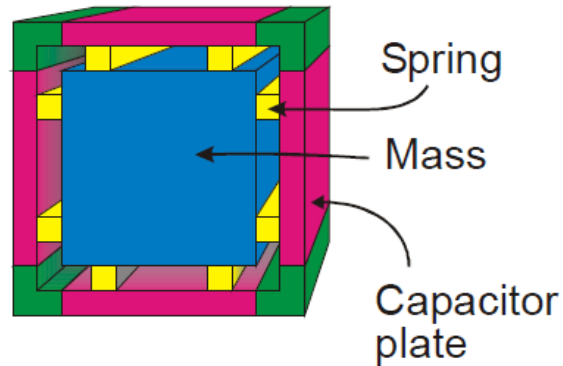


Figure 1.10: MEMS 3 axis accelerometer sample schematic, [42]

Instead of assembling a tri-axial accelerometer from single or dual axis accelerometers, it can also be constructed using a single fiducial mass. A 3D accelerometer, based on only one mass with three translational degrees of freedom, relies on a capacitive distance measurement in three directions to measure the displacement of a cubic mass, suspended in its housing by springs. The measured displacement can be related to the difference between acceleration and gravity in the same way as in the case of a single axis accelerometer.

1.4.6.2 Magnetometers

Magnetometers are sensors for magnetic field detection. In this particular case, we will focus on the Earth's magnetic field in order to obtain a measure of direction in

the Earth magnetic reference frame. Currently, the most popular principles in MEMS magnetometers are the Hall Effect, magneto-resistance and the fluxgate effect. The magnetometers used in this work rely on the torsional resonant magnetometer based on the Lorentz Force principle, and are explained as follows.

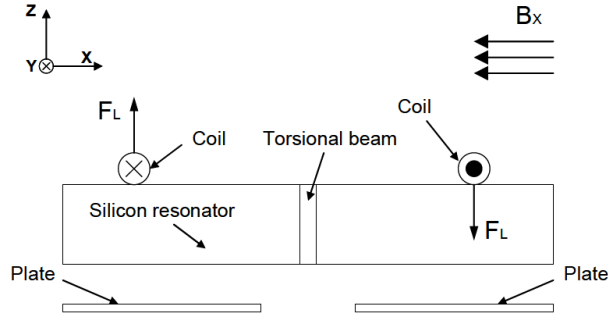


Figure 1.11: MEMS magnetometer working principle, [92]

The operating principle is illustrated in the figure above. If a DC current I is introduced into the excitation coil and a magnetic flux-density B_x in the x -direction is present, for one turn of the excitation coil with length L_c in y -direction, the Lorentz forces perpendicular to the silicon plane are:

$$F_L = IL_c B_x \quad (1.3)$$

The Lorentz forces are generated on both sides of the resonator with opposite directions. Therefore, the torques caused by each component of the Lorentz force are in the same direction, which twists the torsional beams. If a sinusoidal current passes through the excitation coil instead of a DC current, the silicon resonator will vibrate around the torsional beams due to the alternating directions of the Lorentz force F_L in:

$$i = I_0 \sin(2\pi ft) \quad (1.4)$$

$$F_L = iL_c B_x = I_0 L_c B_x \sin(2\pi ft)$$

When the frequency f of the current is equal to the resonant frequency of the silicon resonator, the vibration amplitude will dramatically increase. This vibration amplitude will then be converted into the differential change of the two sensing capacitances as shown. A capacitance detection circuit could be used to measure the capacitance change that reflects the value of the magnetic flux-density. For the Lorentz Force Magnetometer, the input is the magnetic flux-density B_x in x-direction and the output is the capacitance change ΔC . The transfer function can be expressed as:

$$\Delta C = S_M S_\psi S_C B_x = \frac{\partial M}{\partial B} \frac{\partial \psi}{\partial M} \frac{\partial \Delta C}{\partial \psi} B_x \quad (1.5)$$

Here, S_M , S_ψ and S_C respectively stand for the transfer function from the magnetic flux-density B to the excitation torque M , from the excitation torque M to the torsional angle ψ , and from the torsional angle ψ to the capacitance change ΔC . When the structural parameters of the resonant magnetometer are determined, it can be shown that S_M , S_ψ and S_C are approximately constant under a small deflection condition. Therefore, the capacitance change ΔC has a linear relationship with the magnetic flux-density B_x . The output signal of the capacitance detection circuit can accurately represent the magnetic flux-density B_x in the x-direction when the torsional angle of the resonator is in an appropriate range[92].

1.4.6.3 Gyroscopes

Angular rate sensors (traditionally referred to as “gyroscopes”) are based on a number of different designs, such as spinning rotor gyroscopes, laser gyroscopes, and vibrating

mass gyroscopes. The conventional spinning rotor gyroscope and laser gyroscopes are mainly used for navigational purposes. They are not suitable for human motion analysis because they are both expensive and bulky [42]. The vibrating mass gyroscopes are small, inexpensive and have low power requirements, making them ideal for human movement analysis. Many different geometries are used, but each one is based on the principle of a vibrating mass undergoing an additional vibration caused by Coriolis force. It consists of a mass that is actuated in the direction given by r_{act} , often using a piezoelectric element. The displacement of the mass is measured in the direction perpendicular to the actuation direction. If the housing is rotated with an angular velocity perpendicular to the plane, the mass will experience an apparent force (Coriolis force) in the direction perpendicular to the angular velocity and momentary mass speed. This force is only apparent in the sensor coordinate system, not in the inertial coordinate system. The magnitude of the Coriolis force f_C is given by:

$$f_C = 2mv\omega \tag{1.6}$$

where m is the mass, v is the instantaneous mass speed, and ω the angular velocity. Thus the displacement caused by the Coriolis force is proportional to the angular velocity, and is used therefore as a measure of angular velocity.

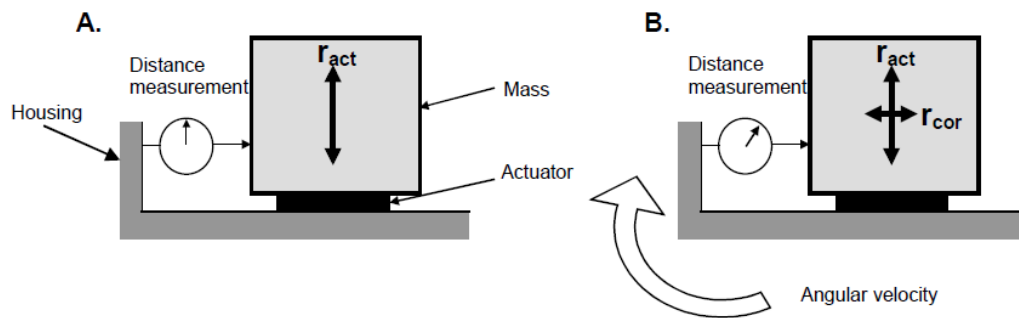


Figure 1.12: MEMS gyroscope working principle, [42]

Like the 3D accelerometer setup, a 3D gyroscope can be assembled using three single axis gyroscopes. The output of the calibrated 3D gyroscope system is the angular velocity vector, expressed in the coordinate frame of the sensor housing as in the single axis case.

1.4.7 The human system and pressure suits

Pressure suits are designed around the human wearer; therefore, understanding the kinematics of the human system is of crucial importance. The pressure garment is required to maintain both the breathing atmosphere for the wearer and sufficient pressure to maintain homeostasis throughout the body. At the same time, it must be articulated with as close to the same kinematics as the wearer as possible, at low enough resistance to allow the subject to perform dexterous activities in the space environment.

The human skeletal system serves as a framework for the body and serves the following functions: support, protection, and articulation. It consists of 206 individual bones joined by cartilage, ligaments and tendons, and its configuration creates the kinematic structure for body motion. Figure 1-13 shows the main bones in the human skeletal system. For pressure suit designers, attempting to enable all the human joints in a suit would be a daunting and impractical task. Not all the joints in skeletal system contribute in the same way to the overall dexterous capabilities of the human; therefore, a reduced set of joints is used as a reference. A simplified model of the skeletal system is going to be used in this work to address the mobility of the user. In order to quantify mobility, a standard set of primitive motions is defined in figures 1-14 and 1-15. The basics mathematical nomenclature and techniques are summarized in Appendix A.

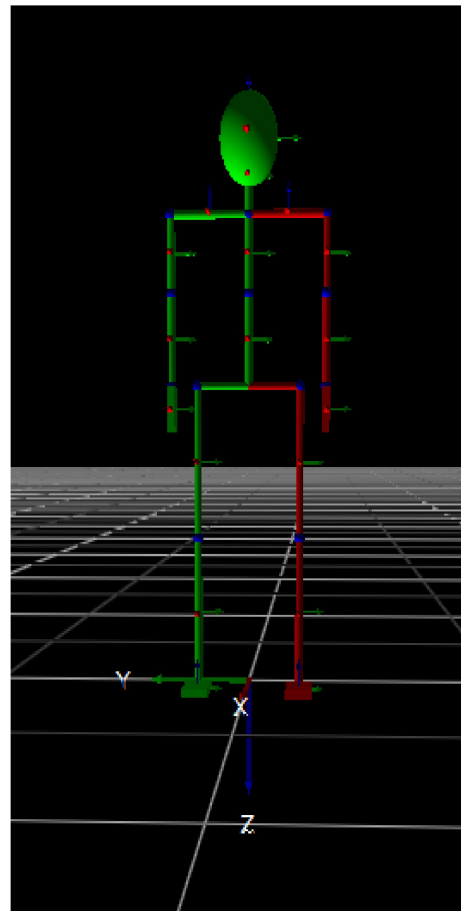
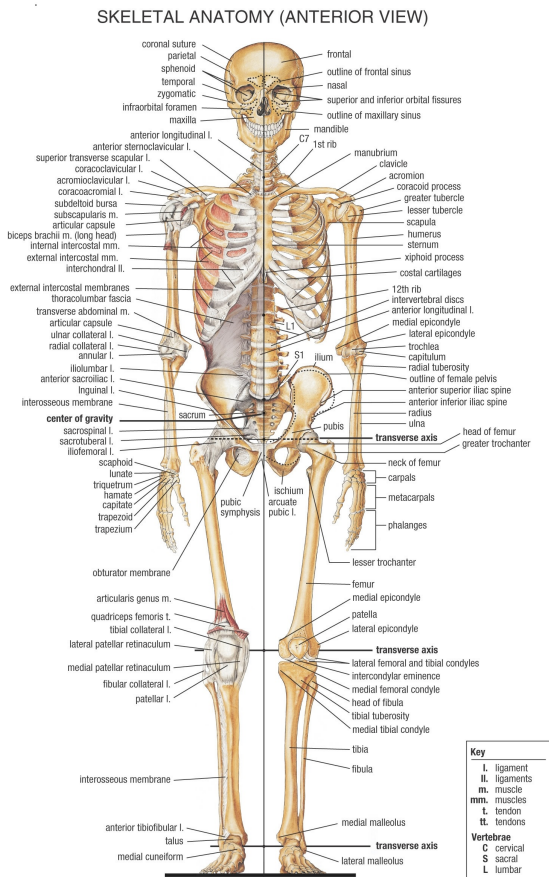


Figure 1.13: Human skeletal system (left) and simplified model (right),
 source:<http://paulkroththerapy.com>

Joint Motion	Illustration	Dynamic Segment	Static Segment
Shoulder flexion/extension		Upper arm	Torso
Shoulder adduction/abduction		Upper arm	Torso
Shoulder lateral/medial rotation		Upper arm	Torso
Elbow flexion/extension		Forearm	Upper arm
Hip flexion/extension		Hip	Torso
Hip adduction/abduction		Hip	Torso

Figure 1.14: Human motion standard nomenclature [21]

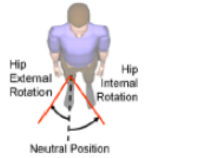
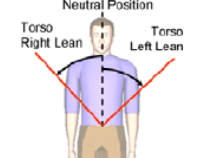
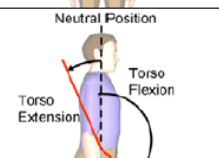
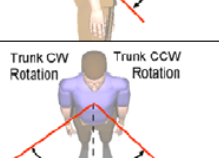
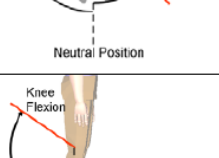
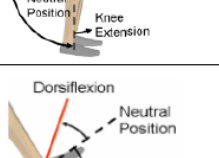
Hip (ankle) rotation		Hip	Torso
Torso lean		Torso	Hip
Torso flexion/extension		Torso	Hip
Torso rotation		Torso	Hip
Knee flexion/extension		Shin	Hip
Ankle flexion/extension		Foot	Shin

Figure 1.15: Human motion standard nomenclature (continued) [21]

Chapter 2

BPMS Design, Implementation, and Performance Assessment

The Body Pose Measurement System (BPMS) is an embedded system designed to measure the orientation of the major limb segments of a human test subject. The concept was originated by the desire to measure human body motion in an affordable and unconstrained fashion, where the investigators would be able to record and later analyze the data that was acquired, without requiring an extensive equipment overhead and without constraining the measurements to a laboratory setup. The system relies on a network of MEMS (Micro Electro Mechanical Systems) IMUs (Inertial Measurement Units) and microcontrollers that allows the accurate identification of the attitude of the sensors in the Earth magnetic reference frame. This chapter will guide the reader through the design, development and validation of the system. All phases will be described in full detail, and hopefully will serve as a template for future development and diagnostics of the platform.

BPMS is the result of several iterations and extensive experience with optical motion capture systems. By using such systems, several limitations were immediately recognized. In order to fully understand the system limitations, one must understand its general working principle. First of all, optical motion capture relies on a network of cameras (from a minimum of 3 cameras to a general standard of 12 to 14 but more can be used).

The camera network is appropriately oriented to "look" at a volume of interest and then calibrated. Once the camera positions have been recognized and the ground plane is defined, the system is ready to acquire marker positions. Optical motion capture relies

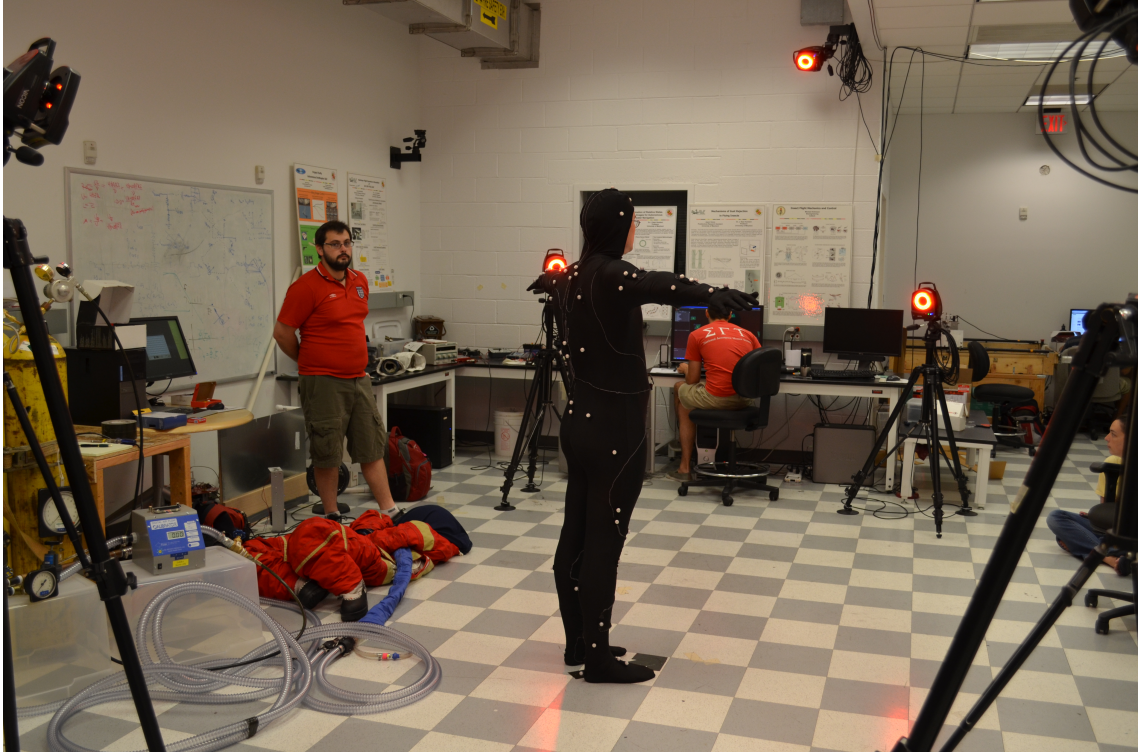


Figure 2.1: Vicon motion capture setup in the AVL laboratory

on retro-reflective markers (usually in the shape of spheres, but other shapes can be used based on the specific application needs). Each camera usually includes a light source (in the visual or infra-red range) that shines a pulsing light that is then reflected from the markers and captured by the cameras. When more than three cameras detect a reflection, based on the known positions and orientations of the cameras, the marker position is identified by triangulation. From this initial description of the optical motion capture algorithm, one can immediately see three limitations: first, the system will not be able to recognize the position of a marker that it cannot see (occlusion) with at least 3 cameras; second, the volume of interest is intrinsically defined by both the number of cameras and the required accuracy (the sparser the cameras, the less accurate the system becomes);

and lastly, the system relies on a known frequency of light source, which restricts the use of such systems to a very controlled indoor laboratory setup. While those limitations are usually considered negligible for most applications, they become disabling or very hard to work around when one's interest is focused on studying space suits and space suit operations.



Figure 2.2: BPMS during the 2012 desert FLEAS trials

Since this work focuses specifically on those applications, a new strategy was required. BPMS is the SSL's initial attempt to a novel and much more practical human based motion capture system. Since MEMS sensors have become both widely used and accessible, they have enabled a series of applications that were up to now conceptually valid but impractical in their implementation. Attitude estimation is nothing new; the

aviation industry has made use of IMUs constantly in the past decades, and the only major issue for its transition to a portable system such as BPMS was due to their mass and size. MEMS sensors today allow for analogous attitude estimation systems with similar sensitivity and resolution, but in a much smaller package.

2.1 BPMS Garment

BPMS is composed of a garment that serves as a mounting point for the core electronics components and cables. This section will describe its selection and outfitting with the instrumentation hardware.

2.1.1 Garment Selection and Sizing

When designing instrumentation systems for human use, sizing is of great concern. Making a single system that can be used by a large array of users usually leads to a system that can be used badly by many and well by few. A common solution to the problem is to develop the system in multiple sizes, but this solution is usually adopted once the system leaves the prototype stage. In this particular case we are dealing with a prototype system, and therefore a single size was used. The garment was envisioned mainly for the 50th percentile American male (size L).

In order to allow for a wider subject pool, material choice was the next problem to solve. A great item of concern in the biomechanical field is to ensure that sensors conform with the body. Any relative motion between the sensors and the body would lead to measurement errors, and therefore limit the systems effectiveness. The garment material should conform to the human body and maintain enough pressure to restrain the sensors, but not so much as to cause blood pooling, discomfort, or restricted mobility, all while

still encompassing a wide array of subject sizes. An very elastic material was needed, and Spandex met all the requirements. Spandex is a very elastic synthetic polymer-based fabric that can be stretched to up to 600% of its initial length and return to its original form without losing its integrity. As one can imagine, those properties degrade over time and over several cycles of stretching and release, but the degradation period is well beyond the expected BPMS system useful lifetime. Also Spandex, due to its polyurethane base, reduces the chances of allergic reactions.

The Spandex garment chosen for the BPMS prototype is a commercially obtained single piece, large size, full body suit with built in feet, gloves, and cap, with a back zipper for donning and doffing. The garment itself is very light and breathable, and was able to fit a large pool of test subjects ranging from a 6.5 ft tall athletic male to a 5.2 ft tall slim female. All subjects reported no apparent reduction in range of motion or any discomfort. The only issue recorded and mitigated was with the smaller subjects, for which the sensors required elastic straps to remain conformal to the body. This issue was mostly present in the forearm, arm and hip sections, but was never an issue in other portions of the body.

2.1.2 Sensor Placement and Cable Routing

As previously stated, the BPMS garment is the main building block for sensor placement and cable routing. The first item of concern when assembling the systems was where to place the sensors and how to route the cables. Since the main objective of the system is to measure the attitude of the major long bones on a wide array of subjects, the sensors had to be placed somewhere in the mean center of the those bones so that by adjusting the fit of the suit on the various subjects, the sensors would still fall on the desired bone. The system includes 18 sensors located on the on the following body

segments: Head, neck, hands, fore-arms, arms, shoulder blades, feet, legs, thighs, hip and mid-spine. In order to appropriately place the sensors on the garment, three representative test subjects were asked to wear the garment while the desired sensor locations were marked. Later the marked locations were compared and the the mean position between the three markings was identified. In order to verify if the identified mean position was adequate, the subjects were asked to don the garment again and in all cases the final sensor positions were considered suitable.



Figure 2.3: BPMS sensor locations and cables routing

The second step then was to define the cable routes. Although elastic electrical

cables do exist in commerce, they were not used in this particular application, due to their electrical properties and their incompatibility with the current electrical design. A more in-depth explanation of the rationale behind this choice is given later in the electronics design section. Standard shielded cables were required, and therefore a careful routing plan was necessary. The BPMS Spandex garment is very elastic, and routing inelastic elements on it can be quite a challenging feat. To aid the decision making process, the concept of non-extension lines came to aid. In 1970 Iberall [4] described the concept of lines of non-extension as the curves on the surface of the body along which the skin does not stretch during body movement, therefore by placing the inelastic wires on those lines, it could be possible to include the wiring without inducing any discomfort of mobility constraints.

Unfortunately though those lines of non-extension vary between subjects, and do not necessarily allow for connection routing between the sensors. An alternative approach was therefore required. By observing how subjects moved in the garment, it was apparent that large deformations were not present (except during donning and doffing), and that possibly a series of serpentine paths would allow for both on-and off-axis elongations. Electrical requirements on cable lengths defined the number of possible turns that could be done in each serpentine, and the cable stiffness defined the maximum practical radius of curvature. The final step was integrating the whole electronics assembly on the garment by sewing the cables and sensors on it. This phase was entirely done by hand, and was aided by stuffing the garment with polyester batting so that it assumed a mannequin like form, which allowed for sewing without risking sewing both sides of the garment together. This also made the cables' serpentine routes very easy to trace and sew. A lesson learned during this phase was that the sewing pitch needed to be very small. Initially a 1 cm

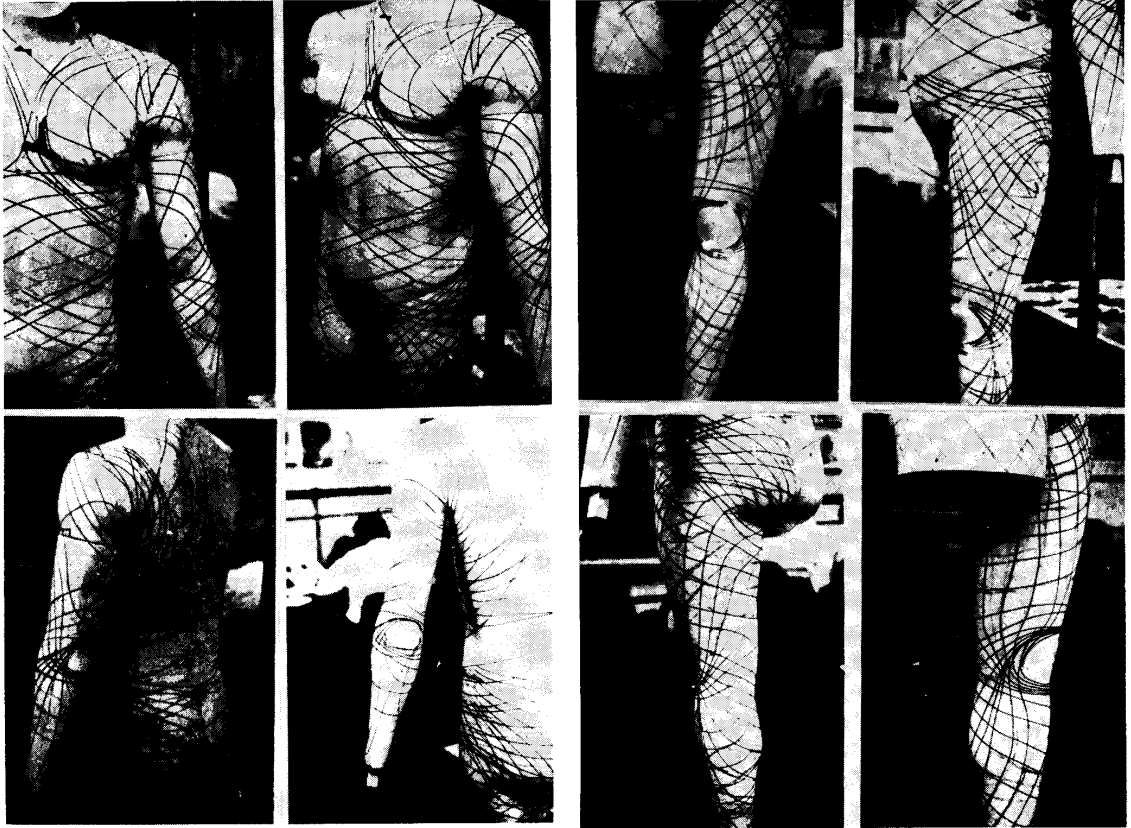


Figure 2.4: Lines on non extension on the human body [4]

stitch pitch was chosen; as soon as the inner batting was removed from the garment, the cables tended to bunch up and create a series of tension points on the garment. Later, the stitch pitch was reduced to approximately 1mm, and the issue was resolved.

2.2 Electronics

This section will describe the BPMS electronics architecture and the various lessons learned during the four main iterations that lead to a working prototype. All the electronics schematics, PCB design, and routing was accomplished by using CadSoft's EAGLE V6 (Easily Applicable Graphical Layout Editor) CAD package along with several open-source

(sparkfun.com) and custom part libraries. Additionally, the systems makes extensive use of and reference to the Arduino IDE and the relative hardware architecture.

2.2.1 System Definition and Requirements

In 2009 the SSL was awarded the NASA LASER (Lunar Advanced Science and Exploration Research) four-year grant NNX11AB42G focused on the investigation of advanced human -obot interaction to aid future lunar geological exploration. The SSL joined forces with ASU (Arizona State University) to accomplish this goal. Yearly field trials were planned to execute extended sortie simulations with simulated suits and astronaut-assistant rovers. As of the writing of this work, a preliminary and two official desert field trials have been executed, which are documented in the following publications [61, 29, 63]. The initial focus of the work for the LASER grant was to investigate advanced controls and displays for future EVA operations. As part of this research, a novel gestural interface was implemented and tested. A single instrumented arm equipped with 3 IMUs and 10 piezoelectric bend sensors was designed and built, and used in the field to teleoperate RAVEN (Robotic Assist Vehicle for Extravehicular Navigation) [63].

This system is capable of visualizing the pose of the user's hand, including finger bend, with surprising accuracy. This instrumented arm was the progenitor of BPMS, since it inspired the conception of an instrumented garment capable measuring the pose of a test subject inside a pressure suit or in shirt-sleeves. The first questions that arose were whether we could extend the instrumented arm concept to the whole human body by selecting a series of significant body segments to monitor, record and/or stream their pose in space. It was immediately noted that the challenge would lie in two main fields: electronics development and filtering schemes.



Figure 2.5: Instrumented arm during the 2011 Desert FLEAS trials

The first step was to define requirements that were necessary to drive further design and development of the system. BPMS had to be able to measure, accurately, the pose of a human test subject both inside and outside a pressure suit, while not being limited to pressure suit applications. It also had to be low cost, and minimize as much as possible the requirement for overhead equipment to monitor, record and visualize the acquired data. BPMS had to be usable in outdoor environments, and must be able to accommodate a large selection of test subjects. It must not constrain the subject's mobility, and must

provide data at significant sampling rates for human kinematic analysis.

Based on the initial observations made on the instrumented arm prototype, it was decided not to include bend sensors for the fingers, but focus exclusively on the major long bones of the body. Possible future implementations might bring back this functionality, but it was deemed unnecessary to prove the concept. As previously mentioned, the electronics development process had to be addressed first, and the above requirements quantified as much as possible.

In order to benefit from prior experience (instrumented arm), it was initially decided to group IMUs in clusters for two reasons: to reuse as much as possible of the available architecture of the instrumented arm, and to enable a parallel architecture that would allow for faster sampling rates. Sampling rate of the system is a very important parameter, since it will later drive the design and tuning of the filter scheme, but also from the requirements above it must be compatible with human kinematic measurements. From the available literature, it was concluded that a sampling rate of 50Hz would be more than sufficient for this purpose; therefore, it was chosen as the reference rate for BPMS. In order to achieve such a sampling rate, and given the performance for the communication systems envisioned for this application, it was soon evident that a single stream system would not be sufficient. Clusters of varying size were tested by connecting an increasing number of IMUs to a single cluster board. The cluster board was required to read and forward the incoming information. Since BPMS was envisioned to read from 18 IMUs, we had to scale the allowed time for this operation based on the number of IMUs on each cluster. Through various iterations it was noted that, in order to achieve 18 IMU measurements in 20ms (50Hz), clusters of three IMUs were necessary. This was a result of the time required by the cluster board to switch the multiplexer channels, request data

from the sensors, and forward it over the common serial link. Adding more IMUs to a cluster would increase the time due to multiple multiplexer switches, while reducing the number of sensors per cluster would possibly enable higher refresh rate, at the price of increased system complexity and cost. The three IMU cluster architecture was chosen to minimize the number of microcontrollers in the system and therefore power draw, but this eliminated the possibility of on-board filtering. Unfortunately, though, the clustered architecture adopted requires an arbiter that fuses the data streams together and allows a single communication channel to an external logging system. To do so, an additional microcontroller was required, bringing the total on BPMS to seven.

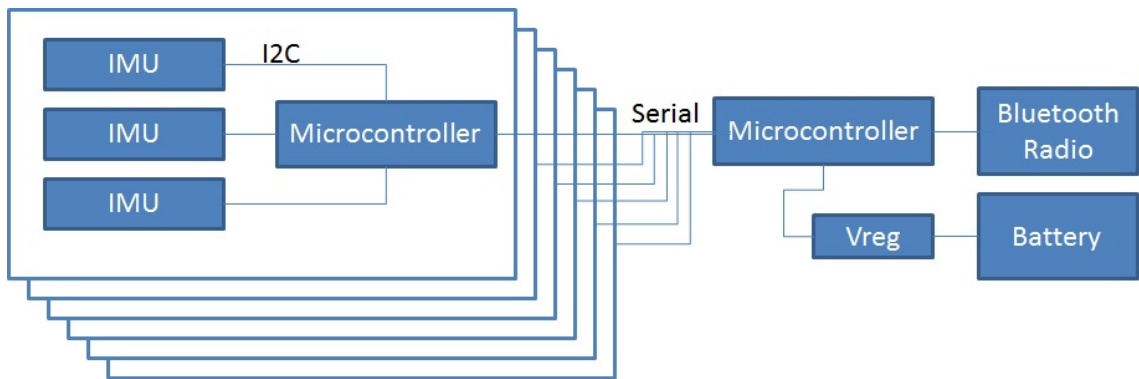


Figure 2.6: BPMS Electronics Architecture

Once the number of microcontrollers and their functions were defined, it was time for sensor selection and wireless interface definition. The following sections cover those two phases of the design process in detail.

2.2.2 Sensor Selection

The IMU sensor arrays are the core of the BPMS system. From the initial requirements, miniaturization and accuracy of the overall system were key parameters to meet. Measuring attitude is by no means a new or simple feat. There have been many, very accurate, methods to achieve such task in the past but, with the absence of MEMS technology, those were confined to large and expensive systems. MEMS technology brought to the masses the ability to sense physical parameters with great accuracy and sensitivity at an affordable price and very small packaging. For the BPMS sensor selection, the search was confined to MEMS sensors. As previously mentioned in Chapter 1, there are several ways to estimate attitude, for each method, a set of sensors is required and also a series of limitations are imposed. For BPMS it was chosen to use the complete set of sensors (accelerometer, magnetometer and gyroscope) to eliminate induced drift and increase measurement accuracy therefore reducing the system limitations at the cost of a slightly more complex system and orientation filter. In addition to the above, having the complete array of sensors doesn't preclude the implementation of simpler filters and algorithms. As far as electronics design and manufacturing goes, MEMS sensors are incredibly hard to design, manufacture and mount on PCB boards, and since it was beyond the scope of this work to design and build such systems, the commercial route was chosen. The possible options were to acquire the bare sensors, design the required support electronics and PCBs and finally manufacture and populate them. Unfortunately with inertial sensors, even a small sensor misalignment can be very detrimental to the orientation estimation filters, therefore, since the initial stages of the selection process, it was considered a requirement to acquire robotically pre-assembled sensor boards. Such assemblies guarantee high precision in the sensor placement and also reduce the complexity of populating boards with very

small pitch surface mount components with hidden pads. Given this decision the search began. Several options are commercially available and those options increase every day. The most advanced and affordable sensor array found at the time of the sensor selection process was the sparkfun SEN-10724 IMU sensor stick.

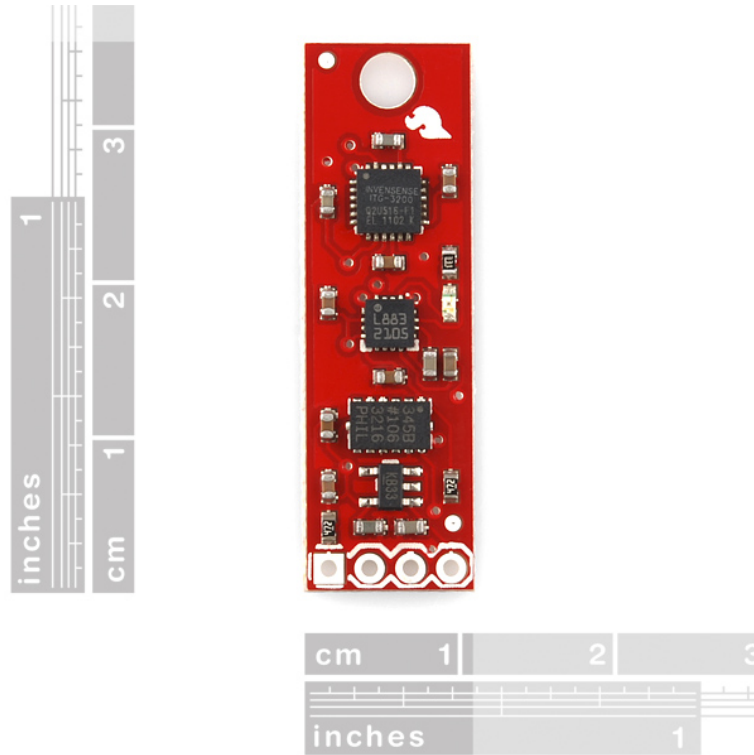


Figure 2.7: Sparkfun SEN-10724 IMU sensor stick. Source:Sparkfun.com

The SEN-10724 is a nine degrees of freedom sensor array that includes a three-axis ADXL345 accelerometer, HMC5883L magnetometer, an ITG-3200 gyroscope along with voltage regulation for all the sensors and I2C data lines. In this specific application, all the sensors are initialized to sense at their full resolution, at a frequency of 50 Hz. The actual sensor sampling rate is much faster, but internal filters and averaging logic is used to provide more accurate estimates of the readings at the given frequency. In addition to the above, the gyroscope was also configured to use its internal low pass filter which was

set to 42Hz. The following sections will describe the main characteristics of each sensor.

2.2.2.1 Accelerometer

The ADXL345 is a small, thin, low power, 3-axis accelerometer with high resolution (13-bit) measurement at up to 16 g. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I2C digital interface. The ADXL345 is well suited for mobile device applications. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (4 mg/LSB) enables measurement of inclination changes less than 1.0. Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion and if the acceleration on any axis exceeds a user-set level. Tap sensing detects single and double taps. Free-fall sensing detects if the device is falling. These functions can be mapped to one of two interrupt output pins. An integrated, patent pending 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor intervention. Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation. The ADXL345 is supplied in a small, thin, 3 mm × 5 mm × 1 mm, 14-lead, plastic package. Additional information on the sensor is available in the following datasheet [94].

2.2.2.2 Magnetometer

The Honeywell HMC5883L is a surface-mount, multi-chip module designed for low-field magnetic sensing with a digital interface for applications such as low-cost compassing and magnetometry. The HMC5883L includes, high-resolution HMC118X series

magneto-resistive sensors plus an ASIC containing amplification, automatic degaussing strap drivers, offset cancellation, and a 12-bit ADC that enables 1 to 2 compass heading accuracy. The I2C serial bus allows for easy interface. The HMC5883L is a 3.0x3.0x0.9mm surface mount 16-pin leadless chip carrier (LCC). Applications for the HMC5883L include Mobile Phones, Netbooks, Consumer Electronics, Auto Navigation Systems, and Personal Navigation Devices. The HMC5883L utilizes Honeywells Anisotropic Magnetoresistive (AMR) technology. These anisotropic, directional sensors feature precision in-axis sensitivity and linearity. These sensors solid-state construction with very low cross-axis sensitivity is designed to measure both the direction and the magnitude of Earths magnetic fields, from milli-gauss to 8 gauss. Additional information on the sensor is available in the following datasheet [95].

2.2.2.3 Gyroscope

The ITG-3200 features three 16-bit analog-to-digital converters (ADCs) for digitizing the gyro outputs, a user-selectable internal low-pass filter bandwidth, and a Fast-Mode I2C (400kHz) interface. Additional features include an embedded temperature sensor and a 2% accurate internal oscillator. The ITG-3200 triple-axis MEMS gyroscope includes a wide range of features such as: Digital-output X-, Y-, and Z-Axis angular rate sensors (gyros) on one integrated circuit with a sensitivity of 14.375 LSBs per /sec and a full-scale range of 2000/sec., three integrated 16-bit ADCs provide simultaneous sampling of gyros while requiring no external multiplexer. Enhanced bias and sensitivity temperature stability reduces the need for user calibration and reduced low frequency noise simplifies application development and allows for more-responsive motion processing. Digitally-programmable low-pass filter, low 6.5mA operating current consumption for long battery

life and wide VDD supply voltage range of 2.1V to 3.6V. Flexible VLOGIC reference voltage allows for I2C interface voltages from 1.71V to VDD, a standby current of 5A all in a small and thin package for portable devices (4x4x0.9mm QFN) that doesn't require a high pass filter. The ITG-3200 has a turn on time of 50ms, a digital-output temperature sensor and is factory calibrated (scale factor). It is also 10,000 g shock tolerant and allows for fast Mode I2C (400kHz) serial interface. The on-chip timing generator clock frequency is accurate to +/-2% over full temperature range while optional external clock inputs of 32.768kHz or 19.2MHz can be used to synchronize with the system clock. Finally, the MEMS structure is hermetically sealed and bonded at wafer level and is RoHS and Green compliant. Additional information on the sensor is available in the following datasheet [96].

2.2.3 Wireless interface

Being limited by connector cables, or an umbilical was since the beginning deemed unacceptable. BPMS had to be either a standalone or a wireless system that didn't constrain the mobility of a test subject. Wireless capability is extremely appealing especially since it allows to monitor in semi-realtime the status of the system and it also allows to observe the acquired data as it is collected but it can lead to several potential issues. Wireless capabilities require both a transmitter and a compatible receiver radio and therefore it adds an additional level of complexity. Also, wireless communications can be plagued by interference and data corruption. On the other side a local recording system is much more simple and reliable alternative. The decision was made to implement a wireless solution while still retaining the ability to add a data logger later on by allowing for a breakout of the outbound serial link. The first step in the definition of the wireless interface was

to define the data rates involved in the transmission. Also one must keep in mind that in order to command the system, a full duplex (2 channel radio for transmitting and receiving at the same time) radio is required. Once the outbound communication protocol was identified (Serial TTL Level protocol), to evaluate the required data rates, the following steps were taken. First from the given requirements, a sampling rate of $f = 50Hz$ was required. Each sensor board, return two bytes of data per axis per sensor, therefore for each measurement, a total of 18 bytes per IMU are returned. In addition to the sensor data, each IMU cluster returns two error bytes that encodes the validity of each of the measurements on that cluster. Further details on the error bytes will be given in the software section. Since each cluster muxes three IMUs, a total of 56 bytes per measurement are returned (54 data bytes followed by 2 error bytes). Finally given that there are 6 IMU clusters in BPMS, we are left with a grand total of 336 bytes per measurement. Since each byte set is position encoded (Given the position of the byte in the transmission it assumes a specific meaning. i.e the first byte is the Most Significant Byte (MSB) of x axis of the accelerometer of the first IMU in the first IMU cluster, and so on), a series of beginning and end of sentence characters are required. For this application, the beginning of sentence characters are "\$#" and the end of sentence are the same but reversed "#\$". This strategy allows for minimal overhead and diagnostics data transmitted and results in a total of 340 bytes per measurement. The last piece of the puzzle is evaluating the minimum required data rate. In order to evaluate it, we can use the following relation:

$$bps = 8fBpp \tag{2.1}$$

where bps stands for bits per second, f is the sampling rate and Bpp are the bytes per packet. In the equation above we pre multiply by a factor of 8 due to the byte-Bit relation

(1 Byte = 8 Bits) and obtain:

$$bps = 8fBpp = 8 * 50 * 340 = 136Kbps \quad (2.2)$$

From the simple relation above, we can see that the data rate is quite large especially when compared to standard baud rates for serial wireless interfaces which is usually in the range of 9.6 to 57.6 Kbps. Another thing to keep in mind is that with increased data rate, power draw increases and transmission range decreases, also a higher likelihood for packet loss is inherent therefore transmission reliability is reduced. In order to mitigate those factors and reduce unknowns, in the first BPMS prototype it was decided to use a familiar interface. The Digi XBee family of wireless radios running the ZigBee wireless protocol, were widely used in the lab and had also been implemented in several side research projects. The XBee modules are very versatile 2.4GHz full duplex radios that allow data rates up to 250 Kbps and ranges up to 3.2Km (not both at the same time and under ideal line of sight in the more powerful 63mW XBee-PRO) therefore they were the first alternative to come to mind. Initially the module used was a pair of X-Bee-Pro ZB with PCB integrated antenna [97] seemed to work as desired and data was received reliably over very close ranges (5m range). Unfortunately though as soon as the radios were set apart, missed packets or incomplete packets began to appear. The radio module in this first iteration was programmed to run at 250Kbps at maximum transmission power. Initially this behavior was attributed to wireless interference over the over-cluttered 2.4GHz band, but it was soon realized that the issues were not limited to band usage. More experiments with the system lead the following observations: The radios over a brief amount of time (usually 5-10 min) of continuous transmission, were heating up to approximately 40-50 °C, also power draw increased. Performance at short range was nominal while as temperature increased, reliability of communication over longer ranges decreased. In order to better

test the wireless communication system, a simple setup was implemented. An Arduino 2009 was used to simulate the data source (BPMS) by randomly generating data bytes and streaming the data to the X-Bee module. The board was then powered by a power supply that enabled constant regulated power output and also allowed for a first cut measurement of current draw. The use of a power supply in this case simplifies the overall system, since during the initial test, there were several suspicions that the power source (Lipo battery and a buck-boost regulator) was not enough to reliably power the radio at that high of a data rate. The last thing that was done before running additional evaluations, was to reduce as much as possible radio interference (turning off all 2.4Ghz devices in the area where the test was being conducted). The result of the evaluations was not promising. Once again as soon as the range was increased to about 10m, the signal would become unreliable and data would be lost in the transmission. A different approach was required. After a careful study of the XBee documentation [97], it was possible to narrow down the possible cause of the issue to the high data rate. A few more experiments were conducted at slower data rates (with a reduced outbound data set) and it was observed that by reducing the data rate, higher ranges were possible. This observation drove the second iteration of the design of the BPMS main board that made use of two radios instead of one. The two radios would work on two separate circuits and would transmit the data from 3 IMU clusters each at a data rate of 115Kbps. In this second prototype setup, higher ranges were possible but the maximum reliable range resulted to be approximately 20m. The range was still too low for our application and while the two radios didn't overheat any more, the power draw now doubled to approximately 600mA. The experimental results so far were far less than promising and it was soon clear that a different approach had to be taken. Unfortunately, high data rate, long range

and low power wireless interfaces are very hard to find and the only other promising candidate was Bluetooth technology. Bluetooth transmission protocols are very appealing since they guarantee complete, encrypted packet delivery at very high data rates (up to a theoretical 2Mbps with Bluetooth V2.0) This implies that we would suffer much less radio interference and still maintain a reliable connection. For as much as this sounds great, Bluetooth in general is not a long range communication protocol. Bluetooth radio range is usually limited to a few meters when a class 2 power output radio is used (common radio power output in most Bluetooth enabled devices), thankfully, class 1 radios exist and they enable transmissions in line of sight up to 150m. In the final iteration of BPMS, a Class 1 Roving Networks RN-41 Bluetooth module was implemented [98]. The bluetooth radio was then evaluated with the same experimental testbed as the XBee radios and in all evaluations it outperformed the previous setups. Data transmission was reliable up to 40m in line of sight (maximum distance tested) at 2Mbps data rate with an average power draw of approximately 60 to 80 mW. The radio doesn't heat up but it does induce a new item of concern, timing. While the ZigBee protocol doesn't guarantee packet delivery, it does guarantee it's semi-synchronous timing. XBee radios do not induce any significant time delay between when the packet is received by the radio and when it is transmitted. On the other hand, Bluetooth radios, due to encryption and data encoding for lossless transmission, induce time delay. The time delay so induced is first of all random, and it can vary between a few ms up to 300ms. While time delay for our application is not necessarily a concern, it does imply that a severe timing scheme on the embedded system is required in order for the orientation filter to properly work.

2.2.4 Power distribution

BPMS is a low power system drawing under nominal operating conditions, approximately 190mA at 3.3V (≈6W). In order to provide the required regulated power, two main choices had to be made; Power source and power regulation. The choice of the power source would drive the design of the power regulation system and define the system operational endurance. From the requirements section, we desire a system endurance of at least 4 to 7 hours. Given the above system current draw, we immediately see that we would be required a power source capable of at least 1330 mAh at 3.3V (for 7 hour endurance). The back of the hand calculation above does not take in consideration inefficiencies in the regulation system and the unavailability of all the stored power in the battery. Once the power requirements were defined, the selection process required the definition of the battery technology to be used. High energy density is a highly desirable parameter since it would imply, smaller and lighter batteries, but given the specific application of the system, another even more important item of concern is stability and safety of the power source. The system is meant to be deployed in pressure suits that can include pure O_2 environments and in general any type of hazard associated with any component of the system would automatically disqualify it from its meant purpose. Lithium Ion(Li-Ion or $LiCoO_2$) batteries possess the highest energy density but are intrinsically unsafe. These batteries can explode or catch fire when improperly used and as of now are not flight rated from manned applications. The options immediately reverted to two main types of batteries: Lithium Iron Phosphate or Nickel Metal Hydrate. Both are very safe batteries and both are compatible with space suit applications. The table 2.1 below shows a brief performance comparison between battery technologies.

From it, we can immediately see that the ideal choice lies in $LiFePO_4$ batteries for:

Table 2.1: Battery technology comparison table

Chemistry	Voltage	Energy Density	Work- ing Temp.	Cycle Life	Safety	Environ- ment	Cost based on cycle life x wh of SLA
$LiFePO_4$	3.2V	>120 wh/kg	-0-60 $^{\circ}C$	>2000	Safe	Good	0.15-0.25 lower than SLA
Lead acid	2.0V	> 35wh/kg	-20 - 40 $^{\circ}C$	>200	Safe	Not good	1
NiCd	1.2V	> 40wh/kg	-20 - 50 $^{\circ}C$	>1000	Safe	Bad	0.7
NiMH	1.2V	>80 wh/kg	-20 - 50 $^{\circ}C$	>500	Safe	Good	1.2-1.4
$LiMn_xNi_y-$ Co_zO_2	3.7V	>160 wh/kg	-20 - 40 $^{\circ}C$	>500	Unsafe without PCB or PCM, better than LiCo	OK	1.5-2.0
$LiCoO_2$	3.7V	>200 wh/kg	-20 - 60 $^{\circ}C$	> 500	UnSafe without PCB or PCM	OK	1.5-2.0

Their safety record, higher energy density, longer life and higher voltage per cell than their NiMH counterpart, not to mention lower cost. By using $LiFePO_4$, we can implement a single cell setup with a buck-boost regulator that would set the output voltage of the circuit to 3.3V and automatically shut down the system when the battery voltage drops below 2.6V to protect the battery from over-discharge. The voltage regulator adopted in this design relies on a TPS61200 IC Boost converter [99] which allows regulated 3.3V output at 200mA continuous output current draw. For this application a single cell 18650 size, rechargeable 1.5Ah $LiFePO_4$ battery was chosen. The specific battery allows for a maximum continuous discharge current of 4.5Ah and a total energy of 4.32Wh. The battery is also equipped with a small PCB that prevents over-under voltage and it also limits current draw therefore adding an additional level of safety to the unit. The battery specifications are provided in the following datasheet [100].



Figure 2.8: LiFePO4 18650 Rechargeable Cell: 3.2V 1500 mAh [100]

The battery size allows for a theoretical system endurance of 7.89h but it was experimentally tested that on average batteries tend to last for a minimum of 6.3h to a maximum of 6.5h. The reason behind the discrepancy between the theoretical and the real endurance values, lies in the inefficiency of the power regulation system (which is non linear and depends on input voltage, which is changing as the battery discharges.) as well as a non constant power draw (the system is not constantly transmitting and therefore the mean current draw is a smaller than the maximum measured).

2.2.5 System architecture

The following section will bring together all the previous portions of the electronics design and describe the final implementation of BPMS. In order to produce a usable prototype two main boards had to be designed: The sensor cluster boards and the the main arbiter. To design the physical layout and electrical schematics, CadSoft's EAGLE V6 was used. EAGLE is able to aid the design and manufacturing of custom PCB board by enabling the user to design schematics that are linked to a PCB layout editor and finally from those produce CAM files that can be sent to a manufacturer for production. BPMS as previously introduced is composed by 18 IMUs each containing three sensors; An accelerometer, a gyroscope and a magnetometer. Each of those sensors communicates with a cluster board where an AVR ATmega328P [101] initializes and queries the sensors over I2C. Since the sensors on each IMU have a set address, a multiplexer was required to acquire data from multiple IMUs through a single I2C line. Once the data and error flags have been acquired by the sensor cluster board, the AVR waits for a query from the main arbiter. The main arbiter is again an AVR microcontroller that in sequence queries all the sensor cluster boards via a high speed serial channel. The serial channel makes use of a multiplexer to direct the queries to the right sensor boards and opens the transmission to the bluetooth radio. Below are the schematics and layouts of the various BPMS PCB boards.

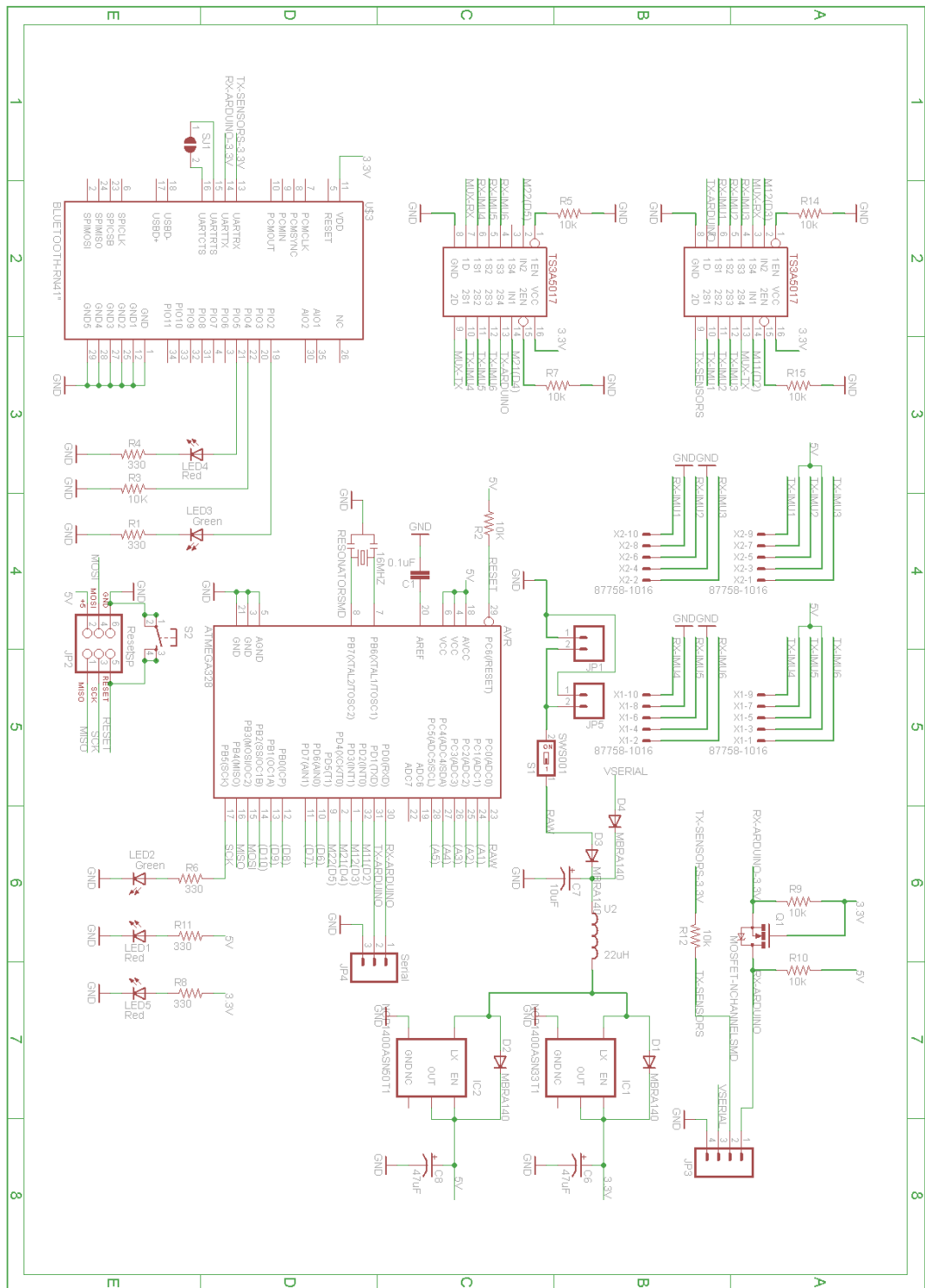


Figure 2.9: BPMS final iteration main board schematic

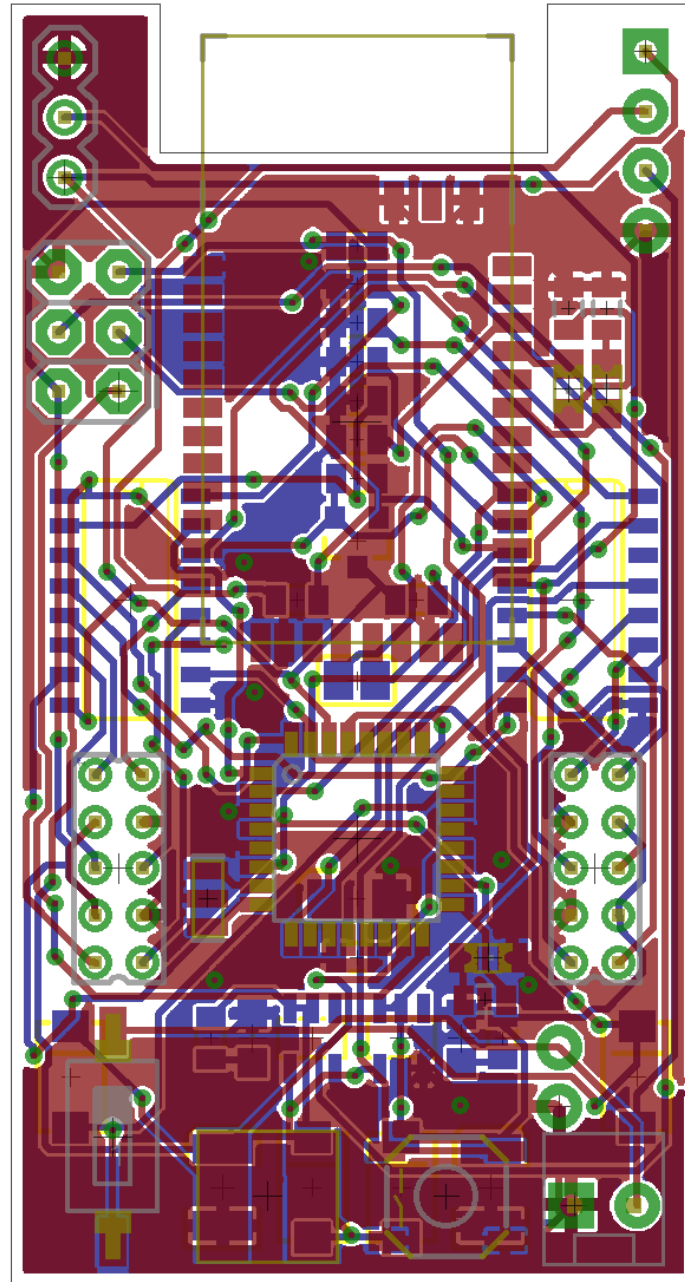
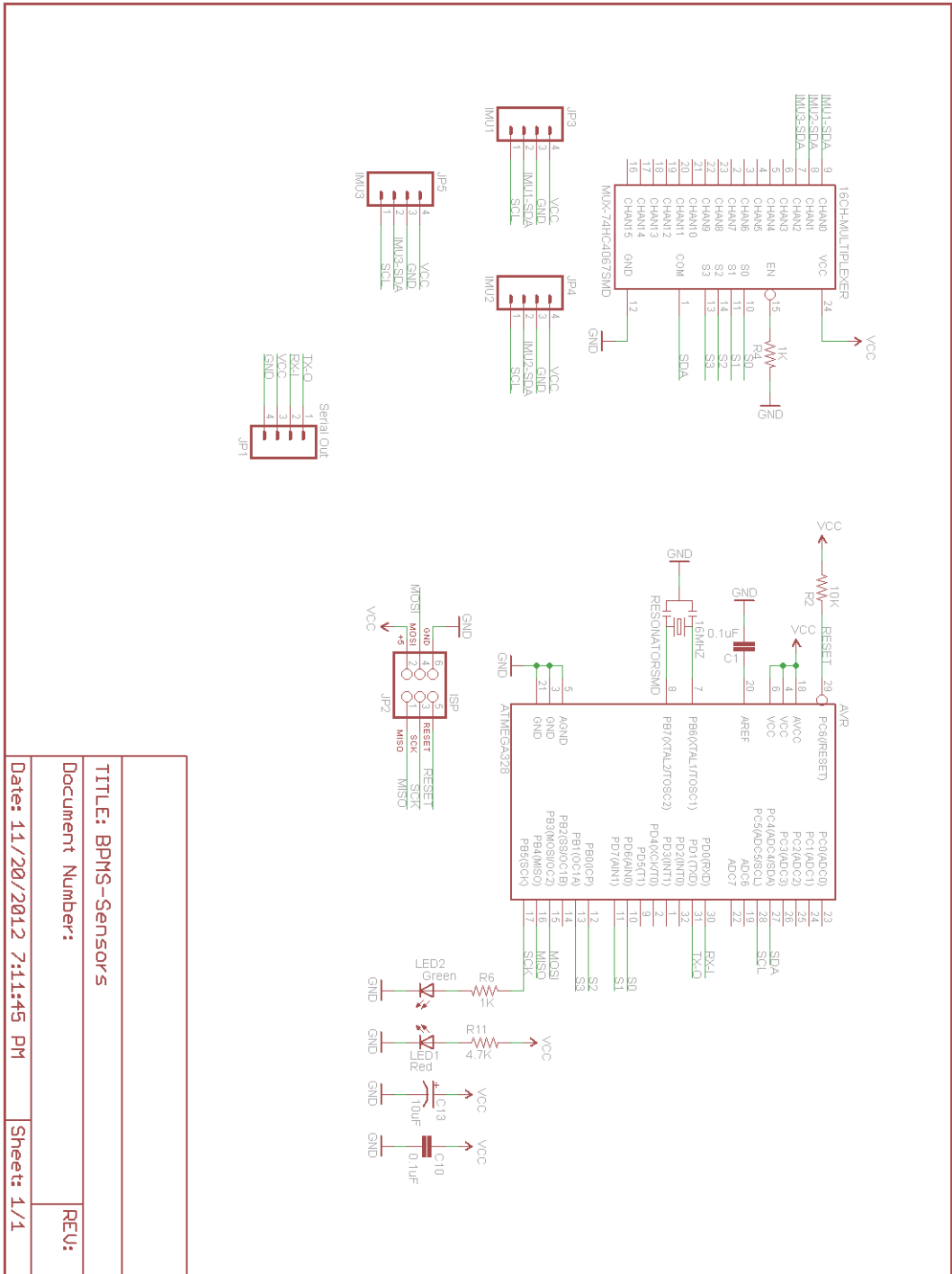


Figure 2.10: BPMS final iteration main board layout



TITLE: BPMS-Sensors	
Document Number:	
Date: 11/20/2012 7:11:45 PM	Sheet: 1/1
REU:	

Figure 2.11: BPMS final iteration sensors board schematic

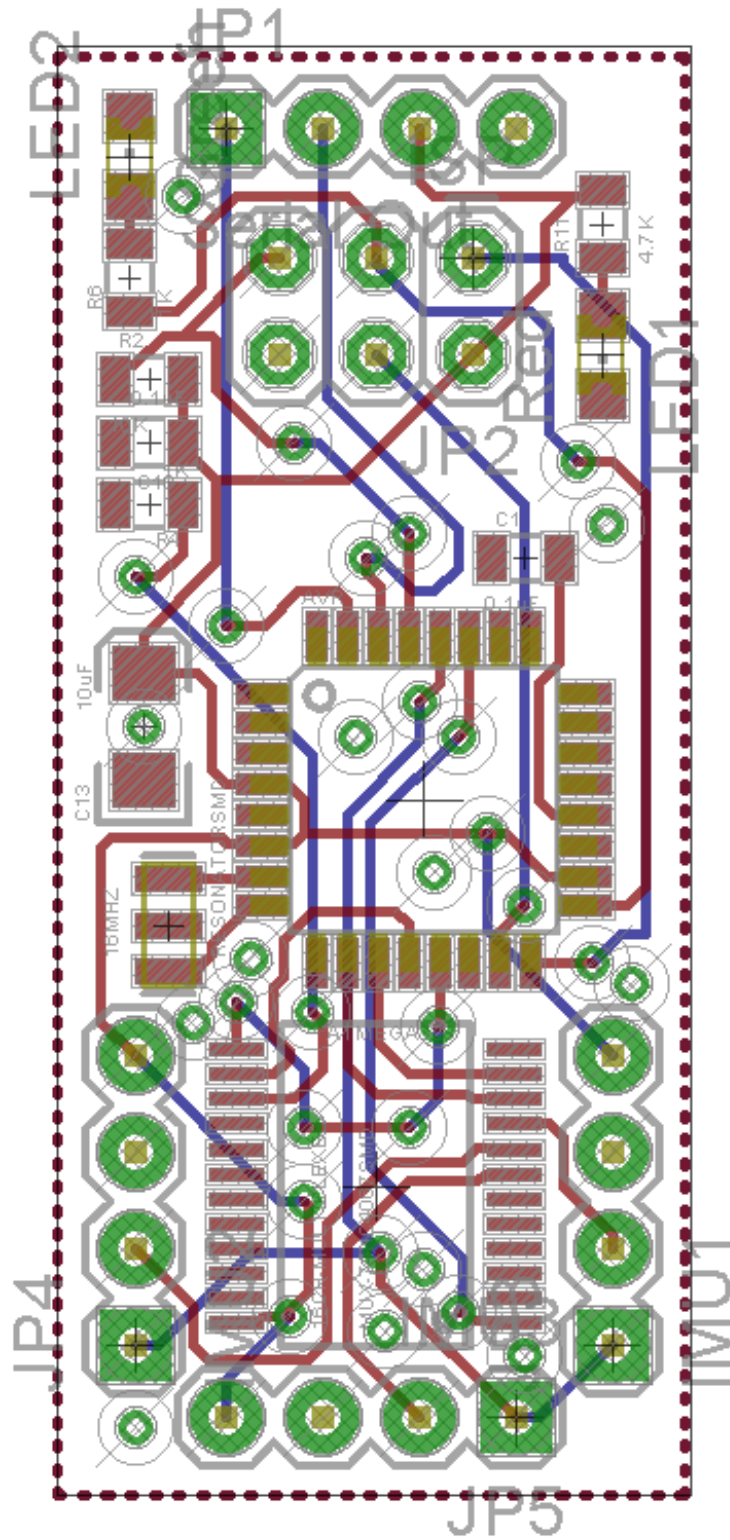


Figure 2.12: BPMS final iteration sensors board layout

2.3 Software

The following section describes in detail all the software that was written for this project. Initially the embedded code on the two BPMS boards and their concepts of operations will be described. This code relies on the Arduino IDE V1.0 [108]. Following the embedded code section, the BPMS main software is described. This portion of code is written in Python V2.7 and takes advantage of several open source libraries. All the dependencies will be defined and their versions noted. The code is platform independent (it can run on Windows, OS-X, or Linux) but it has been developed mainly on Windows. The BPMS main control suite has also been tested on Linux based systems without any modifications and it is assumed to work on OS-X, although the latter has not been tested.

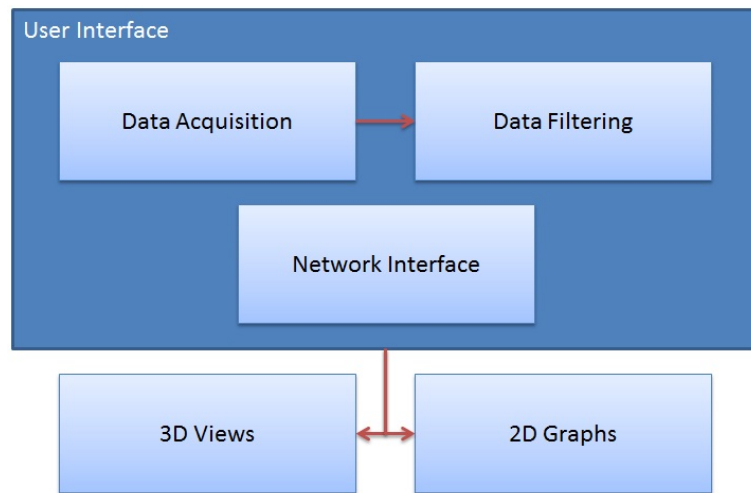


Figure 2.13: BPMS main software architecture diagram

2.3.1 Embedded software

BPMS's as previously described, makes use of seven small microcontrollers to acquire data from the sensors and forward it in a usable fashion to the wireless interface. In order

to do so, logic is required. Of those seven microcontrollers, six execute the same task (acquire data from the sensors) while a seventh arbitrates the communications to the wireless interface. In this section, the logic running on both groups of sensors is described as well as the necessary steps required to prepare the various microcontrollers to run said logic. All the microcontrollers make use of the Arduino programming language basic functions and the "wire" library for I2C communications (all libraries required are provided with the Arduino software package and are all Open-Source [108]). In this implementation, the Arduino V1.0 IDE (Integrated Desktop Environment) was used (most up to date version of the IDE at the time) therefore any reference below doesn't necessarily apply to later versions of the software. In order to upload and run an Arduino sketch, the bare AVR processors must be equipped with a bootloader. A bootloader is a program that resides on the microcontroller that allows the microcontroller to initialize properly and recognize commands. The bootloader is, in some way, the operating system of the microcontroller and it is the first program to run as soon as the microcontroller is turned on or powered up. Since in BPMS all the electronics were designed and assembled in house, the AVR processors used were virgin (virgin as in without any bootloader installed) so, in order to successfully upload and execute a program on the processors, a bootloader was required. The following section will guide the reader through all the necessary steps.

2.3.1.1 Flashing the Arduino Bootloader

Flashing the bootloader on a microcontroller is a simple task as long as the all the necessary steps are taken. There are several online forums and how-to guides available [110] but unfortunately none of them refer specifically to the boards here used for BPMS. For this reason all the necessary steps will be here explained. One must be aware that

for uploading the bootloader on the BPMS boards, the boards must be fully populated but they do not need to have any external connections (to either other sensor boards or sensors) wired in. There are slight differences between the sensor boards and the main arbiter board but those do not justify treating the problem differently therefore a single description is given and when necessary the differences will be highlighted. First of all in order to upload the bootloader, we require an AVR programmer. There are several commercial products available and all work more or less in the same way. The programmer needs to be connected to the target board via the ISP (In-System-Programmer) breakout. The ISP breakout provides power and data and is self contained. Based on the type of programmer used, it is possible that a custom cable harness will be required. For this specific application, since a proper AVR programmer was not available an Arduino 2009 was used to function as a programmer. In order to use an Arduino as an AVR programmer, there are a series of tutorial and how-to guides online [109] that must be followed. Once the programmer device is set up and the target board connected, since there is no auto-reset function enabled on the BPMS boards, in order to enable this necessary feature, add a .1uF capacitor between the GND and RESET pins of the ISP connector. Once all is ready, power up the programmer, open the Arduino IDE, select the "Arduino Mini W/ ATmega328", select the appropriate programmer interface and finally burn the bootloader. If all goes as planned, you should now have the bootloader on the target board and the board will be ready to receive it's first program. This procedure is valid for both BPMS boards, but one additional step must be taken with the main arbiter board. Make sure that the Bluetooth module power jumper is disconnected otherwise there can be conflicts in the data transmission and the bootloader upload may not be successful. Due to the lack of autoreset capability on the BPMS boards, there is one last consideration to be

made that applies to uploading any code to the boards. A special harness has been made for this specific purpose and allows to manually trigger the required reset pulse. In order to upload any code on the BPMS boards, we need a reliable serial connection (at TTL level) between the programming computer and the board. First of all make sure that the usb to serial (FTDI) power source is set to 3.3V (Sensor boards can take up to 5V input as well as the arbiter board as long as the Bluetooth transmitter jumper is disconnected) connect the board to the serial lines and short the ISP reset pin to GND with a jumper cable. On the Arduino IDE once the code is ready, select the right board type (Arduino Mini W/ ATmega328) and upload your sketch. Once the upload begins (first blink of the TX led on the FTDI adapter) quickly disconnect the Reset to GND jumper. There is a small 200ms delay between the reset pulse (first blink) and the beginning of the upload, if the jumper is not disconnected in that brief window, the upload will fail and the process has to be attempted again until successful.

2.3.1.2 Sensors cluster logic

The following section will focus on describing the logic on the BPMS sensor boards. All sensor boards are programmed in the same way and the discrimination of "who is who" (sensor wise) is done at software level based on the arbiter query scheme. In simplistic terms, the sensor boards manage a cluster of 3 IMUs by initializing the sensors and timing the sensor query appropriately. Additionally the sensor boards verify that the sensors are operating nominally and return two diagnostic "error" bytes at the end of each transmission. Those bytes are encoded in such a way to enable pin pointing which sensor on which board has malfunctioned. In the remote case of a sensor malfunction an LED has been incorporated in the board. If the led turns on, an error has occurred and

has been detected. The visual feedback is mainly used for preliminary system diagnostics and it is not meant for general system operations. The general principle of operations of the sensor board is to initialize the three sensors and read from them. The board once the initial setup is complete will then wait indefinitely for an incoming serial communication. Once an incoming serial byte is detected, the contents of it are discarded and the current stored value of the sensors is returned and a new one is requested and stored in local memory. This strategy allows for very fast response time but it will always give back an old reading. As it will be later explained in the main software section, it is important to disregard the first sensor reading obtained after any time of inactivity of the system.

2.3.1.3 Arbiter logic

The following section will focus on describing the logic on the arbiter sensor board. If compared to the sensor boards logic, the arbiter logic is much more simple. It's main purpose is to request a new sensor reading from each of the six clusters and forward it to the Bluetooth radio and manage a constant 50Hz sampling rate. The working principle for the arbiter is to initialize the two on board multiplexers and then wait for an incoming transmission from the Bluetooth module. When a byte is received, based on it's content a specific output is triggered. recognizable bytes are 0x00, 0x01 and, 0x01. If a 0x00 byte is received, the arbiter in sequence sends two characters "\$#" to the radio, then sends a 0x00 to each sensor board, connects that board's output to the bluetooth module and waits for the transmission to end. Once the first board has completed it's transmission, the second board is queried and so on until all six clusters have sent the sensor data to the radio. Once all the sensor data has been sent to the radio, the arbiter sends a "#\$" to the Bluetooth module and waits until 20ms have passed since the beginning of the

transmission and then repeats the loop until another byte is received from the Bluetooth radio. Instead, if a 0x01 is received, the arbiter will not trigger the data return, instead it will return a battery voltage reading, and if the received byte is 0x02, it will return the following string "BPMS". This last command is useful for auto recognizing on which serial port BPMS is connected. Further details on the use of this function will be given later in the main software description.

2.3.2 Main software package

BPMS on it's own, without the core software is of limited usefulness. The instrumentation by itself is not capable of delivering sensor orientations, but only raw sensor measurements. It is the core software purpose to fuse all the sensor information and provide an accurate, calibrated, body angle orientation estimate. The main software package for the system was developed initially as a flexible tool for debugging and system development. Python was used to write and execute the logic due to it's ease of implementation and cross platform capabilities. The BPMS software suite relies on the following dependencies (module name, import name, version):

- Python Distribution version 2.7.2.3
- PyQT, "PyQt4", version 4.9.4
- VPython, "visual", version 5.74
- Standard Time library, "time", version 2.7.2.3
- Standard String library, "string", version 2.7.2.3
- Standard Threading library, "thread", version 2.7.2.3
- Numerical Python, "numpy", version 1.6.1
- PySerial, "serial", version 2.5.0
- Standard Multiprocessing library, "multiprocessing", version 2.7.2.3
- Standard Socket library, "socket", version 2.7.2.3

- Standard Math library, "math", version 2.7.2.3
- Standard String Manipulation library, "struct", version 2.7.2.3
- Standard MS VC++ routines library, "msvcr", version 2.7.2.3
- Standard System level functions library, "sys", version 2.7.2.3
- Standard Operating System level functions library, "os", version 2.7.2.3
- Standard Pickling library, "pickle", version 2.7.2.3
- Standard Random Number Generator library, "random", version 2.7.2.3

The software suite is split in 17 files that group the various functions in an organized fashion. The BPMS software suite is multi-threaded and can be complex to follow but it is not the scope of this section to describe all the internal functions required for stable operations and we will here focus exclusively on the functionality enabled by the application and not the guts of how it is accomplished. The BPMS software suite has been created to allow an investigator to use the system without any particular knowledge of its low level functionalities. In order to maximize the usability of the software infrastructure, particular care has been given in the definition and implementation of the various graphical interactive menus and windows. In order to launch the BPMS suite, all the aforementioned dependencies must be installed on the host computer. Once the dependencies are installed, the user can launch the program by double clicking on the "BPMS-Launcher.bat" or by opening a console window, navigating to the folder where the BPMS scripts are located and typing "ipython -pylab -wthread Main.py". The Application will load all the required libraries and will display a start-up menu.

The start-up menu allows the user to choose the operations mode for the system. The options include reading the realtime data from BPMS, receiving or broadcasting real time data over the network or reading from a pre recorded raw data file. In addition to the modal setup, the start-up menu allows the user to manage the filter and system options

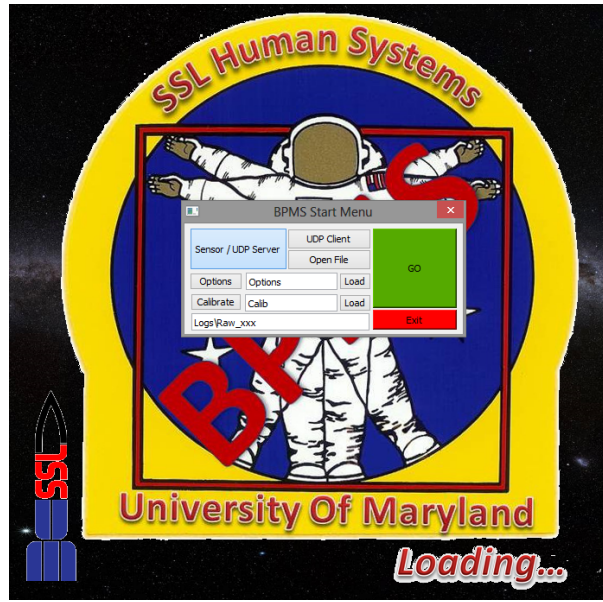


Figure 2.14: BPMS start-up user interface

or to load a pre stored options configuration file, and finally it allows importing a custom calibration file or to initiate the sensor calibration of a connected BPMS system. If the user chooses to disregard the options menus, the default options and calibration files will be loaded. In order to modify the default start-up configuration files, the user must only modify the "Option.cfg" and "Calib.cfg" files. Those files can be modified manually or through the GUI by overwriting the default configuration files. In case the system doesn't detect any of the default configuration files or if a non compatible configuration file is loaded, the system will proceed to load the factory default configuration parameters and will notify the user via a warning window.

Once the user has configured the system by pressing the green GO button the system will be initialized and depending on the mode chosen different options and functions will be enabled. Before expanding on the specific modal functions, the common functionalities will be described:

2.3.2.1 Standard Data View

The standard data view is the core of the BPMS software suite. It enables the user to control data steaming, manage the system options, log data and most of all visualize the incoming information in practical and calibrated way. The data view can be modified to the user's liking by editing the Options menu. calibration options such as sensor gains can be modified and applied in runtime as well as the visualization of the filtered euler angles in both radians or degrees. The main data view allows the user to select single sensor data in any order (by pressing the ctl key and selecting multiple cells) this feature is particularly useful when managing 2D data plots.

Credits	AccX	AccY	AccZ	MagX	MagY	MagZ	GyroX	GyroY	GyroZ	Roll	Pitch	Yaw	Error
R Leg	9.589	-1.102	-1.461	62.739	-60.451	-2.449	-11.332	5.722	2.166	-134.7...	78.562	127.804	---
R Foot	-5.989	-2.102	7.128	-58.418	25.839	1.589	0.728	-9.274	-0.957	-16.629	-39.047	152.807	---
R Shank	9.335	0.949	1.2	105.433	48.338	16.982	-1.677	-0.62	-0.617	45.095	80.48	142.051	---
R Hand	-9.663	-2.18	-0.245	-82.598	33.52	-70.788	5.691	7.726	-1.916	-91.04	-76.035	-124.4...	---
R Arm	9.338	0.774	2.097	79.338	-57.606	5.408	-4.607	6.205	-5.807	11.374	76.385	-67.117	---
R Forearm	9.431	-3.084	1.004	79.905	13.344	23.751	-7.759	1.386	-3.614	-75.855	70.899	37.965	---
Hip	-9.336	-0.019	1.202	-53.55	43.535	87.257	-6.648	-0.427	6.698	5.322	-84.644	21.925	---
Neck	7.915	-1.828	5.205	0.694	34.071	73.34	-0.035	-0.872	-1.356	-16.8	53.602	131.996	---
Spine	-9.81	0.388	-0.377	-73.828	-0.298	60.44	-4.147	1.323	-4.102	163.492	-84.794	-166.4...	---
L Hand	-9.847	0.26	0.588	-86.877	-36.523	-64.694	14.136	3.589	1.384	34.693	-85.208	176.114	---
L Arm	9.438	-0.647	-0.96	64.449	88.524	-56.217	-11.42	-2.073	2.67	-177.5...	82.209	-115.5...	---
L Forearm	8.702	3.663	-0.902	99.473	-41.363	67.574	-4.811	5.692	-1.611	108.468	67.474	-28.902	---
L Leg	9.103	1.911	-1.747	93.611	71.591	-13.802	-0.357	8.646	-1.754	128.98	74.462	-135.23	---
L Foot	-4.765	2.089	8.078	-71.274	35.658	46.827	-1.42	2.825	-2.002	14.221	-29.579	147.437	---
L Shank	9.622	0.203	1.638	66.095	-10.493	43.03	5.32	4.929	-2.158	6.864	80.051	-154.8...	---
R Shoulder	-9.698	-1.317	1.253	-99.252	37.676	50.669	-0.341	0.442	-4.077	-38.509	-79.117	92.648	---
L Shoulder	-9.209	3.782	0.618	-59.936	32.693	57.17	-4.666	7.647	-1.733	78.743	-68.742	-69.062	---
Head	2.763	0.442	-8.716	70.915	37.428	-53.28	-0.158	-1.368	-3.729	177.204	14.042	-31.992	---

Control Panel: Raw Filtered (Time) Start/Stop Logging [Hz] [Error Count] Reset Data Plot 3D View UDP Server Options Start/Stop Data Reset Zero

Figure 2.15: BPMS standard data view user interface

2.3.2.2 Custom plots generation

The ability to generate custom plots of real time data is very useful feature that allows the investigators to monitor the BPMS incoming data or the BPMS filtered IMU angles. In order to generate a data plot, the user must first select the cells that are of interest. Multiple cells can be selected and will return a multi curve plot. The automatic plot generator appropriately labels the plots and presents the user a graphical 60 second history of the data. Single plots can be hidden from the view by clicking on the legend on the right side of the plot and by clicking and dragging with the left mouse button the user is able to zoom in the graph. By pressing the right mouse button, the plot zoom is reset. There are no limitations other than the host computer available power to the number of curves per plot and the number of total plots produced per session.

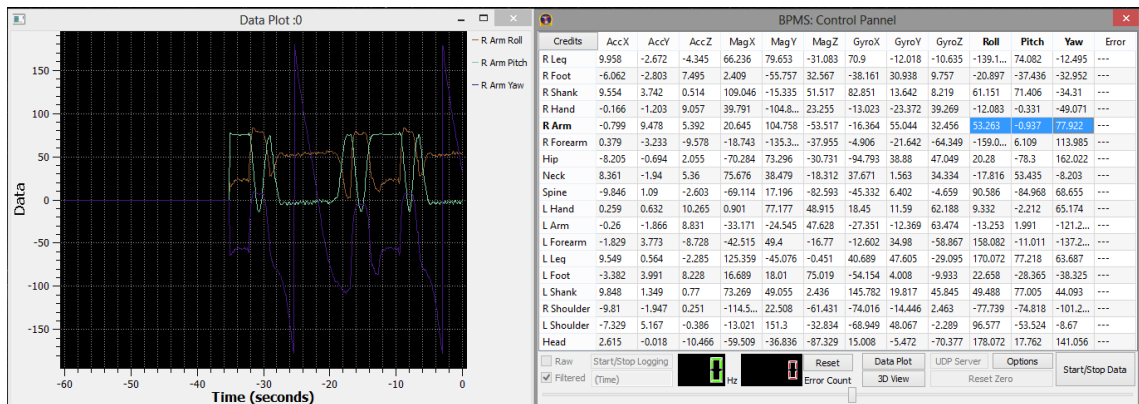


Figure 2.16: BPMS selective data plot user interface

2.3.2.3 3D human model views

By pressing the "3D View" button, the user will be presented with a 3D body model. The user can launch any number of those windows (the number is limited by the

host computer power and might become more laggy as more windows are open. Each window is independent allowing several model perspectives to be viewed simultaneously. The 3D views are interactive and allow the user to rotate (hold right mouse button), pan (arrow keys) and zoom (Right and Left mouse buttons) the view. A few additional features have also been added to enable quick pre-set views (pressing x,y or z will change the view perspective to be perpendicular to the corresponding axis), also a debug option has been included ("d" key) that toggles the display of the reference axis for each IMU. Finally by pressing the "r" key the scene is reset.

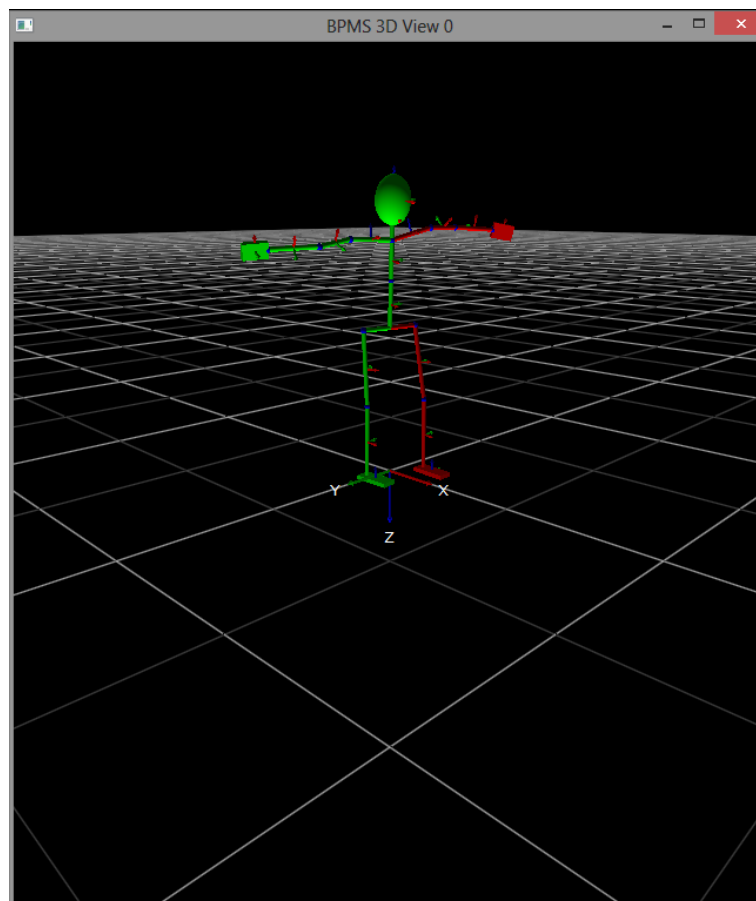


Figure 2.17: BPMS 3D human model view user interface

2.3.2.4 File operations

In the bottom left corner of the standard data view is located the file management portion of the BPMS suite. The BPMS suite is able to record to a file the raw sensor data as well as filtered data (euler angles). In order to enable data recording, the data streaming has to be initiated. The user is then able to input a file name (if none is given, a time stamp will be used). If a name has been given, the system will append a sequential number that will increase every time the user presses the "Start/Stop Logging" button. In order to enable logging at least one of the two logging options must be selected. The options include saving the raw data or the filtered IMU orientations. Keep in mind that only raw data can be played back later. The system will save the logged files with the name given and will add to the beginning of the file name the terms "Raw" or "Filtered" according to what data is stored within.

2.3.2.5 Reset Zero function

The "Reset Zero" button allows the user to toggle the sensor alignment algorithm and allows for corrected filtered values and graphical views. The reset button is only enabled once data streaming has begun and is launched automatically upon initiating the data stream. The user must keep in mind that the reset zero function only affects the logging of the filtered data and not the raw sensor values and it is necessary for a 1:1 correspondence between the 3D model view and real subject motion.

2.3.2.6 Realtime monitoring of errors and filter timing

The main BPMS software suite allows users to monitor in real-time if there have been sensor errors and the average reception rate. BPMS is tuned to operate at 50 Hz,

the monitor present on the standard data view displays the average reception rate at which the system is receiving packets. Any large discrepancies from 50 HZ and the user will know that there are some issues in the data transmission. Issues include but are not limited to: Not fast enough host computer, low battery on BPMS, wireless interference, radio almost out of range, sensor errors, etc. Lastly the error counter indicates if there have been any single sensor errors in the system. Those errors will also appear in the last column of the extended data view and usually represent, low system battery, a faulty IMU sensor, a faulty solder joint or most commonly noise in one of the I2C lines due to increased humidity on the garment (sweat). In this last case it is recommended to stop the test and dry the garment before continuing with the data acquisition.

2.3.2.7 Sensor mode with UDP server capabilities

The BPMS software suite as soon as the "Go" button is pressed, will scan the host system for a connected BPMS system. Once the system is found the standard data view will be displayed. If a BPMS system is not detected, the software suite will exit. In this case, the investigator must verify that the BPMS system has been paired to the host computer and that the batteries on BPMS are fully charged and connected and then try again. Once the initialization has completed, by pressing the "Start/Stop Data" button BPMS is going to commence the data acquisition. All the features described above will be available. In this mode, once the data streaming has begun the user can enable data forwarding to the network by pressing the "UDP Server" button. Before pressing the server button one must verify that the server IP and ports have been adequately configured in the options menu.

2.3.2.8 UDP client mode

The UDP client mode differs from the first mode just in the way the data is received. In the first mode, the BPMS software suite receives the data from the real BPMS sensor system, while in this mode, the data is retrieved over the network. This feature is particularly useful when BPMS is used to control third party systems or if multiple investigators want to monitor and plot on different systems the data being acquired. All plotting, visualization and saving features are enabled in this mode. In order to initiate data reception, the user must press the "Start/Stop Data" button and must have configured properly the server IP and forwarding ports under the options menu.

2.3.2.9 File mode

If the file mode is chosen the user will be presented with a file navigation window that will ease the selection of the file to be opened. Once a file has been selected, the file is verified and checked for compatibility. Once the file is ready to be played back, a series of buttons will activate. On the bottom portion of the general data view, a slider bar will be enabled, allowing the user to navigate through the data file. In order to commence playback the user must press the "Start/Stop Data" button. As the user will notice as the file is played back the slider bar will move accordingly to indicate the advancement in the file. The slider and the options button are deactivated once playback is initiated and are re-enabled when playback is paused. In this mode Raw recording is disabled while Filtered file recording is available once playback is initiated. All plotting and visualization features are enabled in this mode.

2.3.3 Additional tools developed

In addition to the development code explained above, several other scripts have been written to deliver long term usability of BPMS. This section will expand on the most important tools that were developed. While the entire source code for all the BPMS software is left for the SSL archives and possibly for future publication in an open source repository, appendix C has been added to include the source code for the IMU class and the 3D visualization routines that alone enable a new user to start exploring the possibilities enabled by BPMS.

2.3.3.1 IMU class

The IMU class is identical to the IMU class included in the main BPMS software suit, with the only difference that it has been decoupled from the overall BPMS architecture. This class allows the user to import, initialize and begin filtering IMU data in both realtime from BPMS or from a raw BPMS data file. This slim version of the IMU class allows the user to filter raw BPMS data files without having to wait the real time processing of the recorded data hence reducing post-processing time when real-time 3D visualizations are not necessary. All the features of the original IMU class are preserved allowing the user to choose a filtering scheme for the sensors, import sensor calibrations and access rotation matrices and temporary filter parameters, etc . This class was of particular use in the initial development phases of the project since it allowed the investigation of all the variables in real-time without the graphical layout computational overhead and complexity. The class is not multithreaded therefore simplifying even more the understanding of the data flow. In addition to the above the class on it's own allows for an easy integration in third party software applications and relaxes the requirement for new users to fully understand the

complexity of the BPMS main software suite and enables light weight applications of the system on embedded platforms. Also the class only makes use of the following Python standard libraries: VPython, Math and NumPy.

2.3.3.2 3D visualization subroutines

The graphical subroutines enable the users to include properly formatted 3D views of all the BPMS data. This includes a 3D human model that can receive data directly from the IMU class as well as single IMU visualizations. The single IMU visualizations were used to verify the IMU run time offset calibration algorithms on all sensors and allows the user to select between a 3D frame view and a rectangle view of the IMU orientation. This routine also allows the users to display the original offset IMU coordinate frame as well as the offset corrected reference frame.

2.3.3.3 Translator scripts

The translator scripts were mainly developed and used to extract single IMU data from a data recording and produce a comma separated text file that would be easily imported in Matlab for post-processing. The translator scripts allow the extraction of:

- Single IMU calibrated or raw data from a BPMS raw data file
- Single IMU orientation (Euler angles) from a BPMS filtered data file
- Single rigid body orientations from a Qualisys Motion Capture System data file
- Single rigid body orientation from a Vicon Motion Capture data file

All of the scripts above allow for iterative evaluation of a single file making the extraction of multiple sensor data fast and efficient. All the translation scripts are command prompt

based except for the file selection and file saving features, which make use of the operating system graphical file navigation routines. The scripts once launched will pop-up a file navigation windows and ask the user to select the source file from which to extract the data. Once the file is selected it is verified for compatibility with the script. If the file fails the check, the user will be prompted to select a different file. Once an appropriate file is selected, a new pop-up navigation window will appear asking the user for the file name and folder location of the output file containing the extracted data. Once a folder and a file name is entered, the script will return to the command window and ask the user to select the IMU ID to extract (the script will also display all the available options). Once the IMU data is extracted the script will ask the user if another IMU ID needs to be extracted and if not the script will terminate.

2.4 System calibration

Once all the hardware is assembled and integrated on the BPMS garment and the embedded software has been loaded on the BPMS microcontrollers, in order to have a usable system, one last step must be undertaken: Calibration. BPMS requires two types of calibrations, an initial sensor calibration and a task based pose calibration. MEMS sensors usually present some non zero offsets and those can impinge severely in the accuracy, precision and responsiveness of the orientation measurements. In order to eliminate or at least minimize those errors a calibration strategy and the relative software tools have been developed. The initial calibration is required once and must be repeated only in the unlikely event that one or more sensor modules are replaced. In addition to the initial calibration, since the purpose of BPMS is to measure body pose, we must provide a way to reset the initial arbitrary sensor orientations and map them against a simple body model.

The sections below will expand on those two calibration schemes and will provide all the details required to understand and execute those tasks.

2.4.1 Initial system calibration, determination of sensor offsets

As previously introduced, MEMS sensors, especially low cost variants, usually present zero offsets. Those sensor offsets are the result of manufacturing tolerances and in general do not severely degrade the overall sensor performance as long as they are well documented and compensated for in the filtering software. In BPMS each axis of each sensor is inspected for zero bias. Due to the different nature of the the comprising sensors of each IMU, the initial calibration varies for each sensor type. In any sensor calibration, the sensor readings are compared against a known value. For the BPMS sensors, three reference values will be used; for the accelerometers, we will use gravity (1g), for the magnetometers, we will make use of the earth magnetic field, while for the gyroscope we will measure the sensor outputs when the sensor is static (no rotations or accelerations except for gravity). A specific script was written that allows the user to connect to BPMS, select a single IMU and axis to calibrate or calibrate the entire system or group of sensors, and finally produce a calibration file (calib.cfg) that can later be used in the initialization of the BPMS main software package. The script guides the user through the various steps of the process that are here explained in detail. A similar approach was introduced for gyroscope calibration in [91] and has here been expanded to the complete set of sensors.

2.4.1.1 Accelerometer calibration

The calibration of the accelerometer sensors is perhaps the hardest to conduct accurately. It must be conducted one sensor axis at a time and can be very tedious and time

consuming. For each sensor we must identify the maximum and minimum reading of the gravity vector. The script will guide the operator by asking for a specific IMU and axis to calibrate and it will also specify if the measurement is on the minimum or the maximum. Once the user has identified the specific IMU to calibrate and whether it is the maximum or minimum value to calibrate, the user must hold the IMU and begin data streaming. The script will display the raw sensor values for the axis that is being calibrated, the user then has to orient the IMU in order to reach its maximum or minimum value. Once the the user has identified the orientation that gives the maximum or minimum sensor reading, he must hold the IMU as steady as possible and press a key on the keyboard. The script will then proceed to acquiring 200 samples and taking the average of the maximum or minimum sensor reading. The operation has to be repeated for each sensor axis and direction separately therefore resulting in 108 calibration points for the entire system. The calibration script will then also evaluate the corrected acceleration gain in the following way:

$$\vec{Acc}_{offset,IMU,i} = \frac{Acc_{imax,IMU} + Acc_{imin,IMU}}{2} \quad (\text{with } i = 1, 2, 3 \text{ (sensor axis)}) \quad (2.3)$$

$$Gravity_{IMU} = \frac{\sum_{i=1}^3 Acc_{imax} - \frac{Acc_{imax} + Acc_{imin}}{2}}{3}, \quad (\text{with } i = 1, 2, 3 \text{ (sensor axis)}) \quad (2.4)$$

$$AccGain_{IMU} = \frac{9.81}{Gravity_{IMU}}, \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.5)$$

The above acceleration gain represents the conversion value between raw sensor readings and the calibrated acceleration in m/s^2 .

$$\vec{Acc}_{(m/s),IMU} = AccGain_{IMU}(\vec{Acc}_{raw,IMU} - \vec{Acc}_{offset,IMU}), \text{ (with } IMU = 1, \dots, 18 \text{ (sensorID))} \quad (2.6)$$

where $\vec{Acc}_{raw,IMU}$ is the raw sensor output vector.

2.4.1.2 Magnetometer calibration

The magnetometer sensors calibration is very simple and can be conducted on all magnetometers in a single session. Before proceeding to the magnetometer calibration, one must make sure that the environment where the task will be executed is as free as possible from magnetic disturbances. Do not run the calibration in close proximity of electronic devices, big metal objects, or high current devices or wires. BPMS itself is a negligible source of magnetic distortion due to the very low currents involved in the wiring and also due to the fact that all cables and PCB boards are shielded. A very practical approach is to place the garment in a box and calibrate all magnetometers at once. The magnetometer calibration differs from the accelerometer calibration since it can occur in a very dynamic environment. The purpose of the calibration is to measure the maximum and minimum recorded values of each sensor axis. It is important to keep the system in the same location and vary only its attitude. Once the recording has begun the user has 2 minutes to rotate the BPMS garment in every orientation. Once the recording has been completed, the script recognizes the maximum and minimum values recorded for each axis and by evaluating the mean between them it derives the zero bias for each sensor axis in the following way:

$$\vec{Mag}_{offset,IMU,i} = \frac{Mag_{imax,IMU} + Mag_{imin,IMU}}{2} \text{ (with } i = 1, 2, 3 \text{ (sensor axis))} \quad (2.7)$$

$$MagScale_{IMU} = \frac{100}{Mag_{imax,IMU} - Mag_{imin,IMU}}, \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.8)$$

$$\vec{Mag}_{gauss,IMU} = MagScale_{IMU}(\vec{Mag}_{raw,IMU} - \vec{Mag}_{offset,IMU}), \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.9)$$

where $\vec{Mag}_{raw,IMU}$ is the raw sensor output vector.

2.4.1.3 Gyroscope calibration

Gyroscope calibration is the simplest and quickest of all the three sensors calibrations. In order to conduct a precise and reliable calibration, BPMS must be kept as static as possible (reduce any vibration to the absolute minimum) for a duration of 10 seconds. the calibration script will ask the user to press a key to initiate the sensor readings. Once the sensor acquisition has begun, the script will record 5000 readings per axis and average the results. The so obtained value will be used in the initialization of the orientation filter to eliminate the sensors zero bias $\vec{Gyro}_{offset,IMU}$.

$$Gyro_{Gain} = IMU = 0.001214225560612455(rad/s) = 0.06957(deg/s) \quad (2.10)$$

winch is the sensitivity of the sensor and is defined in the sensor data sheet [96].

This parameter is identical for all three axis and for all sensors on BPMS.

$$\vec{Gyro}_{rad/sec,IMU} = Gyro_{Gain}(\vec{Gyro}_{raw,IMU} - \vec{Gyro}_{offset,IMU}), \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.11)$$

where $\vec{Gyro}_{raw,IMU}$ is the raw sensor output vector.

2.4.1.4 BPMS prototype calibration

The final prototype of BPMS was calibrated according to the scheme above. The following table shows the calibration parameters that were obtained:

Table 2.2: BPMS prototype sensor calibration values

ID	Name	Accelerometer						Magnetometer						Gyroscope			
		Gravity	Ax Min	Ax Max	Ay Min	Ay Max	Az Min	Az Max	Mx Min	Mx Max	My Min	My Max	Mz Min	Mz Max	Gx Avg	Gy avg	Gz avg
0	R leg	262.3333333	-271	261	-273	252	-266	251	-422	470	-392	382	-463	511	-45.88	49.25333333	7.136666667
1	R foot	266.6666667	-268	266	-257	275	-276	258	-512	402	-477	400	-512	412	-5.53	26.69666667	-10.75666667
2	R shank	263.1666667	-269	268	-259	268	-260	255	-512	453	-512	511	-456	511	-1.103333333	-7.916666667	-10.86333333
3	R hand	262.6666667	-262	271	-268	254	-270	251	-458	485	-462	433	-378	340	-24.19666667	58.05	-32.54
4	R arm	266	-270	270	-266	266	-274	250	-512	479	-417	443	-512	472	-7.22	40.19666667	-16.47666667
5	R forearm	259	-264	254	-270	258	-263	245	-512	511	-461	419	-416	372	-3.523333333	10.91666667	-25.95
6	Hip	261.1666667	-278	260	-259	264	-262	244	-512	420	-332	414	-382	436	-17.56	30.85666667	62.27666667
7	Neck	260.1666667	-275	253	-259	267	-282	225	-512	413	-490	509	-512	495	2.49	18.47	0.513333333
8	Spine	264.8333333	-274	264	-259	272	-276	244	-467	506	-412	511	-430	332	-20.60666667	-6.983333333	-35.96666667
9	L Hand	262.5	-270	260	-252	276	-275	242	-512	506	-473	371	-512	302	33.19333333	56.59	17.89666667
10	L Arm	263.3333333	-273	255	-275	256	-272	249	-410	511	-401	511	-429	511	-32.14666667	-49.79333333	18.37333333
11	L ForeArm	268	-255	276	-265	268	-264	280	-391	511	-425	511	-331	413	22.85333333	105.81	-14.15333333
12	L Leg	262.8333333	-268	259	-263	276	-263	248	-396	390	-426	439	-509	508	-22.13333333	59.28	-10.20666667
13	L foot	262	-265	260	-255	271	-278	243	-512	483	-310	449	-367	401	-5.413333333	19.61	0.22
14	L shank	258.5	-263	260	-260	271	-254	243	-512	511	-512	511	-401	503	24.47333333	34.85666667	-30.01666667
15	R Shoulder	260.8333333	-280	247	-260	269	-278	231	-384	398	-377	373	-512	511	-19.90666667	4.35	-17.6
16	L Shoulder	259	-269	253	-253	271	-287	221	-512	411	-420	304	-512	438	-33.07	74.91666667	-6.903333333
17	Head	270.3333333	-268	261	-285	270	-297	241	-377	363	-512	417	-512	361	-10.27333333	-41.66	5.4

As previously stated, the BPMS main software package requires the user to import a calibration file on start-up, if the file is not found or is not readable, the start-up script will create a new default one. The calibration file doesn't need to be updated unless an IMU module is replaced and at that point one must only calibrate the specific IMU that was substituted. The calibration files stores each IMU's calibration data in the following way: $Gravity_1, Acc_{1xmin}, Acc_{1xmax}, Acc_{1ymin}, Acc_{1ymax}, Acc_{1zmin}, Acc_{1zmax}, Mag_{1xmin}, Mag_{1xmax}, \dots, Gyro_{1x}, Gravity_2, \dots$

This text file can be manually modified by the user to include manual single IMU updates but in general it is obtained directly from the calibration script.

2.4.2 Initial pose calibration for sensor alignment

In BPMS once the system has been calibrated, the various filters evaluate the orientation of each of the IMUs in the garment in the canonical North-East-Up coordinate frame. Unfortunately, the IMUs are not always aligned with the major long bones. This is due to the assembly orientation offset and on how the user dons the garment. In order to realign those arbitrary initial orientations, a methodology had to be developed. First of all a calibration pose had to be defined. The calibration pose had to be highly repeatable and also had to be achievable in a pressure suit. Figure 2.4.2 below shows the standard calibration pose adopted in this work.

By defining a standard calibration pose, we are able to evaluate the IMU arbitrary initial offsets. Since the methodology is identical for all IMUs, we will here describe the general process that is later applied to all 18 sensor groups.

The BPMS main software suite allows for runtime calibration which means that the researcher can request the user to stand in the calibration pose and click on the calibrate button and the model will evaluate the offsets and return calibrated IMU orientations. This operation can be executed on both realtime data and recoded data therefore before any trial it is highly suggested to ask the test subjects to stand still for a time between 2 to 4 seconds (this allows for all the IMU orientation filters to stabilize and it also allows for accurate post processing calibration points. In addition to the initial calibration pause, it is a good practice to include also a calibration pause at the end of each trial in order to verify if there was any drift or filter deviation during the measurements. This post trial calibration is usually achieved by asking the test subject to return to the initial calibration pose (if possible in the same location of the initial calibration) and remain still for another 2-4 seconds.



Figure 2.18: BPMS Calibration Pose

2.4.2.1 Initial rotation of the sensor readings

Today's 3D graphics software often use euler angles to draw elements on the virtual 3D space and BPMS is no exception. As it is later expanded, the complimentary filter used in this work evaluates estimates of the IMUs DCMs and not of the IMUs euler angles. The rationale lies on the issue of gimbal lock. Rotation matrices are a non singular representation of the attitude of a system and therefore do not suffer from this issue. Unfortunately though the 3D visualizations of the system require as inputs the euler angles for each sensor and given the human body configuration and the original sensor axis alignment, 15 IMU out of 18 are very close to a singularity and therefore make the initial pose calibration an issue. To solve this problem, a simple strategy was adopted. The precise orientation of the sensors is unknown before the system is donned, but the general attitude of the sensors is known since it was determined during the manufacturing of the garment. Given this approximate initial orientation, all the IMUs which would be close to singular in the calibration pose are pre-rotated in the IMU filter initialization. This pre-rotation is obtained by pre-multiplying all the sensor readings by an offset matrix \mathbf{R}_{i}^*IMU before their values are fed to the filter in the following way:

$$\vec{Acc}_{rad/sec,IMU}^* = \mathbf{R}_{IMU}^* \vec{Acc}_{rad/sec,IMU}, \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.12)$$

$$\vec{Mag}_{rad/sec,IMU}^* = \mathbf{R}_{IMU}^* \vec{Mag}_{rad/sec,IMU}, \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.13)$$

$$\vec{Gyro}_{rad/sec,IMU}^* = \mathbf{R}_{IMU}^* \vec{Gyro}_{rad/sec,IMU}, \quad (\text{with } IMU = 1, \dots, 18 \text{ (sensorID)}) \quad (2.14)$$

In the specific, three \mathbf{R}_{IMU}^* were required and are as follows:

$$\mathbf{R}_{IMU}^* = I \quad (\text{for both the feet and head IMUs}) \quad (2.15)$$

$$\mathbf{R}_{IMU}^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad (\text{for the left and right shoulders sensors}) \quad (2.16)$$

Which represents a single 90 deg rotation over the X axis (roll)

$$\mathbf{R}_{IMU}^* = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 10 & \\ -1 & 0 & 0 \end{bmatrix} \quad (\text{for all the remaining sensors}) \quad (2.17)$$

Which represents a single 90 deg rotation over the Y axis (pitch) This pre-alignment allows for a much more consistent and accurate pose calibration strategy and reduces the possibility of inaccurate visualizations due to the definition of the plotting functions.

2.4.2.2 Pose calibration

The main function of BPMS is to estimate the body pose of a human subject and so far in this work, we are able to determine sensor attitudes and not body segments attitudes. The two are related but offset. The first step in converting those estimates requires us to understand the working principle. Each IMU returns the DCM or attitude of itself in the local magnetic inertial frame. The IMU has no recollection of what it is attempting to map therefore before going any further we have to define a simplified human body model that represents the test subject. In this work such model was obtained by defining a known, highly repeatable initial pose. The pose was then defined mathematically

as a sequence of vectors. Each body segment can be defined as a vector \vec{L}_i of magnitude L_i with an initial orientation $\mathbf{R}_{0,i}$. By assuming that each segment, in the initial pose has a $\mathbf{R}_{0,i} = \mathbf{I}$, we can define the calibration pose by engineering the \vec{L}_i vectors so that the stick model so created reflects such pose and applying those vectors at the end of the adjacent body segment vectors. The start of this vector chain in this work has been defined at the hip. Once the model has been defined, it has to be linked to the IMU attitude estimates. In order to do so, we ask the test subject to remain still while in the calibration pose and we sample each IMU and obtain it' initial DCM $\mathbf{R}_{\text{sensor}0,IMU}$. From this matrix, we can then represent the new, calibrated orientation of the body segment as follows:

$$\mathbf{R}_{\text{calibrated},i}(t) = \mathbf{R}_{\text{sensor}0,IMU}^T \mathbf{R}_{IMU}^*(t) \quad (2.18)$$

With the above, we can map the IMU sensors to the i body segments and obtain the initial orientation condition:

$$\mathbf{R}_{\text{calibrated},i}(t_0 = 0) = \mathbf{R}_{0,i} = \mathbf{R}_{\text{sensor}0,IMU}^T \mathbf{R}_{IMU}^*(t_0) = I \quad (2.19)$$

Since:

$$\mathbf{R}_{\text{sensor}0,IMU} = \mathbf{R}_{IMU}^*(t_0) \quad (2.20)$$

In the same way, we can define the body segments vectors as follows:

$$\vec{L}_i(t) = \mathbf{R}_{\text{calibrated},i}(t) \vec{L}_i(t_0) = \mathbf{R}_{\text{sensor}0,IMU}^T \mathbf{R}_{IMU}^*(t) \vec{L}_i(t_0) \quad (2.21)$$

2.5 Complimentary filter

The following DCM based complimentary filter is heavily based on Mahony's work and is widely explained in the following publication [13]. Initially developed for estimating

the attitude of quadcopters, and micro unmanned vehicles, this filter behaves wonderfully when implemented for human motion capture. The beauty of BPMS when compared to similar commercial systems lies in the access to the core of the filtering scheme which allows users to implement new and more advanced strategies as well as granting them full control over all the system's core components. In this section this work's specific DCM based implementation of the Mahony complimentary filter will be explained in detail [8, 16, 20, 54]. The drift free attitude estimation process begins on evaluating the nonlinear differential equation that relate the time rate of change of the direction cosines to the gyro signals. at this point, we wish to compute the direction cosines without making any approximations that violate the nonlinearity of the equations and also assume that the gyro signals have no errors. Later on we will address the issue of gyro drift. Unlike rotating mechanical gyros, which stay fixed in space while the object of interest rotates around them, electronic rate gyros rotate with it, producing signals proportional to the rotation rate. Since rotations do not commute, and the sequence of rotations matter, we cannot get by with simply integrating the gyro rate signals to get the attitude angles. We know that the rate of change of a rotating vector $\vec{r}(t)$ due to its rotation is given by:

$$\frac{d\vec{r}(t)}{dt} = \vec{\omega}(t) \times \vec{r}(t) \quad (2.22)$$

Where $\vec{\omega}(t)$ is the rotation rate vector. From the above, we can make the following considerations:

- The differential equation is nonlinear. The rotation vector input is (cross) multiplied by the variable that we are trying to integrate. Therefore, any linear approach will be only an approximation.

- Both vectors must be measured in the same reference frame.
- Because the cross product is anticommutative, we could reverse the order and change the sign.

If we know the initial conditions and the time history of the rotation vector, we can numerically integrate equation above and track the rotating vector:

$$\begin{aligned}\vec{r}(t) &= \vec{r}(0) + \int_0^t d\vec{\theta}(\tau) \times \vec{r}(\tau) \\ d\vec{\theta} &= \vec{\omega}(\tau) d\tau\end{aligned}\tag{2.23}$$

Where $\vec{r}(0)$ is the initial vector at time 0 and $\int_0^t d\vec{\theta}(\tau) \times \vec{r}(\tau)$ is the change in the \vec{r} vector due to the rotation. The objective is to relate those equations back to an attitude estimate of the sensor. Unfortunately those are expressed in the wrong frame (inertial frame). By referring to Mahony in [13, 16], and returning to the differential equation form of the above, we can write:

$$\begin{aligned}\vec{r}(t + dt) &= \vec{r}(t) + \vec{r}(t) \times d\theta(t) \\ d\vec{\theta} &= \vec{\omega}(t) dt\end{aligned}\tag{2.24}$$

if we then express the above in matrix form by rotating each unit vector of the body reference frame, we obtain:

$$\mathbf{R}(t + dt) = \mathbf{R}(t) \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix}\tag{2.25}$$

where:

$$\begin{aligned}
 d\theta_x &= \omega_x dt \\
 d\theta_y &= \omega_y dt \\
 d\theta_z &= \omega_z dt
 \end{aligned}
 \tag{2.26}$$

The matrix above represents the foundation of the DCM algorithm but it is still not complete. In the ideal world, where gyros return perfect measurements and where there is no finite time evaluation of the attitude, this formulation would return accurate estimates. Unfortunately though, the reality of things is different. Gyro measurements are far from precise and although the above formulation gives acceptable results, it will slowly drift over time. Additionally, the numerical errors will accumulate and the violate the orthogonality constraint of the DCM. So far the only assumption made so far in order to ensure the applicability of the algorithm on real hardware is that the time step must be small enough not to induce large variations in the DCM. In this particular case, the algorithm will be updated every 20ms. In order to compensate for those real world error, the first step is to mitigate the gradual DCM orthogonality condition degradation. To do so, we must renormalize the matrix by computing the dot product between the first and second rows of the R matrix which should be 0 (all the axis of a reference system must be orthogonal to each other) [25] by doing so we obtain a measure of how much the X and Y rows are rotating toward each other:

$$\begin{aligned}
 \vec{X} &= [r_{xx}, r_{xy}, r_{xz}]^T \\
 \vec{Y} &= [r_{yx}, r_{yy}, r_{yz}]^T
 \end{aligned}
 \tag{2.27}$$

$$error = XY = X^T Y = [r_{xx}, r_{xy}, r_{xz}][r_{yx}, r_{yy}, r_{yz}]^T$$

We can then apportion half of the error each to the X and Y rows, and approximately rotate the X and Y rows in the opposite direction by cross coupling:

$$\begin{aligned}\vec{X}_{ortho} &= \vec{X} - \frac{error}{2}\vec{Y} \\ \vec{Y}_{ortho} &= \vec{Y} - \frac{error}{2}\vec{X}\end{aligned}\tag{2.28}$$

It is easy to see that the orthogonality error is thereby greatly reduced keeping in mind that the magnitude of each row and column of the R matrix must be kept approximately equal to one. Apportioning the error equally to each vector yields a lower residual error after the correction than if the error were assigned entirely to one of the vectors. The next step is to adjust the Z row of the matrix to be orthogonal to the X and Y row by imposing the Z row to be the cross product of the X and Y rows:

$$\vec{Z}_{ortho} = \vec{X} \times \vec{Y}\tag{2.29}$$

The last step in the renormalization process is to scale the rows of the R matrix to assure that each has a magnitude equal to one. One way we could do that is to divide each element of each row by the square root of the sums of the squares of the elements in that row. However, there is an easier way to do that, by recognizing that the magnitudes will never be much different than one, so we can use a Taylor's expansion. The resulting magnitude adjustment equations for the row vectors are:

$$\begin{aligned}\vec{X}_{norm} &= \frac{1}{2}(3 - \vec{X}_{ortho}\vec{X}_{ortho})\vec{X}_{ortho} \\ \vec{Y}_{norm} &= \frac{1}{2}(3 - \vec{Y}_{ortho}\vec{Y}_{ortho})\vec{Y}_{ortho} \\ \vec{Z}_{norm} &= \frac{1}{2}(3 - \vec{Z}_{ortho}\vec{Z}_{ortho})\vec{Z}_{ortho}\end{aligned}\tag{2.30}$$

In this approach, there are fewer division and multiplications to execute when compared to other methods as well as no divisions or square roots. This allows for reduced computation times, which are critical in evaluating the full set of sensors of BPMS. So far the algorithm is self-correcting for numerical round off errors but still doesn't correct

for gyro drift. In order to do so, we must use other orientation references to detect the gyro offsets and provide a negative feedback loop back to the gyros to compensate for the errors in a classical detection and feedback loop. We can summarize the required steps as follows:

- Use a set of orientation reference vectors to detect orientation error by computing a rotation vector that will bring the measured and computed values of reference vectors into alignment.
- Feed the rotation error vector back through a proportional plus integral (PI) feedback controller to produce a rotation rate adjustment for the gyros.
- Add (or subtract, depending on your sign convention for the rotation error) the output of the PI controller to the actual gyro signals.

The main requirement for an orientation reference vector is that it does not drift. Its transient performance is not that important because it is the gyros that provide the transient fidelity for the orientation estimate. In this application, the two reference vectors used will be accelerometer and magnetometer readings.

We use accelerometers to provide a reference vector for the Z axis of the sensor and the magnetometer (tilt compensated compass) as a reference for the horizontal projection of the X axis (roll axis) of the sensor. Our two reference vectors happen to be perpendicular to each other. That is convenient, but not absolutely necessary.

For either of the two reference vectors, the orientation error is detected by taking the cross product of the measured vector with the vector that is estimated by the direction cosine matrix. The cross product is particularly appropriate for two reasons. Its magnitude is proportional to the sine of the angle between the two vectors, and its direction is

perpendicular to both of them. So it represents an axis of rotation, and an amount of rotation, that would be needed to rotate the measured vector to become parallel to the estimated vector. In other words, it is equal to the negative of the orientation rotational error. By feeding it back to the gyros through the PI controller, the estimated orientation is gradually forced to track the reference vectors, and gyro drift is cancelled. The cross product of a measured reference vector with a corresponding vector computed from the direction cosine matrix is an indication of the error. It is approximately equal to the rotation that would have to be applied to the reference vector to bring it into alignment with the computed vector. We are interested in the amount of rotation correction that we need to apply to the direction cosine matrix, which is equal to the negative of the error rotation. So, it is convenient to compute the correction by interchanging the order in the cross product. The correctional rotation is equal to the cross product of the vector estimated by the direction cosines with the reference vector. We use a proportional plus integral feedback controller to apply the rotation correction to the gyros, because it is stable and because the integral term completely cancels gyro offset, including thermal drift, with zero residual orientation error. The way that the reference vector errors map back onto the gyros is done via the direction cosine matrix, so that the mapping depends on the orientation of the IMU.

Accelerometers are used for roll-pitch drift correction by measuring the direction of the gravity vector. Unfortunately the accelerometers do not only detect gravity, but they also detect any acceleration acting on the sensor. Between all the accelerations that can appear in the measurement, we can separate three: Gravity, linear accelerations and centrifugal accelerations. Between the three, gravity is what we want to observe, centrifugal accelerations can be modelled and be taken in account through the gyro readings but we

also need an estimate of linear velocity which we do not have hence both centrifugal and linear accelerations remain as disturbances. On a side note, usually disturbance accelerations are not present for an extended period of time and therefore do not usually induce too much error in the estimates and are only used to correct for drift in the gyros. The accelerometer is used to evaluate the roll and pitch correction plane as follows:

$$CorrectionPlane_{roll,pitch} = ([r_{zx}, r_{zy}, r_{zz}]^T \times \vec{g}_{ref}) * W_{dynam} \quad (2.31)$$

where W_{dynam} represents a non linear weight that is used to reduce the influence of the accelerometer when a highly dynamical environment is detected. If the evaluated W_{dynam} is greater than 1 the weight is set to 1 and in the same way if the weight is less than 0 it's then set to 0. In any other case it is evaluated as follows:

$$W_{dynam} = 1 - 2 * |1 - Acc_g| \quad (2.32)$$

where Acc_g represents the magnitude of the accelerometer reading in gs.

Magnetometers on the other side are not subject to the dynamical environment and are also drift free, but still have some drawbacks. Magnetometers measure magnetic fields, on earth we have a constantly acting magnetic field that roughly points north but it's not very strong and can we also have many objects that can alter the magnetic field considerably like ferrous objects, currents, etc. We use the magnetometer to estimate the last piece of the puzzle which is the yaw correction plane as follows:

$$YawCorrectionPlane_{yaw} = (r_{xx} * \cos(yaw) - r_{yy} * \sin(yaw)) * [r_{zx}, r_{zy}, r_{zz}]^T \quad (2.33)$$

the yaw estimate is obtained by evaluating the magnetometer vector component on

the inertial ground plane. The inertial ground plane is defined by the previous iteration estimates of pitch and roll from which the new yaw estimate is obtained as:

$$\begin{aligned}
mag_x &= Mag_{raw,x} \cos(pitch) + Mag_{raw,y} \sin(roll) \sin(pitch) + Mag_{raw,z} \cos(roll) \sin(pitch) \\
mag_y &= Mag_{raw,y} \cos(roll) - Mag_{raw,z} \sin(roll) \\
Yaw &= atan2\left(\frac{mag_x}{mag_y}\right)
\end{aligned} \tag{2.34}$$

Each of the rotational drift correction vectors (yaw and roll-pitch) are multiplied by weights $W_{roll,pitch}$ and W_{yaw} and fed to a proportional plus integral (PI) feedback controller to be added to the gyro vector to produce a corrected gyro vector that is used as the input to the attitude estimation matrix. First we compute a weighted average of the total of the rotation corrections.

$$Correction_{Tot} = W_{roll,pitch} CorrectionPlane_{roll,pitch} + W_{yaw} YawCorrectionPlane_{yaw} \tag{2.35}$$

And then pass it to the PI controller as:

$$\begin{aligned}
\omega_{P,correction} &= K_p Correction_{Tot} \\
\omega_{I,correction} &= \omega_{I,correction} + K_i dt Correction_{Tot} \\
\omega_{correction} &= \omega_{P,correction} + \omega_{I,correction}
\end{aligned} \tag{2.36}$$

In this particular application, the gains and weights used were determined empirically and were obtained from [111] and are here summarized:

$$\begin{aligned}
W_{roll,pitch} &= 0.02 \\
W_{yaw} &= 1.2 \\
K_p &= 0.02 \\
K_i &= 0.00002
\end{aligned} \tag{2.37}$$

We then feed the gyro correction vector so obtained back into the rotation update equation by adding the correction vector to the gyro signal.

$$\omega(t) = \omega_{gyro}(t) + \omega_{correction}(t) \tag{2.38}$$

And then feed this measurement to the following:

$$\mathbf{R}(t + dt) = \mathbf{R}(t) \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix} \tag{2.39}$$

where:

$$\begin{aligned}
d\theta_x &= \omega_x dt \\
d\theta_y &= \omega_y dt \\
d\theta_z &= \omega_z dt
\end{aligned} \tag{2.40}$$

At this point, we have completed a full pass through the calculation which will be repeated in the next time step [16].

2.6 System verification and performance assessment

2.6.1 Experimental setup

In order to address the reliability and performance of the complimentary filter when applied to the specific BPMS hardware elements, a simple servo actuated rig was designed and built.

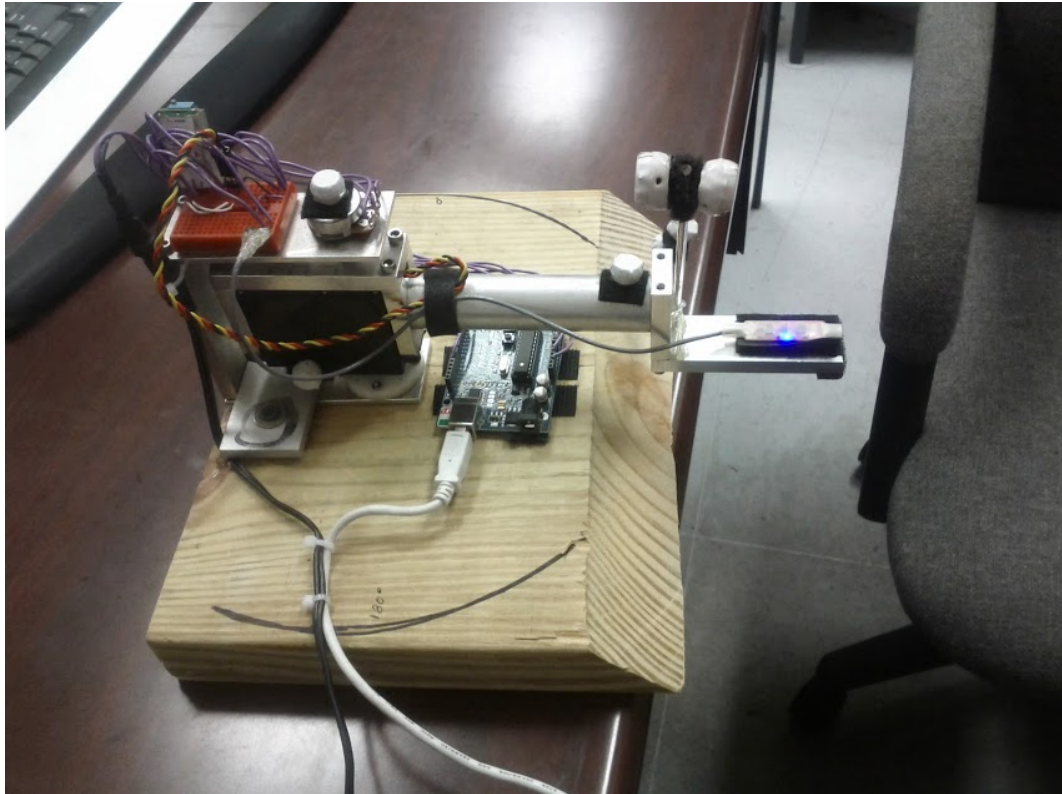


Figure 2.19: Sensor test rig

The test bed system includes an IMU sensor identical to the 18 sensors of BPMS, a linear potentiometer, an arduino and a series of VICON markers. The Arduino has been programmed to acquire the raw sensor readings from the IMU and potentiometer while at the same time commanding the servo to a specific position or angular speed. The microcontroller streams the acquired measurements and receives commands from and to a computer over serial protocol. A simple GUI was designed to monitor and control the system and allows the visualization of the filtered euler angles from the IMU overlaid over the calibrated potentiometer readings. At the same time the interface allows the user to control: servo position slew speed and sine wave frequency.

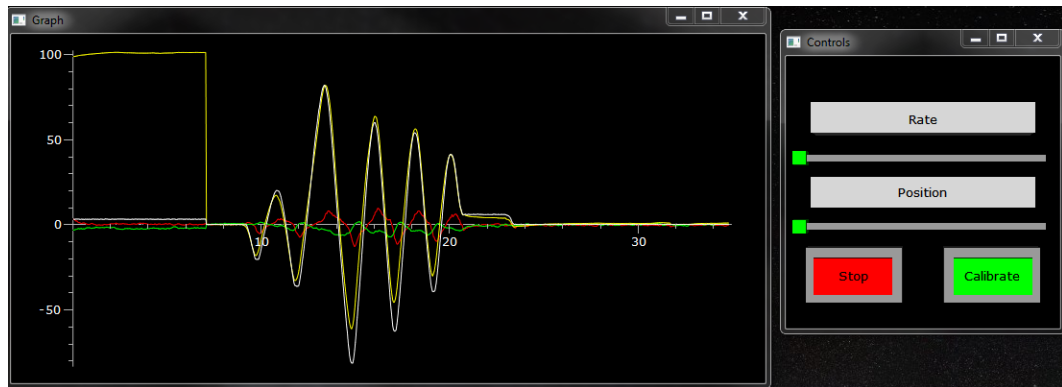


Figure 2.20: Sensor test bed GUI

Once the system was verified and calibrated (IMU and potentiometer) the setup was brought to the AVL lab where data was acquired while monitoring the system with the AVL Vicon motion capture setup. The system was anchored to a table to eliminate the possibility of slipping and performance was measured on the three main axis by commanding the system to execute a pure rotation on each of them. In order to allow for pure single axis rotations, the rig and the IMU were rotated accordingly and their initial attitude verified through a planar bubble level.

2.6.2 Results

2.6.2.1 Attitude estimate performance and IMU filter gains

In the experimental session data from the IMU, potentiometer and VICON was acquired and minimally processed. The data was reduced in Matlab by synchronizing the Vicon data with the potentiometer and IMU (the latter data is synced since it was acquired on the same machine and by the same acquisition device). Once the time offsets were determined, the IMU and potentiometer data was interpolated to allow for a point by point comparison with the Vicon counterpart. Vicon sampled at an average of 90 Hz while the IMU and the pot were set to 50 Hz.

As it was previously stated, the IMU filter gains were obtained from the literature but further analysis and verification was required. In the experimental session, the data incoming from the IMU was iteratively processed using different gains to verify if a new set of gains improved on the filter accuracy or overall performance.

In addition to the gain analysis, in order to address the performance variation as a function of IMU angular speed, a specific test pattern was used. The servo was commanded to execute a series of sine waves at progressively increasing frequencies to allow for the evaluation.

Finally the estimates errors between Vicon and potentiometer and the IMU and the potentiometer were evaluated by subtracting the measurement from a system against the other for four bands of frequencies for all the cases. The following plots summarize the findings:

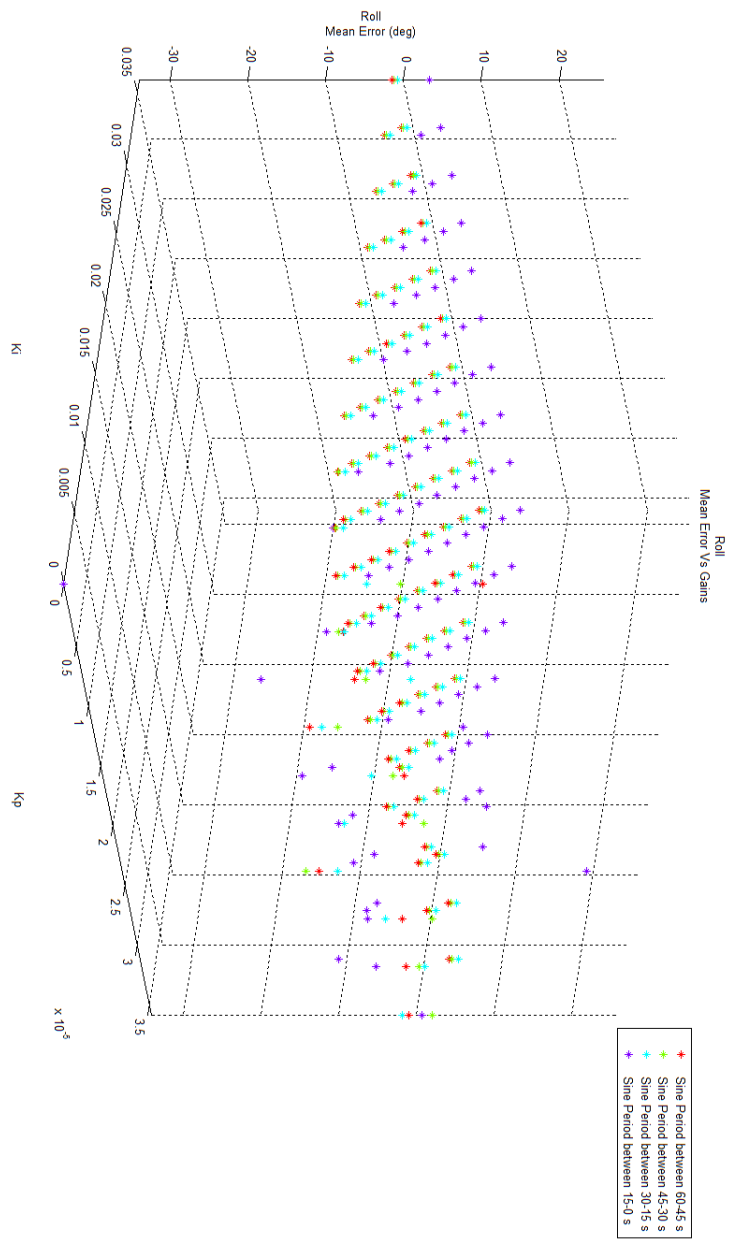


Figure 2.21: BPMS Vs potentiometer Roll mean error as a function of filter gains

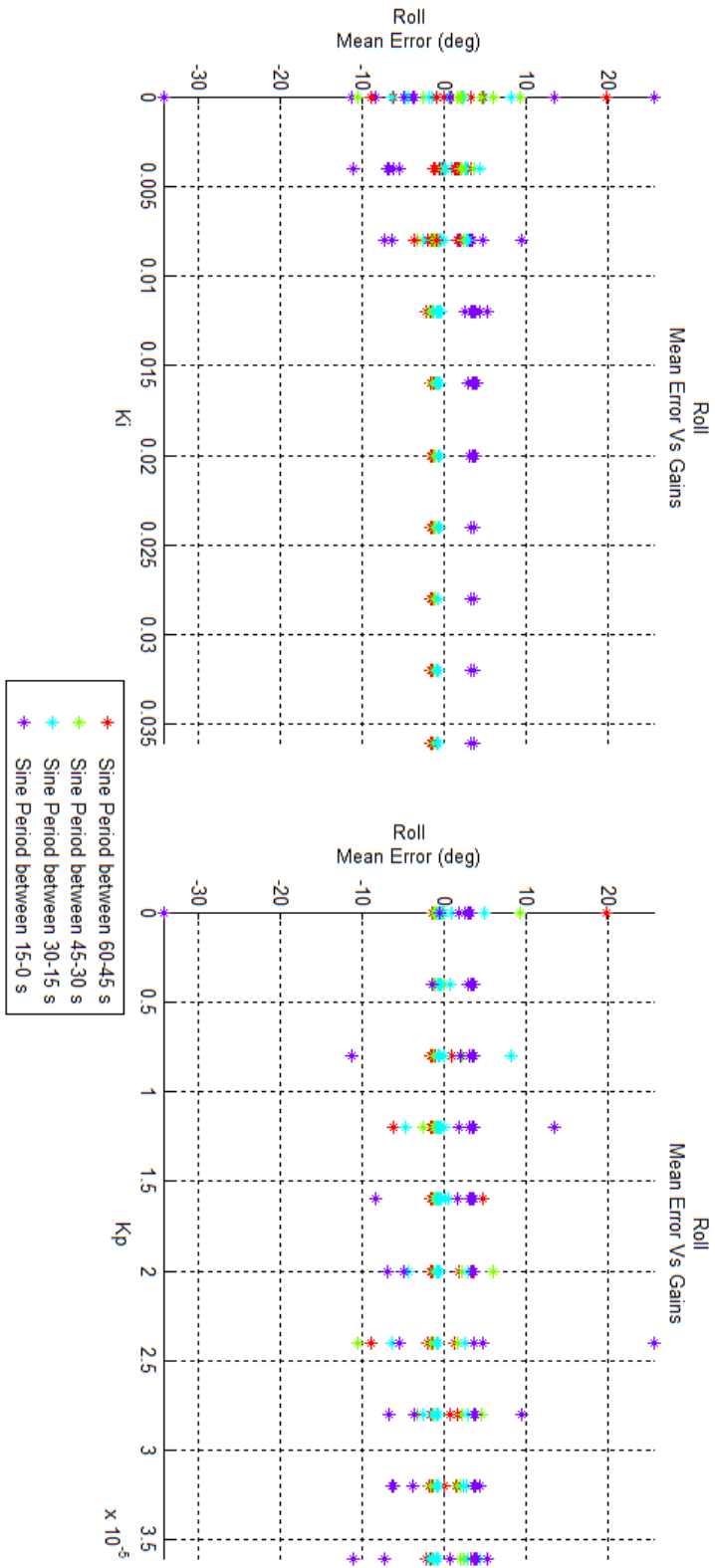


Figure 2.22: BPMS Vs potentiometer Roll mean error as a function of filter gains side views

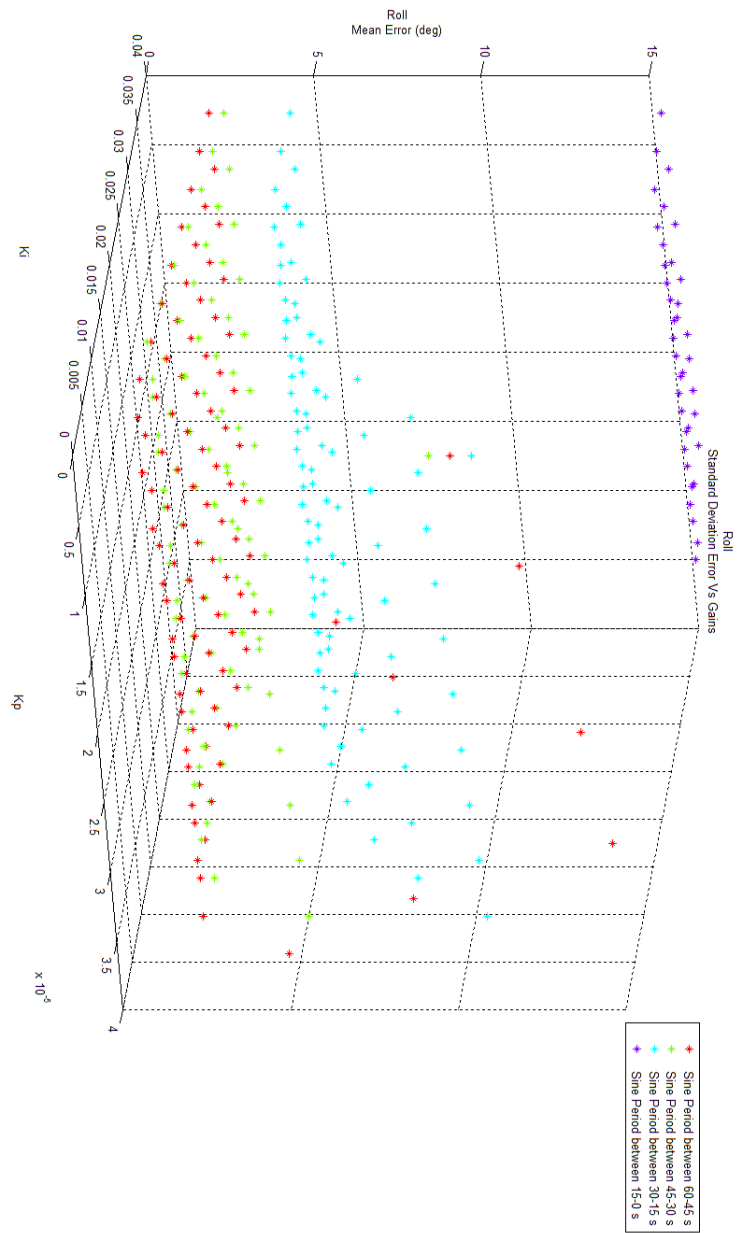


Figure 2.23: BPMS Vs potentiometer Roll standard deviation as a function of filter gains

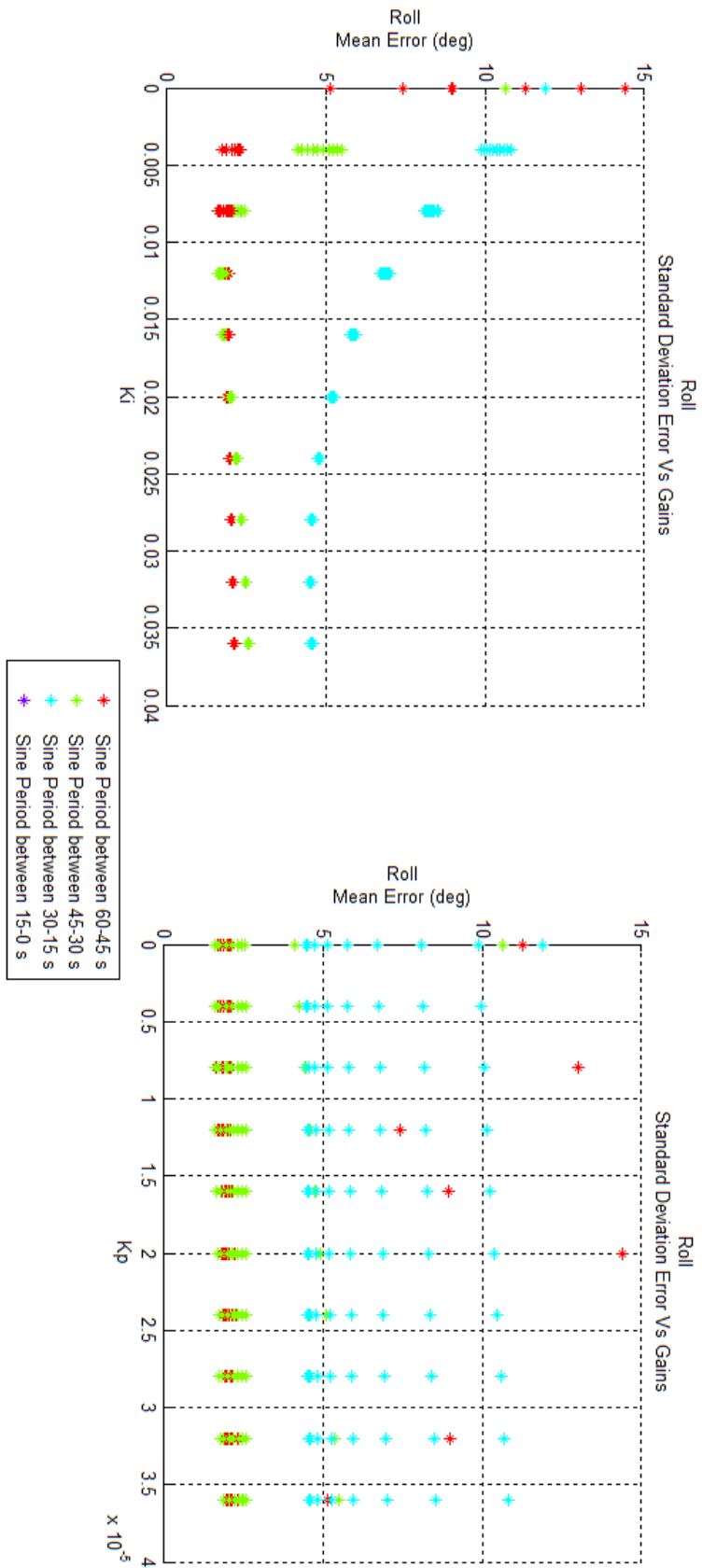


Figure 2.24: BPMS Vs potentiometer Roll standard deviation as a function of filter gains

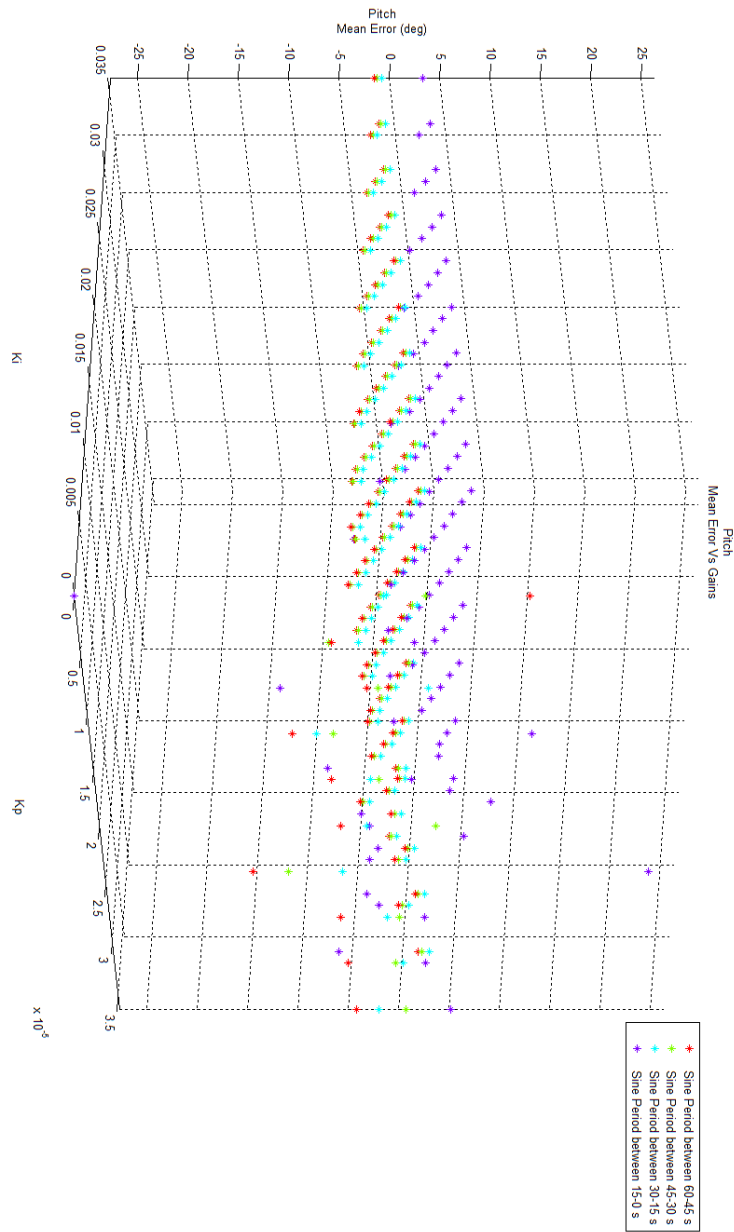


Figure 2.25: BPMS Vs potentiometer Pitch mean error as a function of filter gains

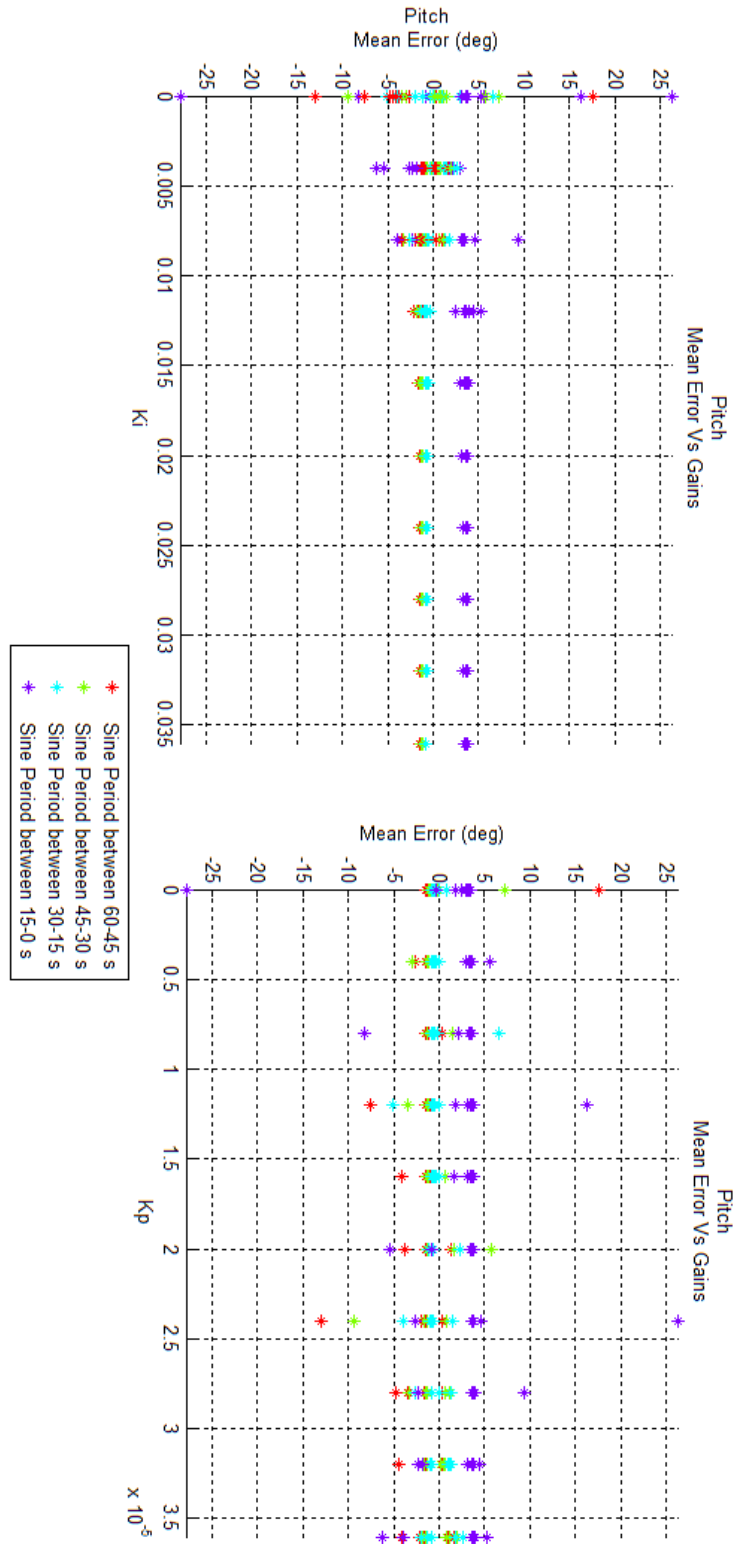


Figure 2.26: BPMS Vs potentiometer Pitch mean error as a function of filter gains side views

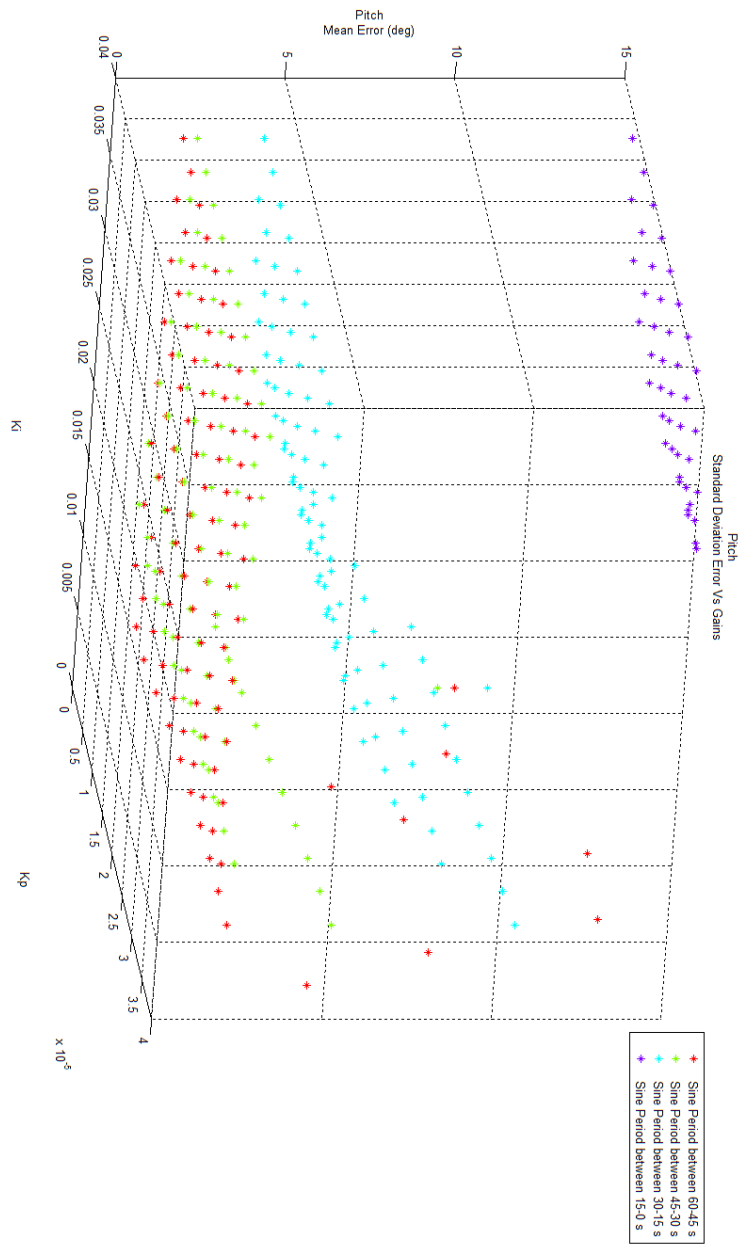


Figure 2.27: BPMS Vs potentiometer Pitch standard deviation as a function of filter gains

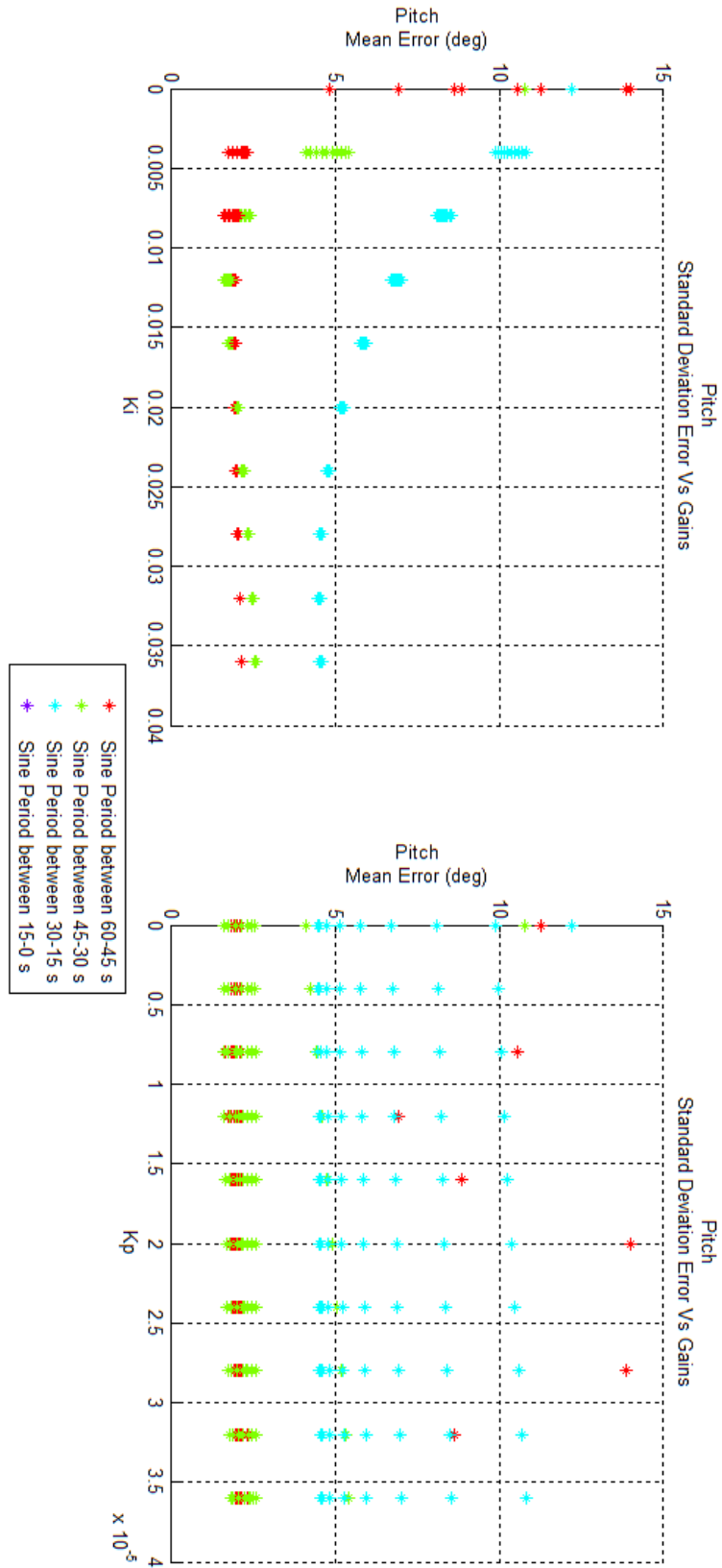


Figure 2.28: BPMS Vs potentiometer Pitch standard deviation as a function of filter gains

side views

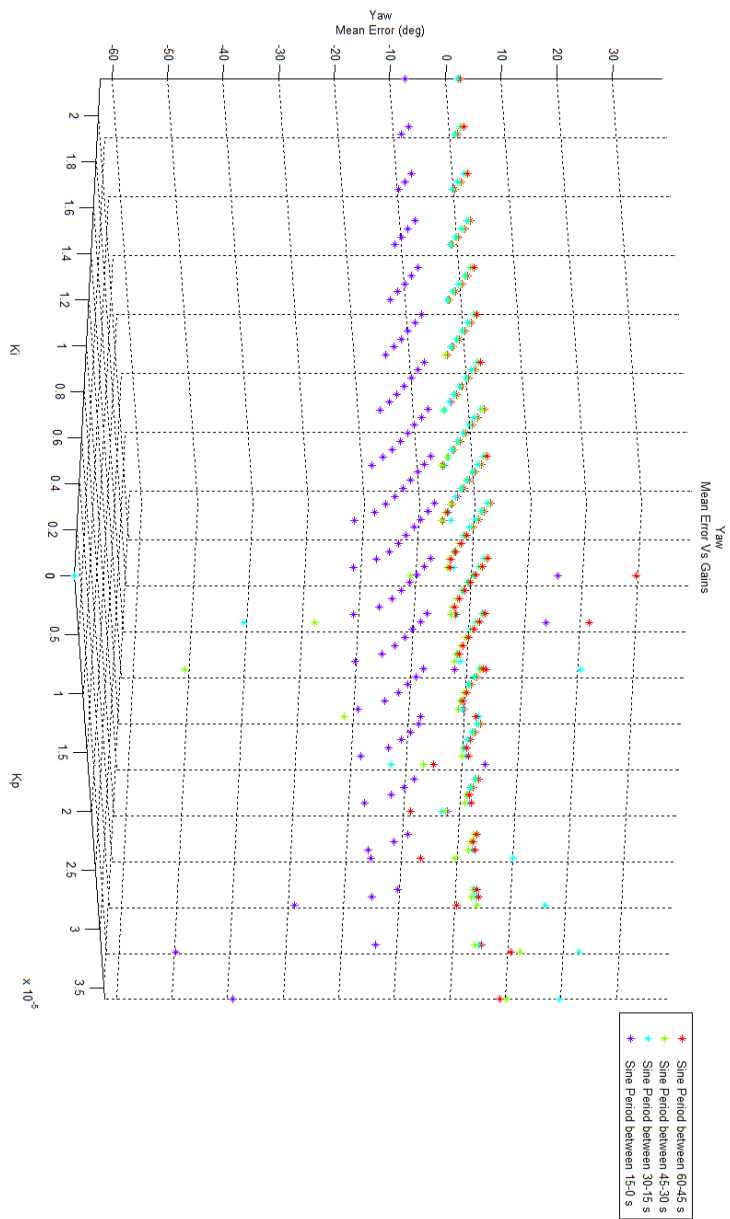


Figure 2.29: BPMS Vs potentiometer Yaw mean error as a function of filter gains

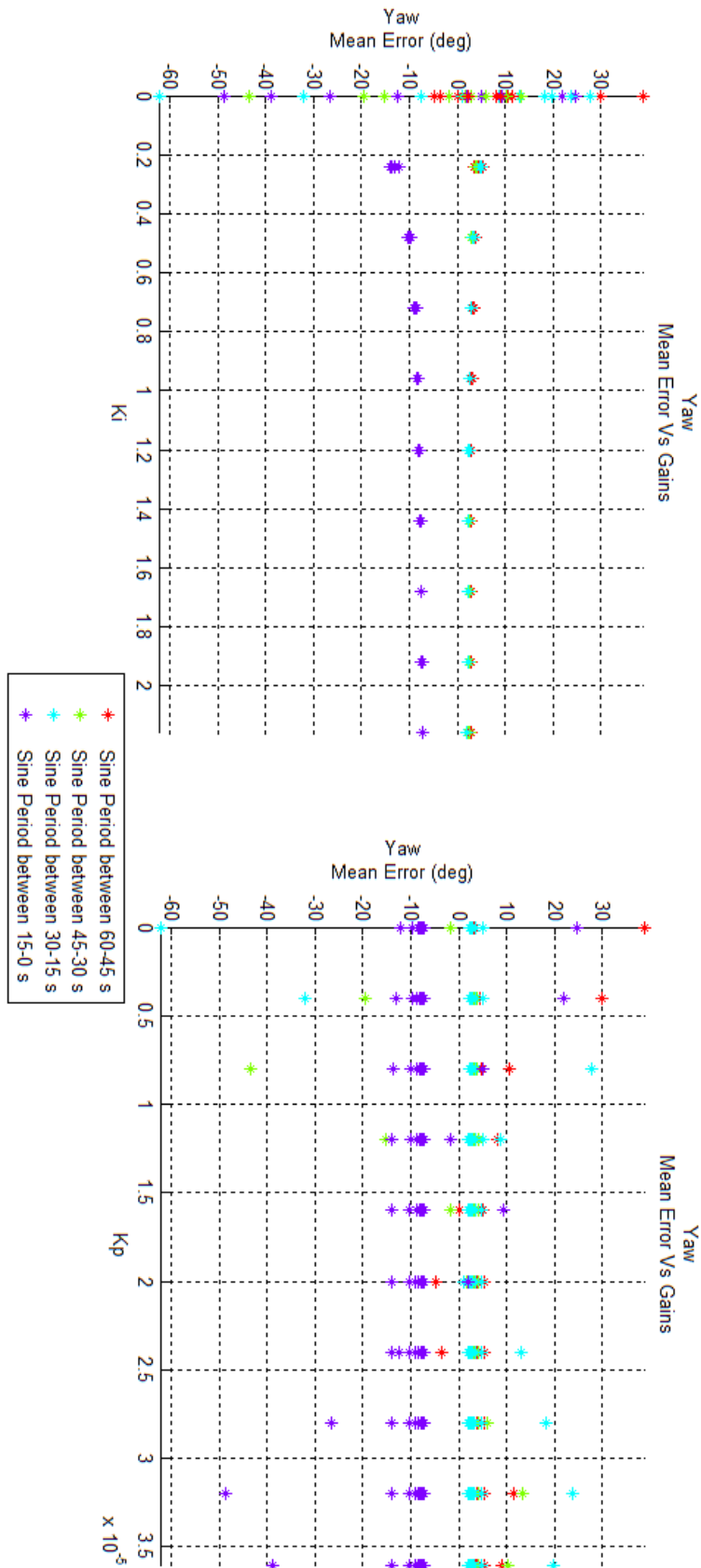


Figure 2.30: BPMS Vs potentiometer Yaw_{mean} error as a function of filter gains side

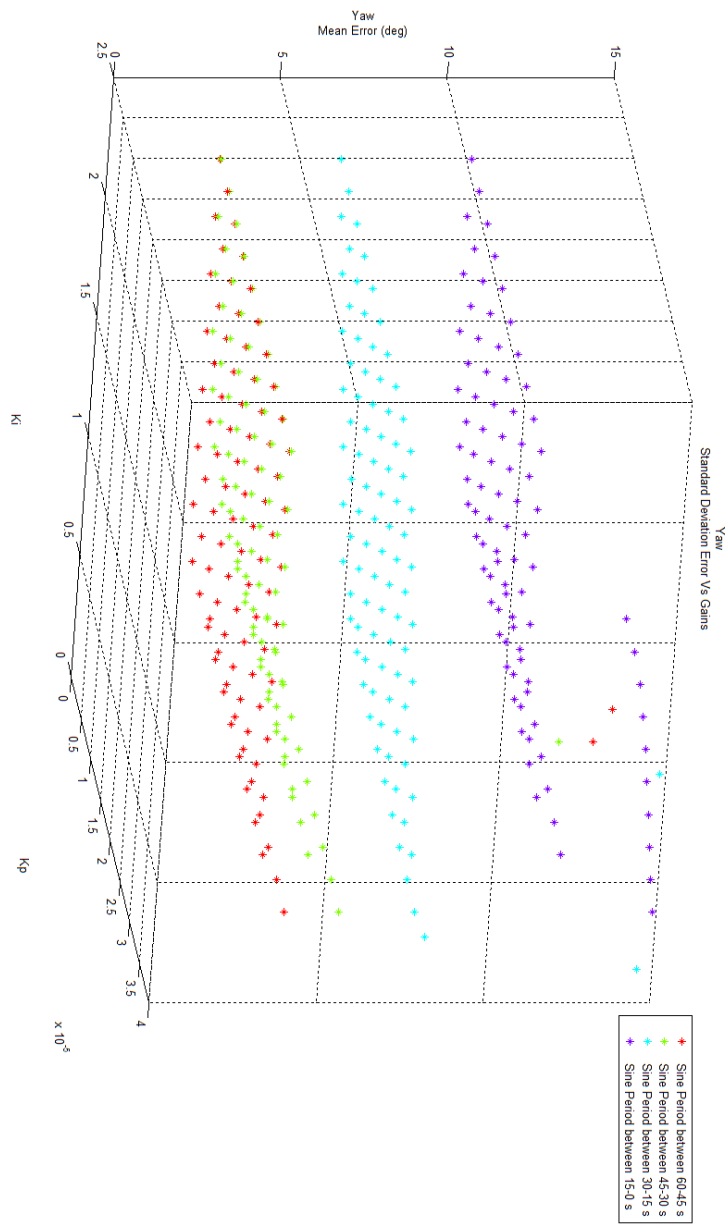


Figure 2.31: BPMS Vs potentiometer Yaw standard deviation as a function of filter gains

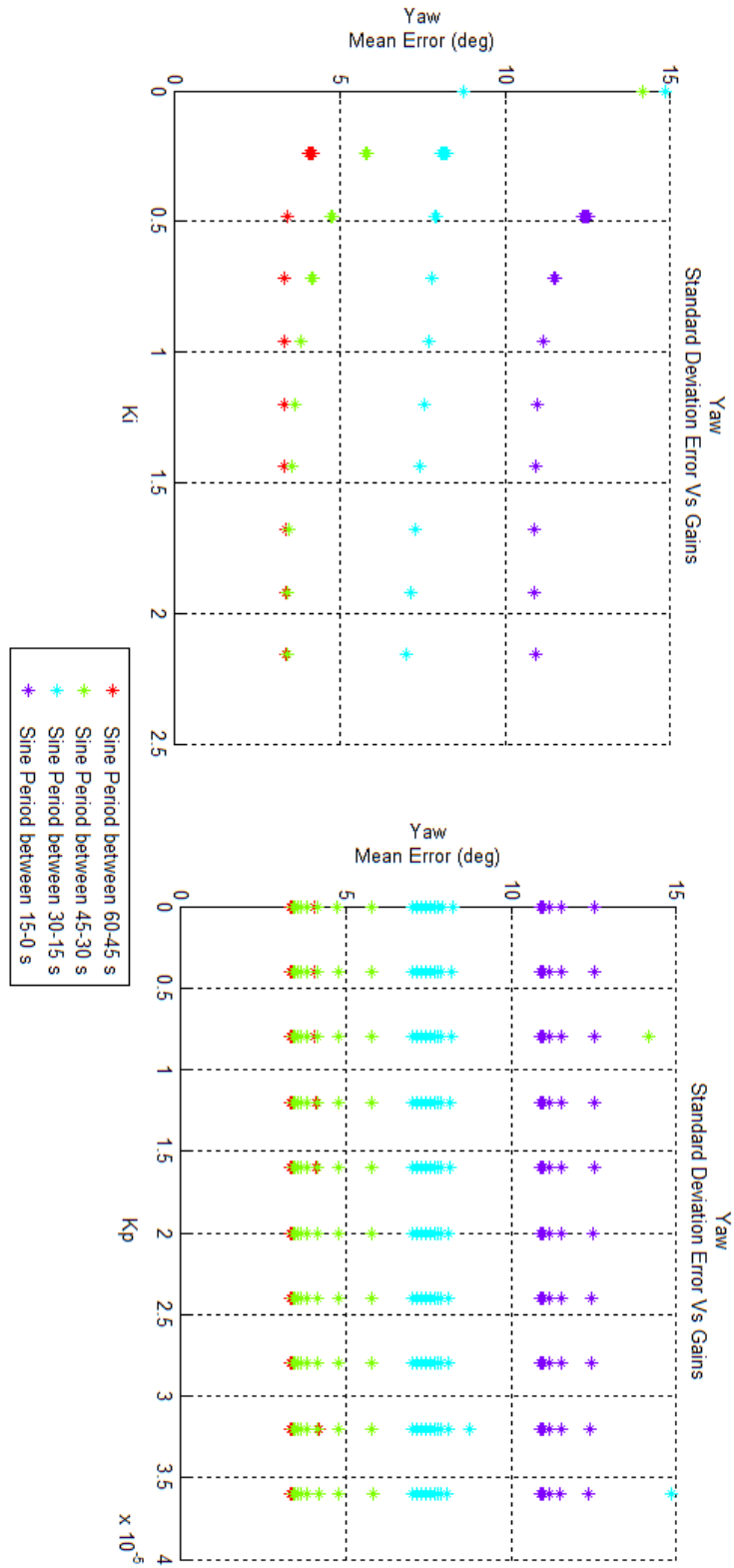


Figure 2.32: BPMS Vs potentiometer Yaw standard deviation as a function of filter gains

side views

From the plots above, we can see that the three systems return very comparable attitude estimates with a few differences to be noted. In this analysis we assume the potentiometer readings to be the ground truth. The confidence level on the linear potentiometer is very high due to the reliability and mechanical and electrical simplicity of the device, not to mention it's intensive use in the past for this scope. The average error on the angle measurement of for the potentiometer is below .1 of a degree for the range of the test with a sensitivity of .01 degrees . The main difference in the above data is between Pitch and Roll measurements errors and the Yaw errors. The error difference is not present with Vicon since the system evaluates the attitude in the three axis in the same way, while BPMS uses different sensors and different gains to reconstruct the estimates. The yaw estimate is corrected by using the magnetometer which is able to detect the magnetic field of the earth, unfortunately, the earth's magnetic field is very weak and small local magnetic perturbations can make the measurement even more complex. Nevertheless all three attitude estimates have errors and standard deviations comparable with the Vicon errors which demonstrated that the system is able to perform the task reliably. In terms of gains, from the data above, it can be seen that large errors are present for small gain values (especially K_i) while variations of K_p do not produce high variations in performance. As it can also be seen, there is not a single optimal gain across frequency bands, therefore a mean value has been chosen for the gains to accommodate for the first three bands which cover the majority of human motion speeds and frequencies. The plots below show the raw data for the chosen gains as well as the errors as a function of frequency for that specific case.

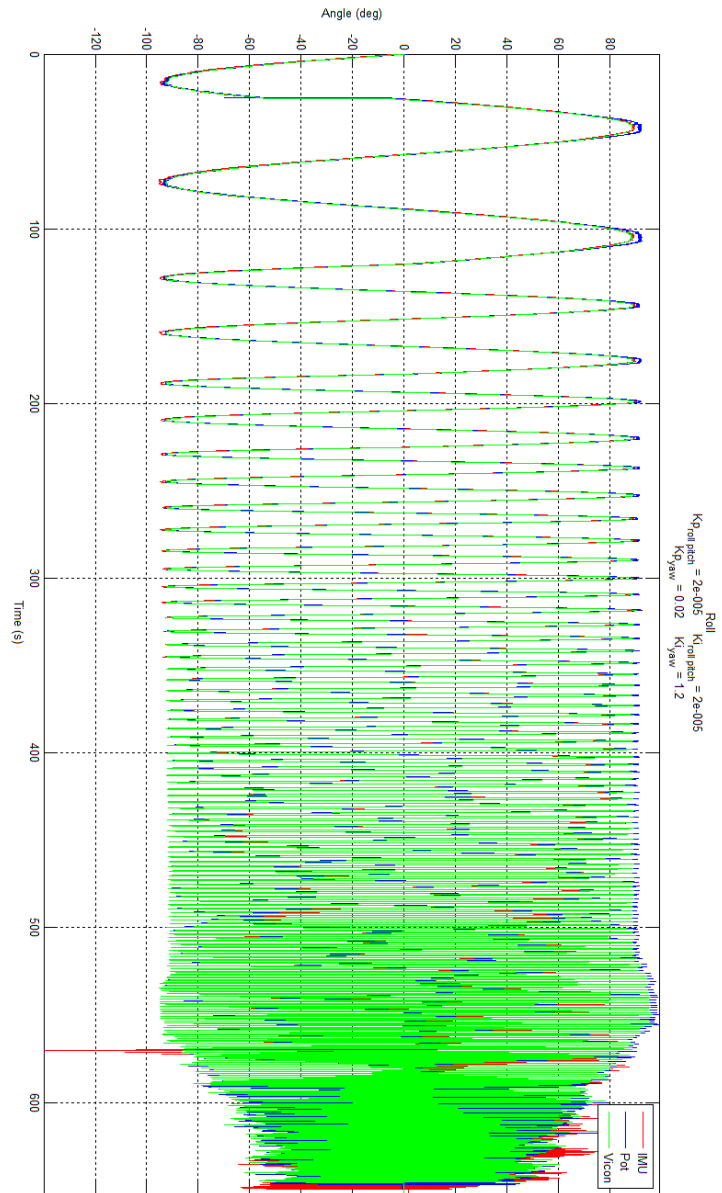


Figure 2.33: Roll measurement comparison between potentiometer, Vicon and BPMS

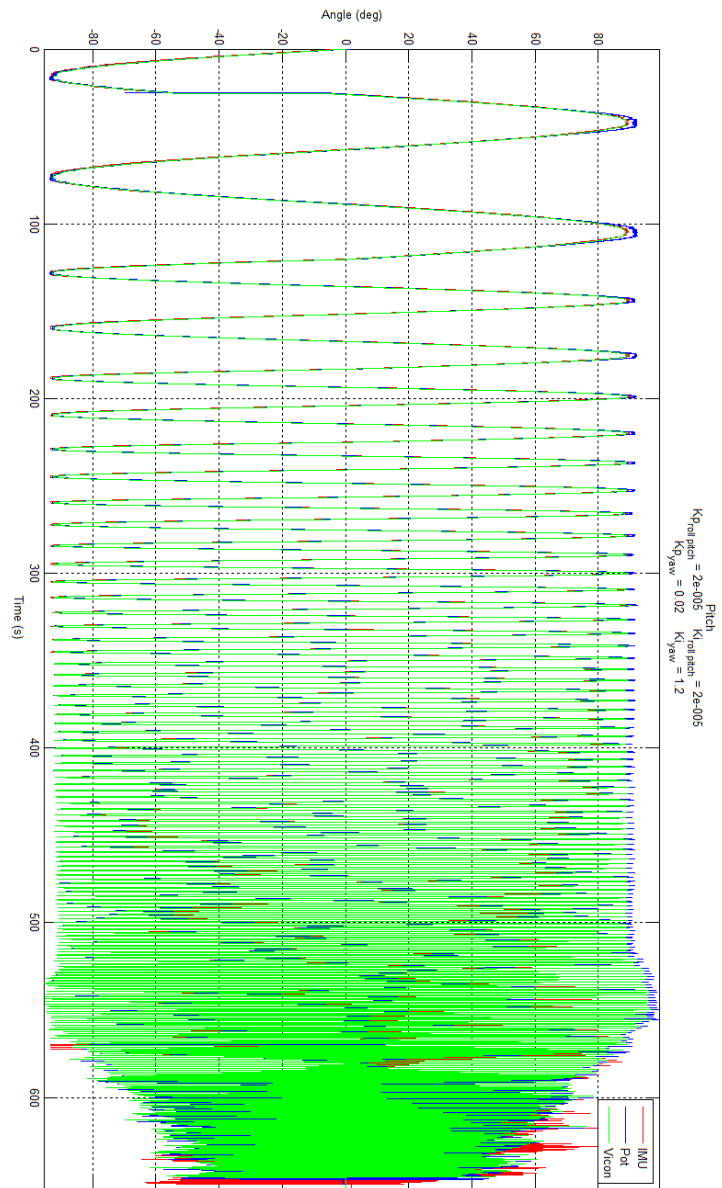


Figure 2.34: Pitch measurement comparison between potentiometer, Vicon and BPMS

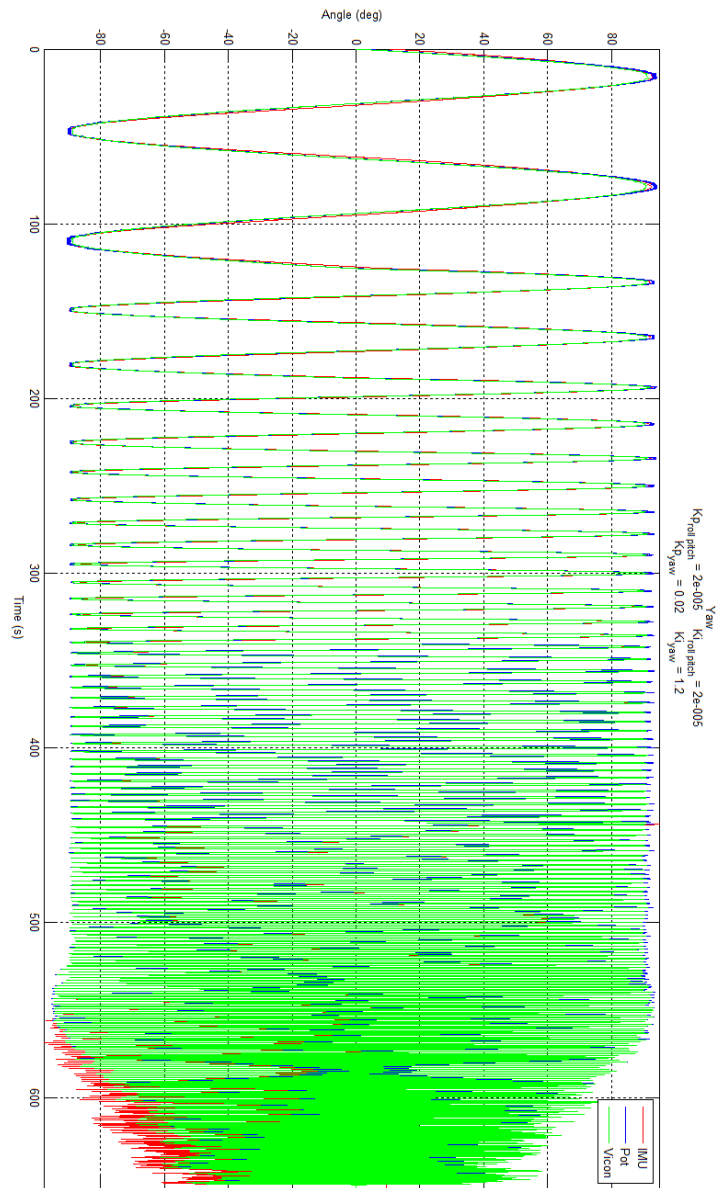


Figure 2.35: Yaw measurement comparison between potentiometer, Vicon and BPMS

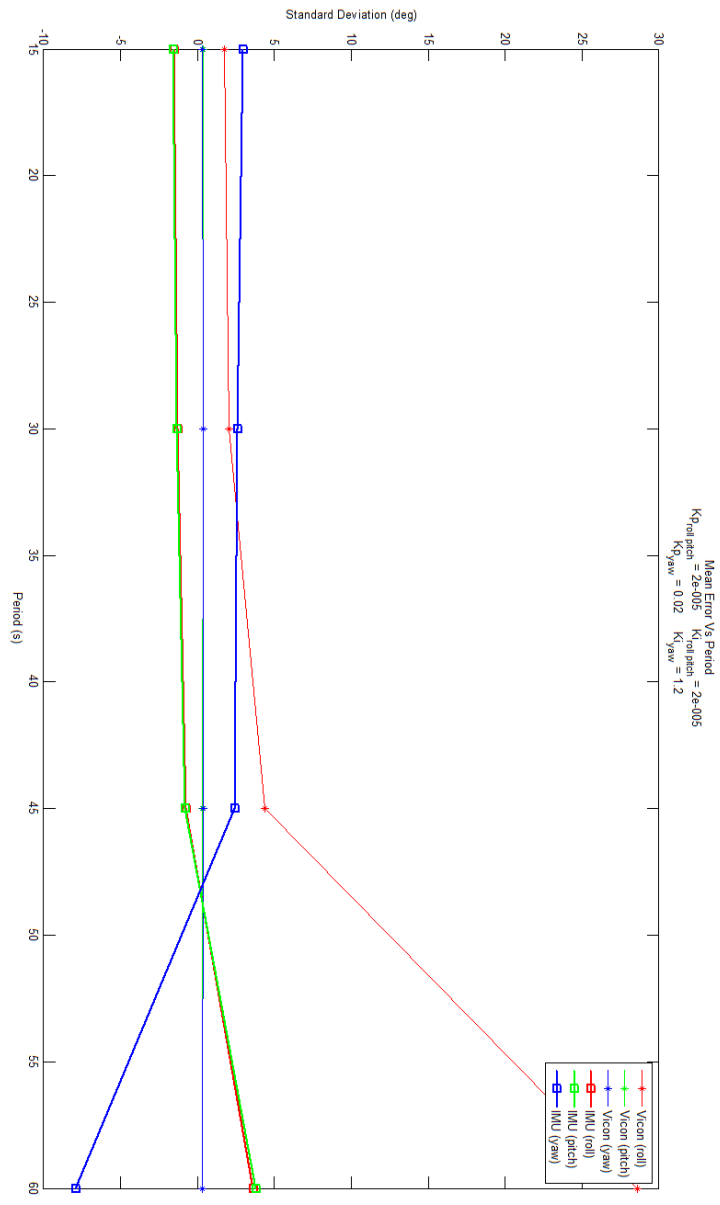


Figure 2.36: Error as a function of period

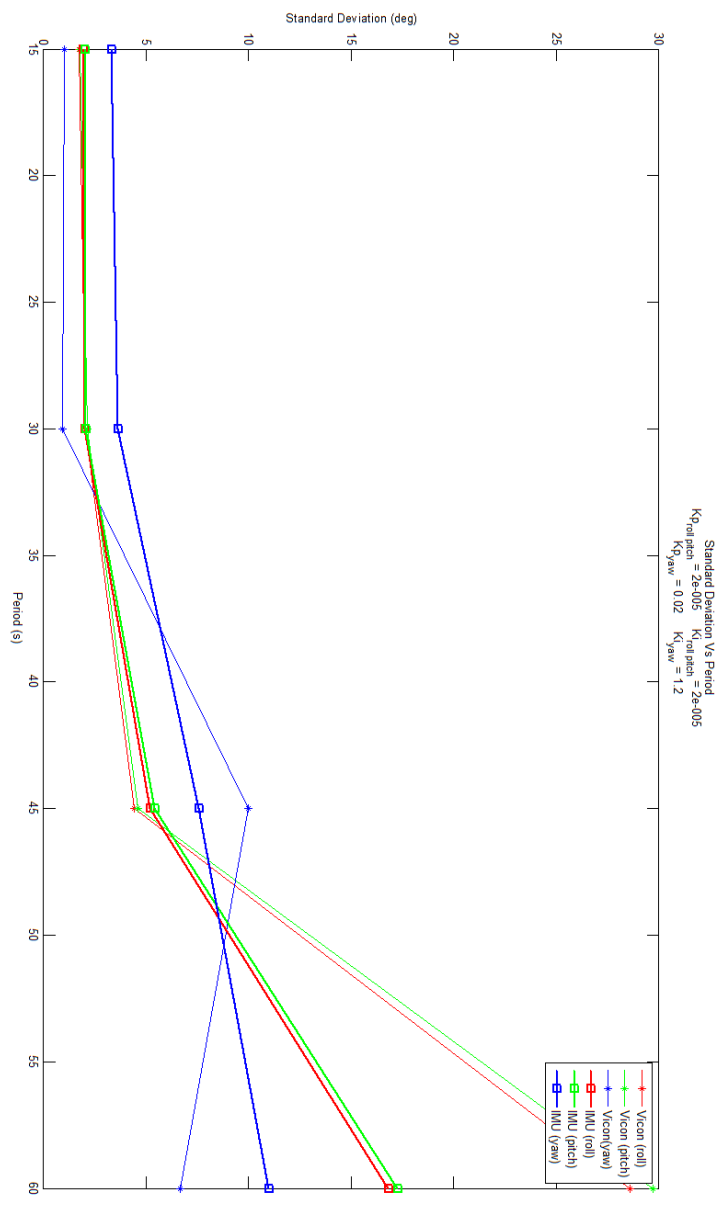


Figure 2.37: Standard deviation as a function of period

2.6.2.2 Position estimate performance

Estimating position from a system like BPMS is very different from measuring attitude. On it's own, the system doesn't provide the user with this piece of information and in order to do so, it requires the use of a base model. The simple working principle for obtaining position information requires the prior knowledge of the user's body segments lengths. If we define the human model as a serial chain of vectors applied each other at the end of the previous link, we can, for example evaluate the position of a extremity, in reference to a point we consider a reference point. Since the attitude of the body segments is given by BPMS, we can easily determine the position of the end of the vector as:

$$\vec{P}_i = \mathbf{R}_i \hat{P}_{0,i} l_i \quad i = 1, \dots, 18 \text{ (sensor ID)}$$

$$\hat{P}_{0,i} = \text{(body segment } i \text{ initial direction in the IMU reference system)} \quad (2.41)$$

$$l_i = \text{(body segment } i \text{ length)}$$

Unfortunately, though, it is immediately seen that in terms of accuracy of the measurements, for a single estimate, the attitude error will be increased as a function of the length of the body segment and when we consider the entire serial chain, the error will add up as we go through the chain. From the previous results on tracking accuracy, we know that the average error is $e_{i,j} \approx \pm 2$ degrees on each angle, therefore, theoretically we will have position errors bounded within:

$$e_{P(i,j)} = \pm l_i \sin(e_{i,j}) \approx \pm 0.034 l_i \quad (2.42)$$

$$j = x, y, z \text{ (axis)}$$

where we can see that the average position estimate is plagued by a $\pm 3.4\%$ error of the length of the segment. As previously stated, as we go forward in the vector chain,

Table 2.3: BPMS theoretical axial position errors on 50th percentile American male

Extremity	chain components	error(cm)
Head	low,mid spine, neck	2.3
Hands	low,mid spine, shoulder, arm, forearm	4.25
Legs	hip, leg, shank	3.4

those errors add up therefore for the extremities like the hands, head and the feet we will have errors in the ranges shown in the table above.

where we have assumed body segments lengths based on the 50th percentile American male and set the reference on the hip (obtained from Dassault Systems Cata V5 Human builder models that references the NHTSA (National Highway Traffic Safety Administration) databases):

From this preliminary analysis, we have evaluated the theoretical maximum errors that we can expect on the position of the extremities of the body and as expected position estimates are plagued by an error proportional to the size of the human wearing the system. This is an expected results since position is a derived property for BPMS. If we were to compare Vicon, we can see that the the two systems have opposite properties. Vicon measures position directly and therefore attitude as a derived parameter is plagued by a greater error as shown in the previous evaluations, BPMS shows the same characteristics but in the opposite measurements.

Table 2.4: 50th percentile American male body segments lengths source:NHTSA

Chain components	length(m)
low spine	0.21
mid spine	0.27
neck	0.20
head	0.22
shoulder	0.16
arm	0.34
fore arm	0.27
hand	0.19
hip (half)	0.17
Leg	0.43
shank	0.40
foot	0.27
tot. height	1.73

2.6.2.3 Comparison between VICON and the BPMS suit attitude estimates

During the ILC Dover test runs it was possible to simultaneously record both BPMS and Vicon data. The BPMS suit was equipped with a series of retroreflective markers that were used to track the attitude of all body joints. This section provides the results from the comparison between Vicon and BPMS tracking abilities and demonstrates the ability of BPMS to produce comparable measurements. The plots below show a sample of the data acquired and the resulting estimates errors.

In the data above we can see the same behaviors of both systems as compared to the test bed data. This extension to the test bed experimental setup demonstrates that BPMS can actually acquire attitude information with performance close to Vicon.

2.7 Ancillary systems

During the initial testing phase BPMS performed as desired but it was soon recognized that the system usability could be greatly improved by implementing a series of additional features that would expand its capabilities. First among them, the ability to record BPMS data without a full size computer or laptop in the vicinity. The Bluetooth wireless system as previously stated grants data exchange to ranges up to 300m in line of sight, but most "in-the-field" operations usually occur much larger ranges (from a few hundred meters to a few kilometers). It was immediately recognized that a fundamental feature to include to BPMS was the ability to record the sensor data locally. Two strategies were envisioned. The main BPMS control board was initially designed to be able to function in both a wired and wireless configuration. To do so, the main serial output was broken out on the PCB. The serial link breakout along with the radio power jumper allows the investigators to disable the Bluetooth wireless transmitter and add a serial data

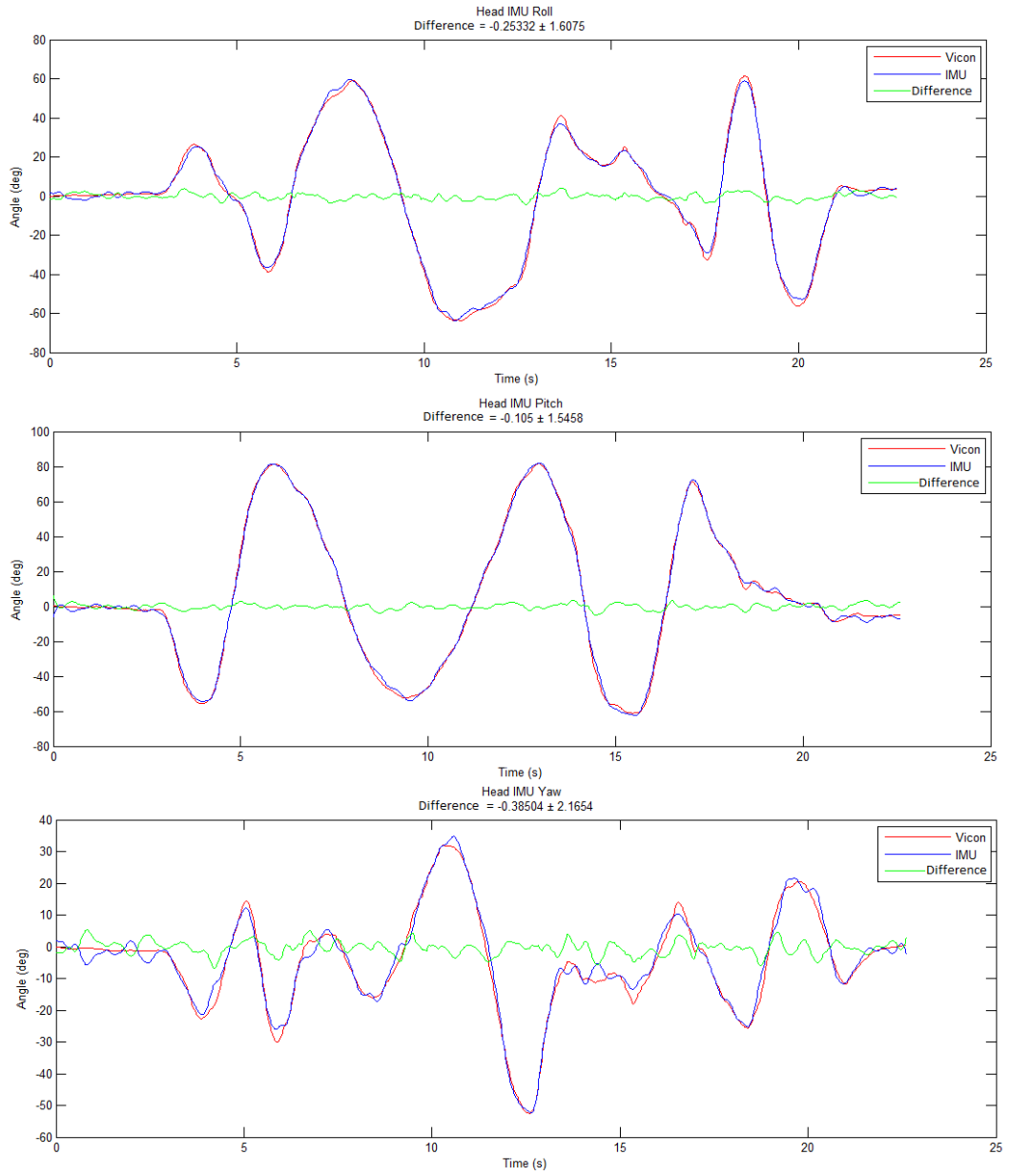


Figure 2.38: Tracking of the Head comparison between Vicon and BPMS

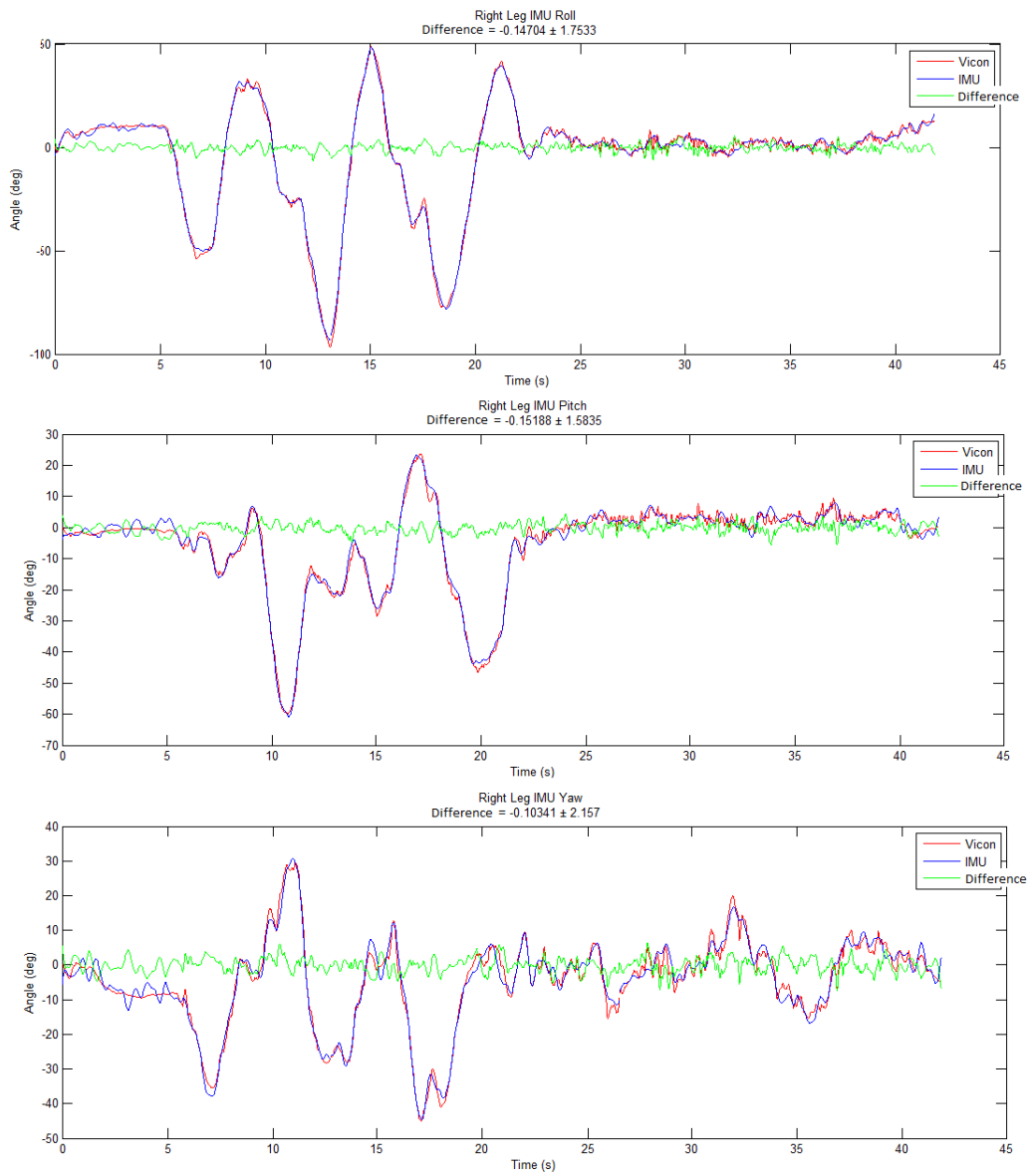


Figure 2.39: Tracking of the Right Leg comparison between Vicon and BPMS

logger. While this approach is perfectly valid in principle, two things made it unfeasible while in the field. First of all, at the time of the test, the only data logger available was not able to handle a reliable recording of the serial stream at the data rates at which BPMS operates (2Mbps), secondly, the initiation of the data stream would have to be triggered somehow and such a feature would have required either a modification on the BPMS main microcontroller software, or the addition of a hardware button. The combination of the above factors made it such that a different solution was necessary at that particular point in time. The second strategy envisioned no modifications at all to the BPMS system. Android based phones and PDAs are widely available and most of them provide high-end Bluetooth capabilities and high available computational power.

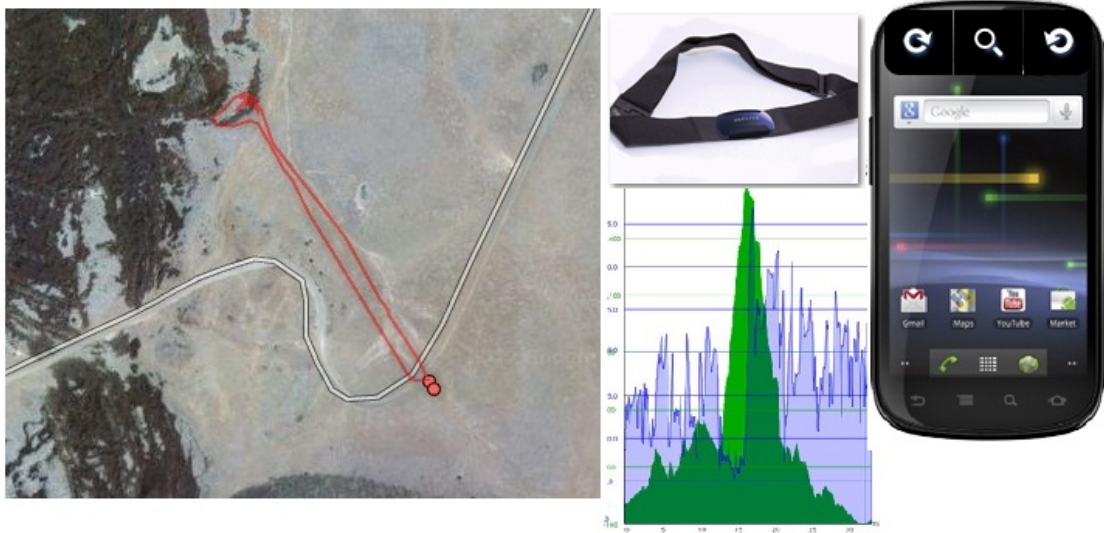


Figure 2.40: BPMS ancillary systems (Zephyr HXM Bluetooth heart rate sensor and Nexus S)

At the time of the test the investigators had a Samsung NEXUS S android phone at their disposal. The device when paired with the SL4A (Scripting Layer For Android)

libraries enabled the investigators to write a simple script in P4A (Python For Android) that allowed the device to pair with BPMS, initiate data transmission and record the data output in a format that is compatible with the main BPMS software package. All the recording occurs on the device's on-board flash memory and its later retrieved. Since the field trials, several modifications to the script have been made and several useful features were added. As of the writing of this work, the android based app, allows the user to initiate, pause, stop and resume recordings through a simple GUI (Graphical User Interface). The app also allows the investigator to control and monitor the status of the recording over WiFi or 3G networks, and additionally allows recording of GPS tracks (by using the device's GPS receiver antenna) and the user heart rate, by simultaneously connecting to a Zephyr HXM Bluetooth heart rate sensor. All of the above recordings (Heart rate, BPMS and GPS traks) can occur separately or simultaneously. In both cases three different files are produced and the trial duration time is exclusively limited by two factors: both system's battery endurance, and available storage space on the android device. By allowing for this local recording scheme, BPMS's range had now become "unlimited" enabling it to be deployed in any field trials without worrying about bulky overhead equipment, and eliminating the need for investigators to follow the test subjects at relatively close range, hence allowing for much more realistic experiments.

Chapter 3

Experimental evaluations

This chapter will cover the experimental portion of this work, where the newly available capabilities of BPMS are used to provide the investigators a new perspective on pressure suit analysis. As seen so far, BPMS is able to return the attitude of each sensor; those measurements can then be mapped onto a simplified human model. The human model so identified resides in a virtual environment and is unaware of the physical limitations of a real human. Each body segment is represented in vectorial form (magnitude and direction) and the only constraints imposed, are the positions of those segments. As an example, let's assume we can measure exclusively the pose of an arm. In order to represent it in our 3D model, we use two cylinders representing the arm and the forearm while we use a prism to represent the hand. From the respective IMUs we identify the direction of each cylinder and the prism and, we define their magnitude (length) by measuring the human arm that we are attempting to model. So far we have the orientation and magnitude of those objects but in order to reproduce the human limb, we must apply them in the right locations. This can be achieved by applying the arm segment at the origin, while the forearm is applied at the end of the arm vector and the hand in the same way is applied at the end of the forearm vector. In this way we can define the states of each segment in the synthetic environment. As previously stated, this approach grants us a visual representation, in the synthetic world, of the pose of the test subject, but, as is, this model doesn't tell us directly the human body angles for each human joint. In order to evaluate this fundamental information a kinematic constrained problem must be defined

and solved. The first portion of this chapter will focus on the definition and the solution of the mapping problem. The results will then be used to describe the range of motion in three different pressure suits under different pressure environments. Finally a last experimental session was conducted during the last desert field trials where a field geologist was asked to conduct a simulated geological survey sortie. During this session the test subject was instrumented with BPMS and with a GPS logger and by using the acquired GPS track as a ground truth, we have attempted to reconstruct the same followed path by using the data from BPMS and compared the two results. This chapter is structured to present each experimental session separately by introducing first the methodology, followed by the data reduction and finally present a discussion on the results.

3.1 Converting IMU attitude to body joint angles

In chapter 1, we have introduced the key body angles and their nomenclature as well as the human skeletal system. In this portion of the work we will provide a detailed description of the IMU attitude conversion to body angles starting from the filter output rotation frame described in the inertial frame. From the complimentary filter we obtain at each time step an updated DCM of each IMU represented by $\mathbf{R}_i(\mathbf{t})$ where i represents the IMU id ($i = 0, \dots, 17$). This rotation matrix describes the attitude of the sensor in the earth magnetic reference frame obtained from a 3-2-1 transformation defined by the angles $([\gamma_i(Yaw), \beta_i(Pitch), \alpha_i(Roll)])$. In order to map the arbitrary initial IMU orientations on the simplified human model, an initial offset matrix is acquired $\mathbf{R}_{c,i}$. In order to acquire this matrix, the test subject is asked to stand in the calibration pose and the current IMU attitudes are sampled $\mathbf{R}_i(\mathbf{t}_c)$. From this sample, we proceed in the evaluation of the offset matrix $\mathbf{R}_{c,i}$ as follows:

$$\mathbf{R}_{c,i} = \mathbf{R}_i(t_c)^{-1} = \mathbf{R}_i(t_c)^T \quad (3.1)$$

By taking the transpose of the current attitude of the sensors, we define a new reference frame that identifies all the attitude of the sensors in the neutral configuration ($([\gamma_i(Yaw), \beta_i(Pitch), \alpha_i(Roll)] = [0, 0, 0])$). In order to do so, we define the new sensor attitude matrix $\mathbf{R}_i^*(t)$ as:

$$\mathbf{R}_i^*(t) = \mathbf{R}_{c,i} * \mathbf{R}_i(t) \quad (3.2)$$

This matrix represents the attitude of the sensors calibrated for an arbitrary initial pose but it is not complete yet. The sensor attitudes are now defined in the reference frame defined by the initial magnetic orientation and to return in the Earth magnetic reference frame, an additional rotation is required. In order to define the body model magnetic heading we need a calibrate against a single sensor. Canonically the reference element of choice is the hip which will also be used here to define the magnetic bearing of the human body model. From the discussion above, we can now define the new sensor attitude that is both corrected for arbitrary initial attitude and bearing defined in the earth magnetic reference frame $\mathbf{R}_{m,i}^*$.

$$\mathbf{R}_{m,i}^*(t) = \mathbf{R}_{\gamma_{Hip}} \mathbf{R}_i^*(t) \quad (3.3)$$

where :

$$\mathbf{R}_{\gamma_{Hip}} = \begin{bmatrix} \cos(\gamma_{Hip}) & -\sin(\gamma_{Hip}) & 0 \\ \sin(\gamma_{Hip}) & \cos(\gamma_{Hip}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

Which is a simple rotation around the local vertical of γ_{Hip} which is obtained from $\mathbf{R}_6(t_c)$ (Hip id = 6) as follows:

$$\gamma_{Hip} = -\sin^{-1}(r_{6,13})(t_c) \quad (3.5)$$

At this point, the last step to take, is to convert the calibrated sensor attitudes in body angles. TO do so, we can divide the human body joints in two categories; The first includes all the single degree of freedom joints such as the elbows and knees, while the second includes all the two or more degrees of freedom joints. A different approach is taken for the two groups of joints. For the first group, we evaluate the flexion/extension angles α_i , in the following way. First of all in the simplified model, we define the arm, forearm, leg and knee through a vector $V_{0,i}$. For all those sections of the body, we can evaluate the current pose of those vectors $V_i(t)$ as:

$$V_i(t) = \mathbf{R}_{m,i}^*(t)V_{0,i} \quad (3.6)$$

where $V_{0,i}$ for the first group joints are defined as (in the same way for both the right and left side):

$$\begin{aligned} V_{0,arm} &= [0, 0, -1]^T * L_{arm} \\ V_{0,forearm} &= [0, 0, -1]^T * L_{forearm} \\ V_{0,leg} &= [0, 0, -1]^T * L_{leg} \\ V_{0,shank} &= [0, 0, -1]^T * L_{shank} \end{aligned} \quad (3.7)$$

Finally, we can evaluate the flexion/extension angles α_i as follows:

$$\alpha_{knee} = \cos^{-1}\left(\frac{V_{leg}^{\vec{}}(t)V_{shank}^{\vec{}}(t)}{L_{leg}L_{shank}}\right)$$

$$\alpha_{elbow} = \cos^{-1}\left(\frac{V_{arm}^{\vec{}}(t)V_{forearm}^{\vec{}}(t)}{L_{arm}L_{forearm}}\right)$$
(3.8)

The second group of body joints needs a brief additional introduction. Human motion is defined in reference to the three main body planes. The body planes are here summarized in figure 3.1.

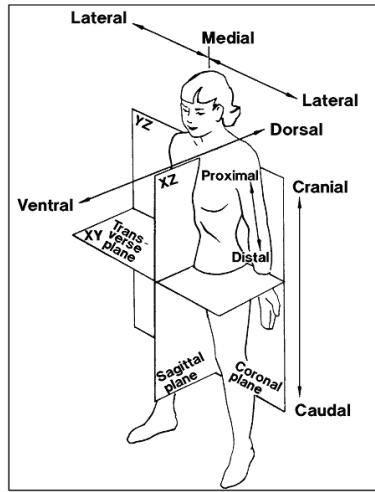


Figure 3.1: Human body planes of reference Source:NASA STD-3000

As we can see, referring all IMU attitudes in relation to the hip, brings it's advantages in this definition. Since the remaining joints such as the shoulders, hip, ankles, wrists torso and head, all are defined in the canonical body reference frame, all we need to do is evaluate the euler angles for each sensor and subtract the euler angles of the previous joint. Once this is done, all we need is a small change in nomenclature and the IMU angles have all been translated to the canonical set of body angles and the following tables summarize the process:

Table 3.1: IMU angles to body angles for multi-DOF joint

Body Joint	Joint motion	R angles	Dynamic Segment	Static Segment
Shoulder	adduction/abduction	Roll	Arm	Shoulder
	flexion/extension	Pitch	Arm	Shoulder
	lateral/medial rotation	Yaw	Arm	Shoulder
Hip	adduction/abduction	Roll	Leg	Hip
	flexion/extension	Pitch	Leg	Hip
	lateral/medial rotation	Yaw	Leg	Hip
Ankle	adduction/abduction	Roll	Foot	Shank
	flexion/extension	Pitch	Foot	Shank
	lateral/medial rotation	Yaw	Foot	Shank
Torso	adduction/abduction	Roll	Mid Spine	Hip
	flexion/extension	Pitch	Mid Spine	Hip
	lateral/medial rotation	Yaw	Mid Spine	Hip
Head	adduction/abduction	Roll	Head	Neck
	flexion/extension	Pitch	Head	Neck
	lateral/medial rotation	Yaw	Head	Neck

For all the above evaluations, the euler angles are resolved as follows:

$$\begin{aligned}
\psi &= \tan^{-1}(r_{12}/r_{11}) = \tan^{-1}\left(\frac{\cos(\theta)\sin(\psi)}{\cos(\theta)\cos(\psi)}\right) \text{ (Yaw)} \\
\theta &= -\sin^{-1}(r_{13}) = -\sin^{-1}(-\sin(\theta)) \text{ (Pitch)} \\
\phi &= \tan^{-1}(r_{23}/r_{33}) = \tan^{-1}\left(\frac{\sin(\phi)\cos(\theta)}{\cos(\phi)\cos(\theta)}\right) \text{ (Roll)}
\end{aligned} \tag{3.9}$$

Where the matrix \mathbf{R} is obtained from the two calibrated DCMs of the static and dynamic joints as follows:

$$\mathbf{R} = \mathbf{R}_{m,istatic}^{*T}(t)\mathbf{R}_{m,idynamic}^*(t) \tag{3.10}$$

3.2 Pressure suits

In the following evaluations, three different pressure suits have been used. This section will describe the three systems and will be crucial to better understanding the following parts of this work. Comparing the results across the three systems is highly discouraging since, as it will clearly be described, the three suits were designed for three very different operational environments and uses. Besides this word of caution, the three systems were subject to the same test cases and environments whenever possible.

3.2.1 David Clark prototype testbed suit

The prototype test bed suit is a high fidelity space suit mock-up that serves as a primary testbed for evaluating new joint technologies at the David Clark company. This particular system presents a very asymmetric configuration where the left and right side of the suit are equipped with different mobility joints. In the adopted configuration, the suit presented different shoulder, arms and knee joints and made use of the CHAPS suit gloves. The prototype testbed suit uses the standard dual layer setup (gas container and restraint layer) with a link-net torso and a non conformal helmet. In the evaluations with BPMS, no TMG or protective garments were used and the system was evaluated unpressurized and at 1 and 3 psid.

3.2.2 David Clark CHAPS (Contingency Hypobaric Astronaut Protective Suit)

The CHAPS suit is David Clark's contingency hypobaric suit currently used by the US military for their U2 pilots. Mainly designed for contingency scenarios, it ensures high upper body mobility when pressurized. The suit has a bias zipper on the front center of the torso that allows it to be pressurized in the seated position. The suit uses



Figure 3.2: David Clark prototype testbed suit

a single wall laminate style architecture with limited bearings and hard elements and a conformal helmet. The CHAPS system is very light weight and comfortable to wear when not pressurized. The system was evaluated; unpressurized and at 1 psid.

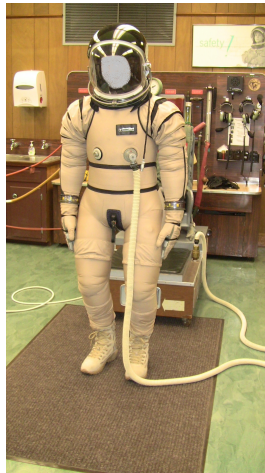


Figure 3.3: David Clark CHAPS suit

3.2.3 ILC Dover Launch and reentry I-Suit

The ILC Dover launch and reentry variant of the I-Suit is a prototype suit is a high pressure zero pre-breath suit that incorporates high mobility joints on both upper and lower body. This particular suit was initially envisioned to allow for modular segments to be exchanged and enable it to be used in multiple operational scenarios. In the following tests the suit was tested in both presence and absence of the high visibility protective garment which was removed during the vicon trials due to unreliable anchoring of the markers. The suit uses a dual layer architecture (bladder and restraint), a non conformal helmet and was evaluated; unpressurized and at 4.3 psid.



Figure 3.4: ILC Dover Launch and reentry I-Suit

3.3 Kinematic evaluation of three pressure suits: Range of motion

In this experimental sessions, the three previously described pressure suits were evaluated by using BPMS to measure the test subject's kinematics while inside the suit. Two experienced suit subjects volunteered for the evaluations, one for each company. Each session begun with the subjects running all the test cases without any suit on (just wearing BPMS) in order to define the natural unobstructed range of motion, and then repeated the test cases with the suits on at various pressures ranging from unpressurized to 4.3 psid (based on the suit). Eight test cases encompass the range of motion evaluation. For each test case, the test subjects were asked to move the specified joint in all direction for as far as they felt comfortable. The eight test cases are as follows:

3.3.1 Data plots

The following section provides the reader with the experimental results from the range of motion evaluation:

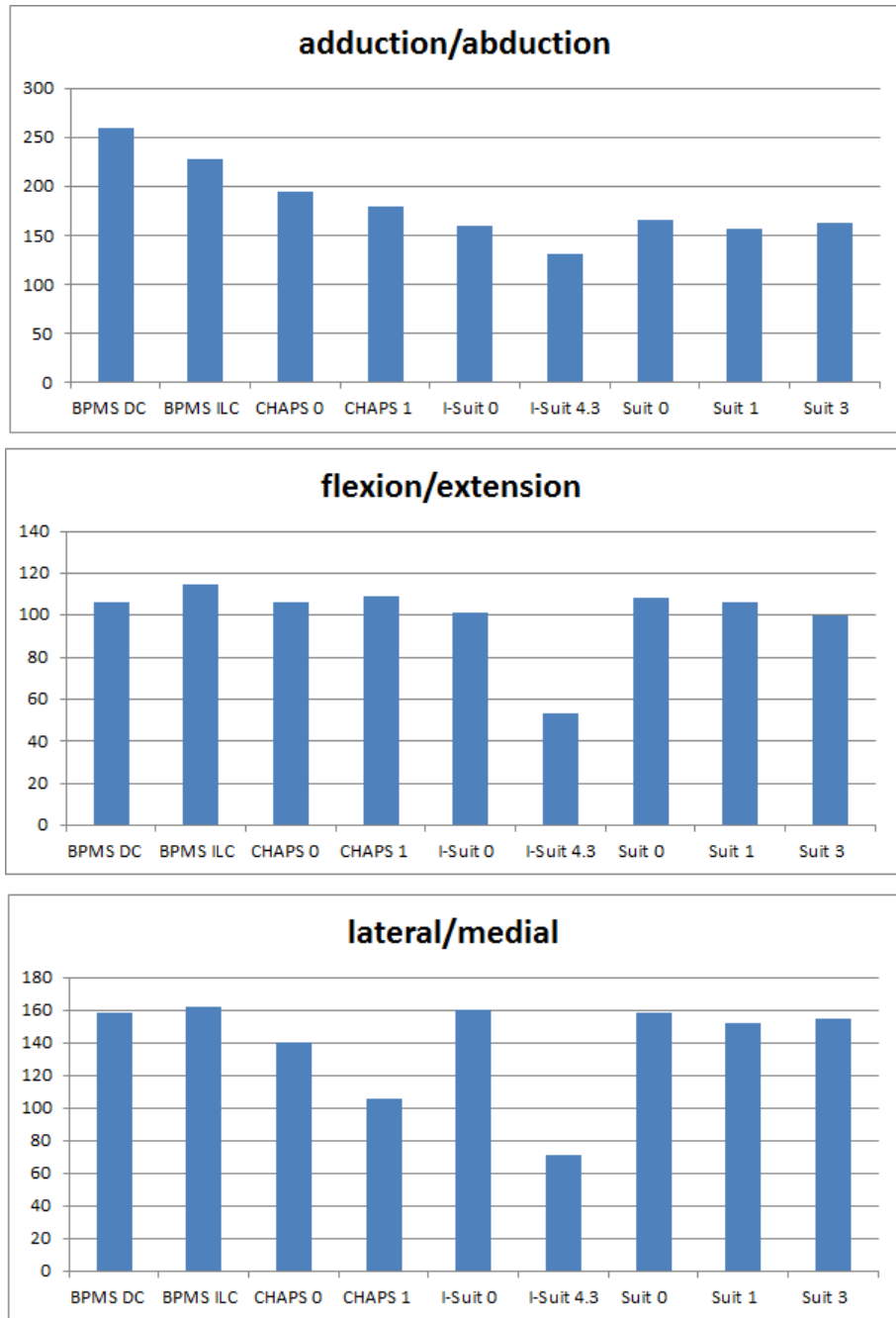


Figure 3.5: Right Shoulder ROM

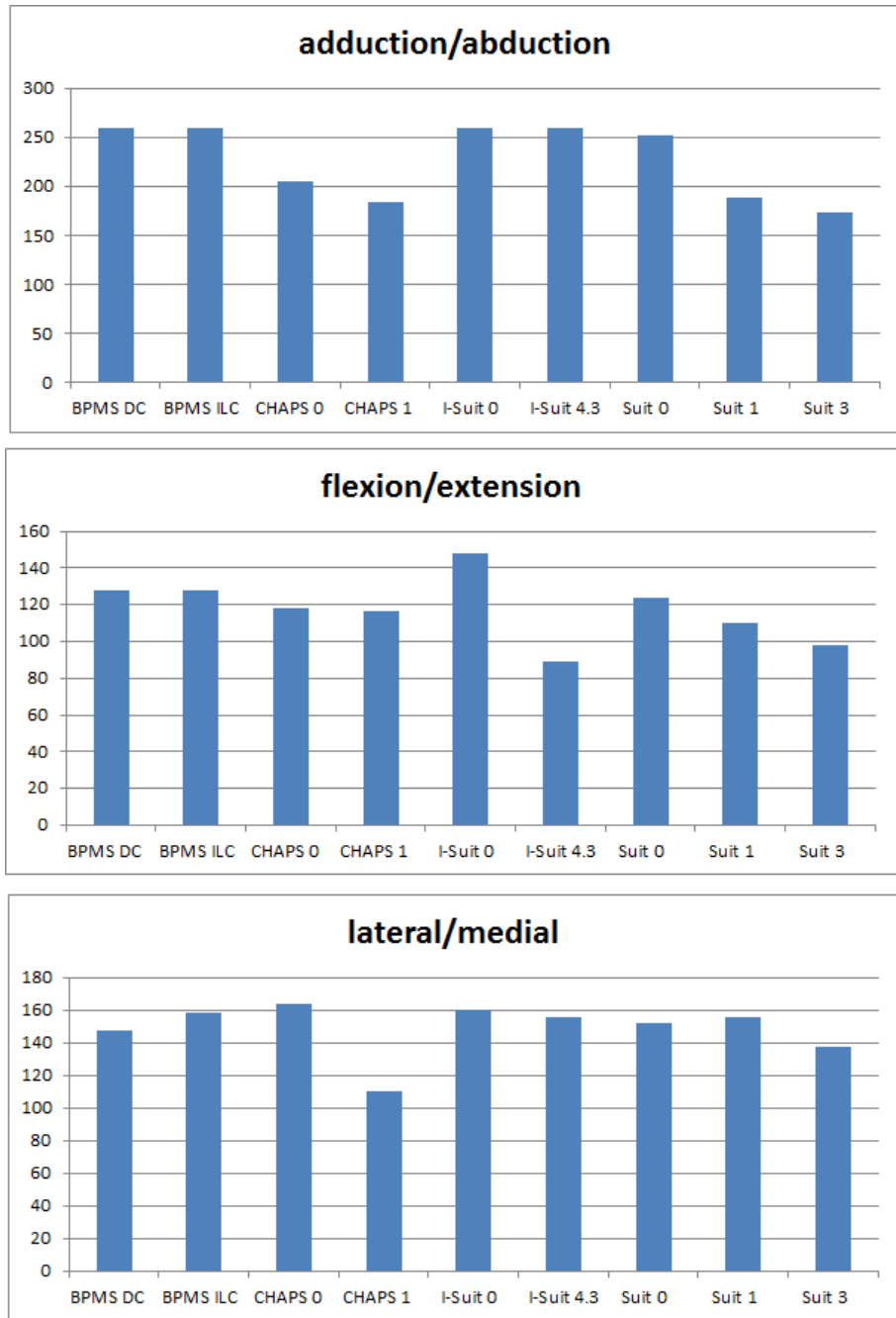


Figure 3.6: Left Shoulder ROM

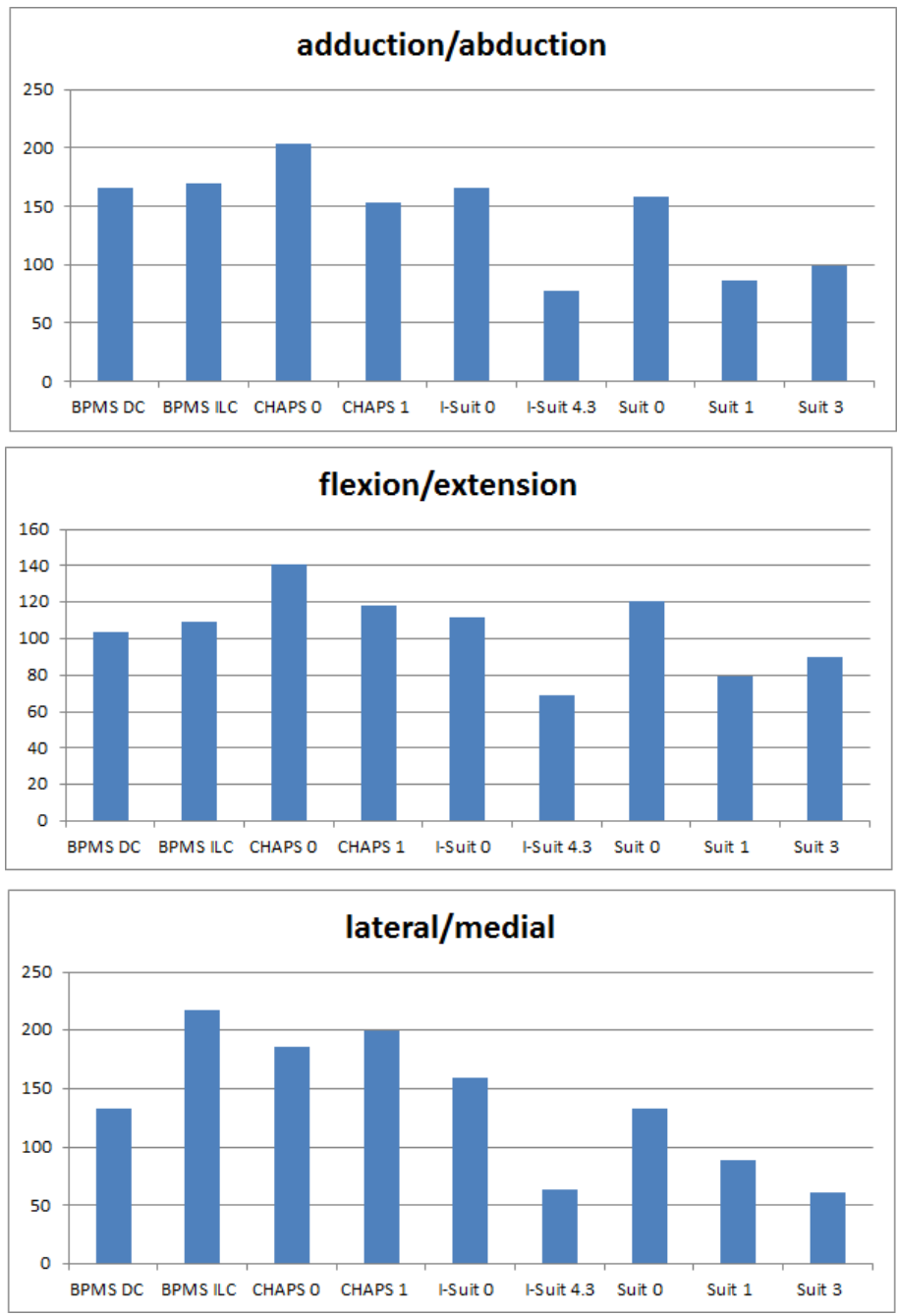


Figure 3.7: Hip ROM

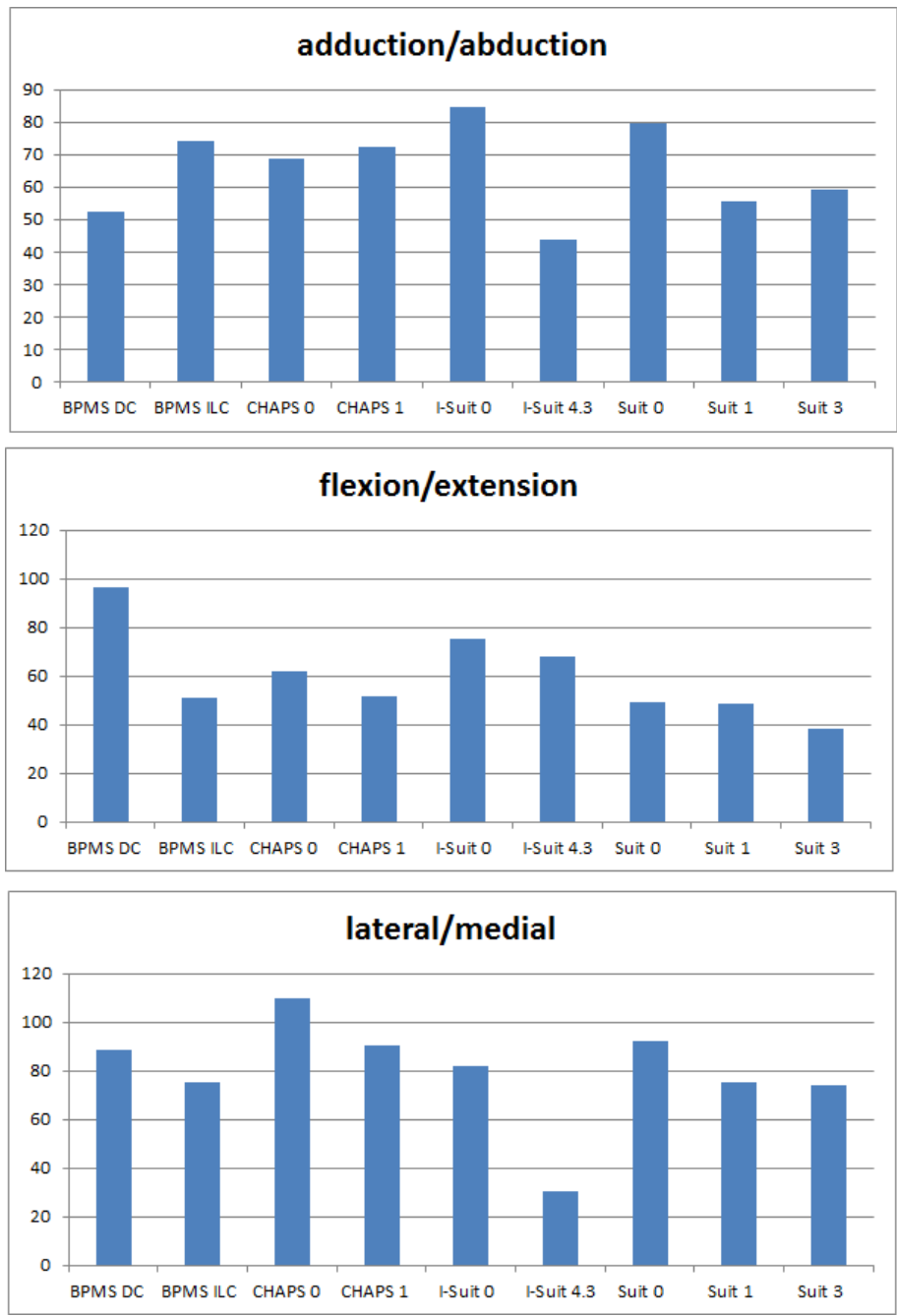


Figure 3.8: Right Ankle ROM

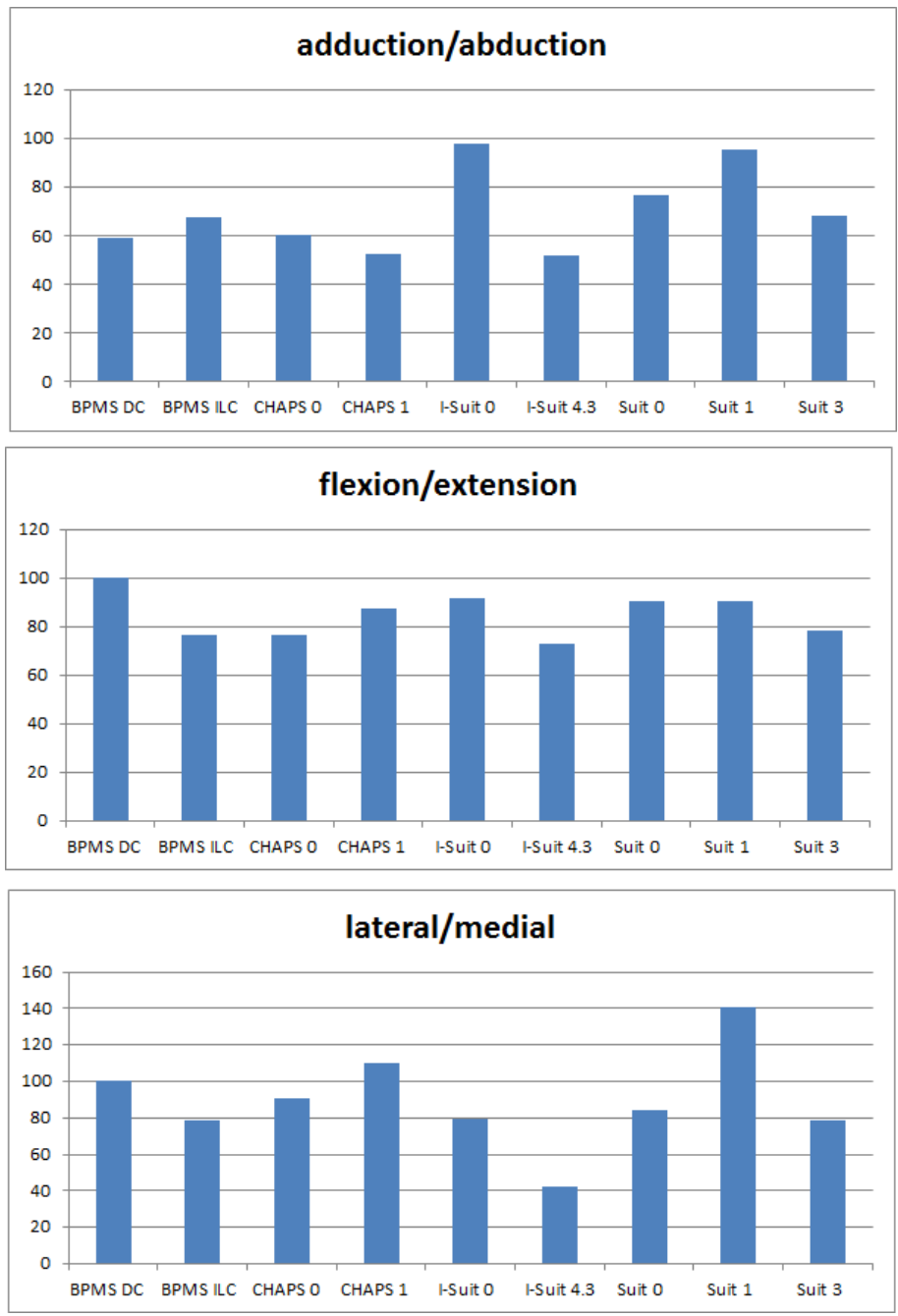


Figure 3.9: Left Ankle ROM

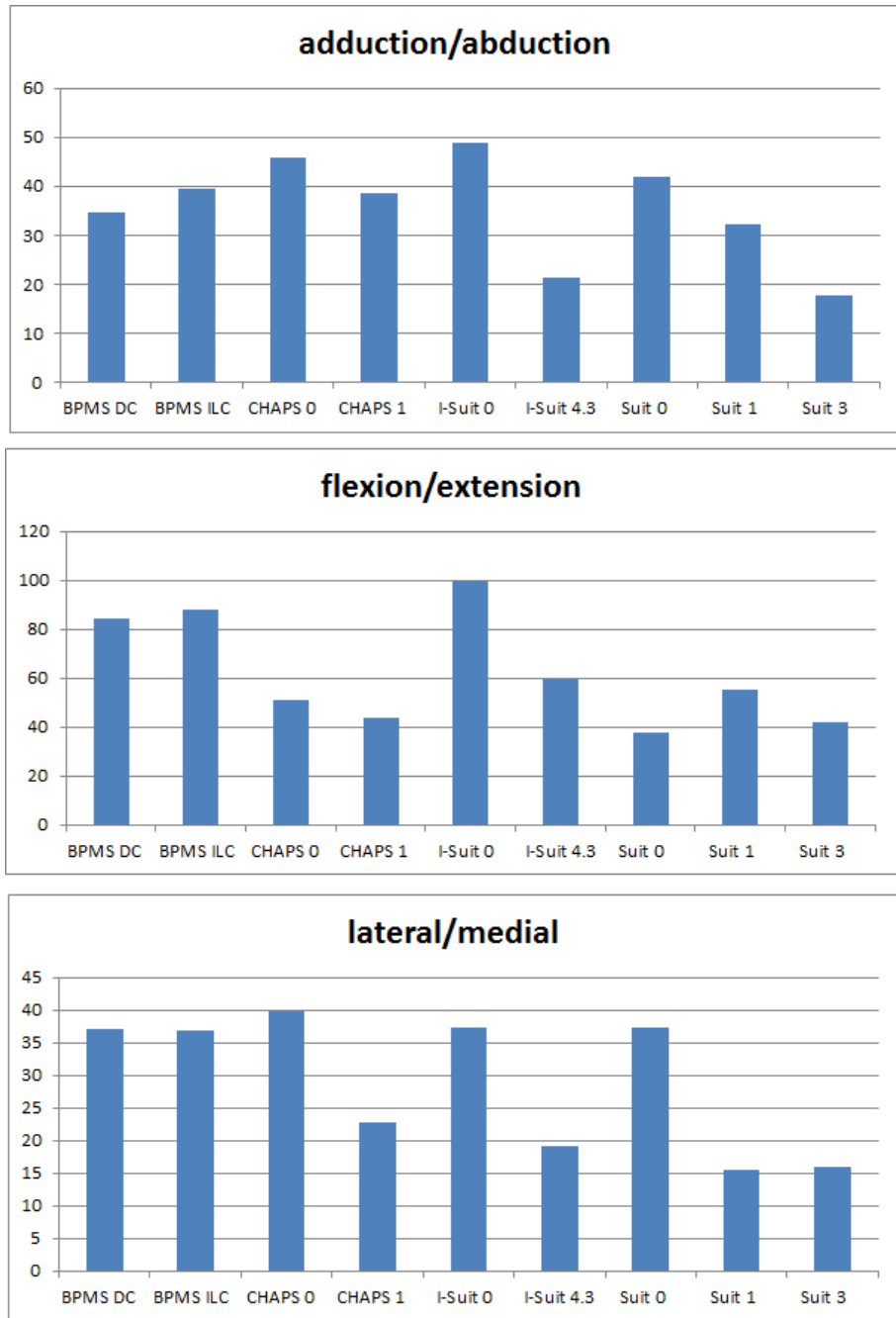


Figure 3.10: Torso ROM

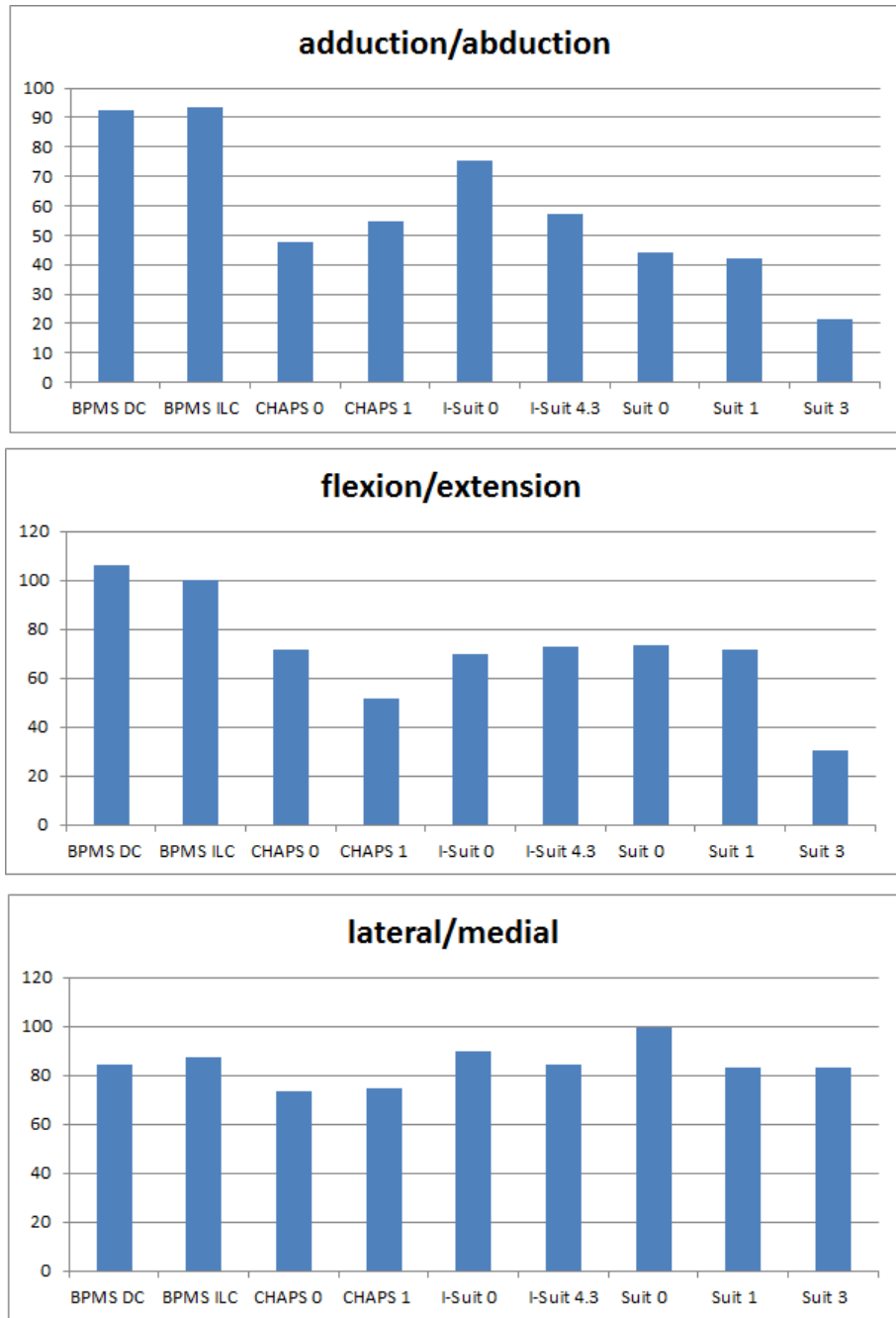


Figure 3.11: Head ROM

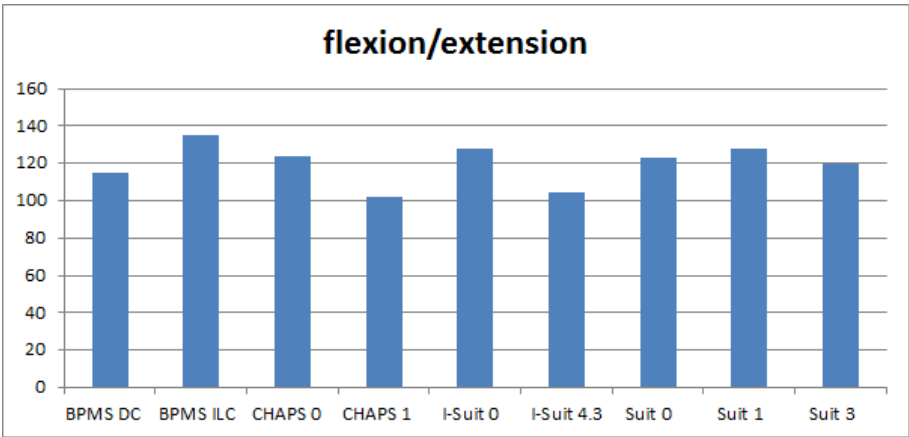


Figure 3.12: Right Knee ROM

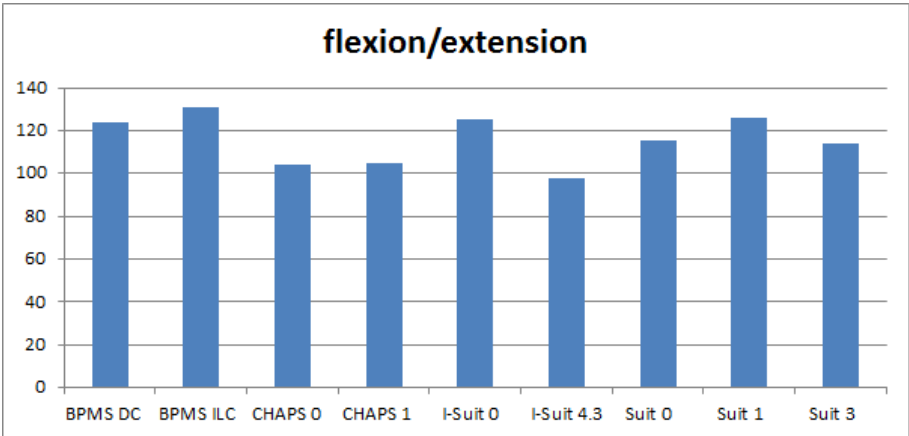


Figure 3.13: Left Knee ROM

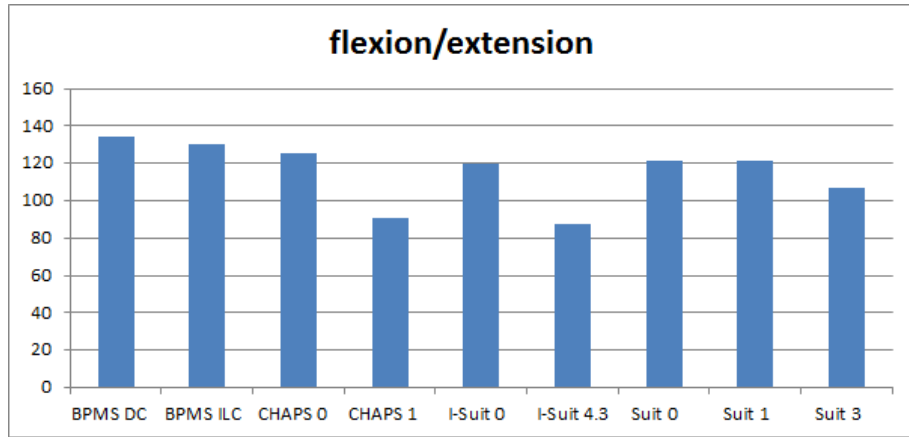


Figure 3.14: Right Elbow ROM

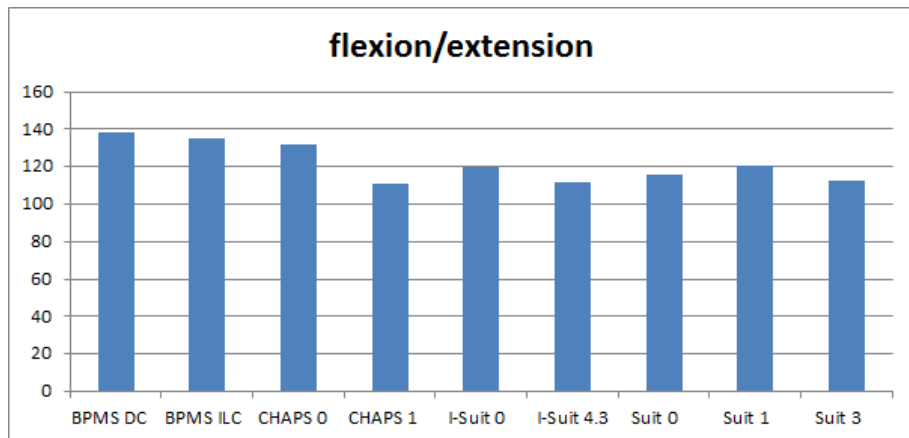


Figure 3.15: Left Elbow ROM

3.3.2 Discussion

Before we begin the discussion on the data, a few very important statements must be made. Since each suit was only evaluated by a single test subject, all generalizations must be avoided. Any visible trend is only valid if seen in the context of that particular test subject. Comparisons between suits must also be avoided since the suits were specifically designed for very different applications. Last thing to note is that in all the evaluations, a metric for effort of motion was not acquired, therefore we cannot make any assumption on the level of effort that was required to achieve such range of motion. From observations during the trials though it was visible that the level of exertion increased as the suit pressure increased. Test subject fatigue was visible during the higher pressure tests while, in the unsuited configuration, motion seemed almost effortless.

From the range of motion analysis, the first general observation to be made is that the results are comparable to the results of several previous studies [21, 2, 39, 65] on pressure suit range of motion. This was to be expected especially since if any variation in attitude between the suit and the wearer inside the suit would be present, those would be small. Due to how the data was acquired, we are thus far unable to quantify this variation. In order to do so, the first step would be to acquire the range of motion data for a statistically significant number of test subjects, also, if we take in account the considerations made by Aitchison in [21], in order to have a valid estimate, the same system (or type of system) would be necessary. In this case we would require two variants of the BPMS system, one worn by the test subject and the second installed on the outer garment of the pressure suit. This approach was unfortunately infeasible for this project due to funding constraints and is left for future studies to address.

In general, from the ROM data, we see that some apparent trends are visible between

the unconstrained case and varying suit pressure but they are different in magnitude for different joints. The following sections will address those differences for each individual suit and joint motion.

Shoulders: In the shoulder range of motion, we can see some variation between the right and left side, but not on the trends. We can see that in the adduction motions the ROM remains almost unchanged for all suits except for the I-suit which as pressure increases, the ROM is reduced almost by half (less on the left side). The same behaviour can be seen in the lateral motion only that this time the major variation occurs for the CHAPS suit. Also in the lateral motion, we see that a wide variation between left and right motion in the i-suit at 4.3psi. It is believed that this variation is not actually representative especially since the two joints are analogous in technology and it is possibly just an artefact of the measurement (the test subject didn't go through the entire ROM in that particular case). Lastly, in the adduction motions, we have very different behaviour in the right and left motions. While the unsuited measurements are comparable, on the right side, we see a large impact on the ROM induced by the presence of the suit while a not so marked impact due to pressure, on the left side though we see the opposite behaviour, a large impact due to pressure with a very small impact due to the suit.

Hip: In general for this joint, apparent trends seem to be related to pressure rather than to the presence of the suit and are consistent across the different motions.

Ankles: Ankle motion seems to be relatively consistent between left and right side and other than an overall reduction in ROM in the I-suit as pressure is increased (in adduction and lateral motion) no particular trends are visible. This result was expected due to the way the ankle joint in a pressure suit is made. In all three pressure suits, there is no proper ankle joint. A military style boot is used instead that wraps around the foot

very tightly. While the gas container is still present inside the boot, both the small size and the high forces that the human system can impose on the joint make it very easy to move.

Torso: In the torso, it is very interesting to see that ROM in adduction and lateral motion varies exclusively due to pressure, while in flexion, except for the I-suit, the variation is induced by the presence of the suit. The torso is the largest and most complex joint in a pressure suit and usually flexion and medial motions are not rarely enabled.

Head: On the head, we can see that in the lateral motion there is no apparent variation due to internal suit pressure. This is expected since both the I-Suit and the David Clark prototype suit have non-conformal helmets, while the CHAPS presents a neck bearing that makes it very easy to rotate the head. On the other hand, we can see that adduction and flexion are both reduced significantly due to both the presence of the suit and internal pressure.

Knee and Elbow: Those two joints are the two single degree of freedom joints of the body. As we can see, the overall performance of all pressure suits on those joints are very high. We can see little to no variation induced by both pressure and presence of the suit and this is the result of high maturity of pressure suit single degree of freedom joints.

As of today, internal motion of a human inside a pressurized suit was never conducted before making this evaluation a first of its kind.

3.4 Geological sortie evaluation

During the 2012 Desert Field Trials, the opportunity arose to evaluate the performance of BPMS while in the field. The tests were conducted at SP Crater, Arizona with the main purpose of evaluating coordinated human and robot simulated exploration sorties. Four field geologists from the Arizona State University participated in the experiments but only one trial was successfully measured with BPMS. During the field trials after the first BPMS trial, two sensors on the garment became unresponsive due to improper donning and doffing procedures and possibly sweat (the sensors were not waterproofed at the time). A field geologist, was equipped with BPMS and with a PDA that recorded simultaneously the GPS position of the test subject along with his body pose from BPMS. During the trial, the test subject conducted a traverse on mostly flat ground, followed by a geological investigation of the border of the SP crater lava flow, followed by a traverse back to the home base. During this 1/2h long session, the subject conducted a series of rock sampling tasks, aided by the use of a rock hammer.

3.4.1 Data acquisition

The test subject was asked to don the BPMS system under his regular clothing and then conduct the geological survey. The focus of the BPMS test was to sample in the field data while minimizing the influence of BPMS on the main task. The subject conducted the sortie and upon his return he was debriefed. A summary of the debrief is provided in the discussion section below.

Additional data was acquired including; audio recordings, handwritten notes and several rock samples. Before the trial was begun, the test subject was asked to stand still in the BPMS calibration pose for 3 seconds and then proceed with the test. On his return



Figure 3.16: Test subject during the unsuited geological sortie with BPMS, Arizona, 2012

to the base camp, once again he was asked to return to the original starting point and remain still again. The two 3 second pauses have later been used to perform an off-line calibration of the system.

3.4.2 Data analysis

The purpose of the data analysis in this case differs from the previous cases since we are attempting to rebuild the path followed by the test subject just by using BPMS and not the user's range of motion. In order to do so, the standard filtering technique is used to retrieve the sensors attitudes as well as the previously defined calibration strategy. Since there was no need to convert the data back to body angles, the so calibrated sensor attitudes are then fed to a simplified lower body model.

This simple model allows us to evaluate the distance between the feet of the test subject as follows: By knowing the DCMs of the legs ($\mathbf{R}_{\text{Leg},\text{R/L}}(\mathbf{t})$), shanks ($\mathbf{R}_{\text{Shank},\text{R/L}}(\mathbf{t})$) and hip ($\mathbf{R}_{\text{Hip}}(\mathbf{t})$), we follow the previously described calibration procedures and find the

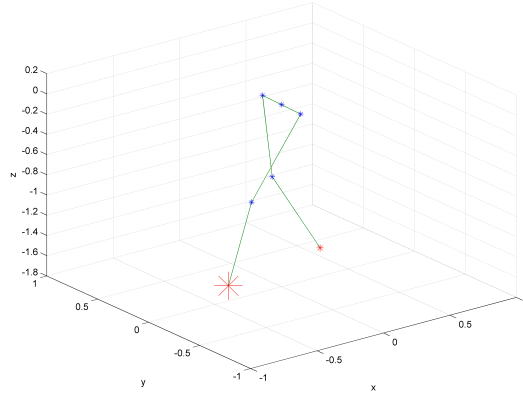


Figure 3.17: Simple lower body model with heel strike detection snapshot (Asterisk size varies with the magnitude of the acceleration)

zero vector in the IMU frame as follows:

$$\vec{V}(t=0)_{limb,IMUframe} = R_{limb}^T(t=0) * \vec{V}_{limb,InertialFrame}(t=0) \quad (3.11)$$

Where $V_{limb,InertialFrame}(t=0)$ are:

$$\begin{aligned} & [0, 0, -1]^T \text{ (For : Legs Shanks)} \\ \vec{V}_{limb,InertialFrame}(t=0) \Rightarrow & [0, 1, 0]^T \text{ (For : Right Hip)} \\ & [0, -1, 0]^T \text{ (For : Left Hip)} \end{aligned} \quad (3.12)$$

From this initial vector, we can evaluate the distance between the feet at each heel strike by first evaluating the position of each foot from the sum of the vectors for each leg as follows:

$$\begin{aligned} \vec{V}_{limb,InertialFrame}(t) &= R_{limb}^T(t) * \vec{V}(0)_{limb,IMUframe} \\ \vec{P}_{Rfoot}(t_i) &= \vec{V}_{RHip}(t_i) + \vec{V}_{RLeg}(t_i) + \vec{V}_{RShank}(t_i) \\ \vec{P}_{Lfoot}(t_i) &= \vec{V}_{LHip}(t_i) + \vec{V}_{LLeg}(t_i) + \vec{V}_{LShank}(t_i) \end{aligned} \quad (3.13)$$

where t_i are the times when a heel strike is detected. Finally, depending on whether it is a right or left heel strike, we define the distance vector between the feet as:

$$\begin{aligned}\vec{D}(t_i) &= \vec{P}_{Rfoot}(t_i) - \vec{P}_{Lfoot}(t_i) \text{ (Rightfootstrike)} \\ \vec{D}(t_i) &= \vec{P}_{Lfoot}(t_i) - \vec{P}_{Rfoot}(t_i) \text{ (Leftfootstrike)}\end{aligned}\tag{3.14}$$

And produce the map by adding the so obtained distance vector to the previous one.

$$\vec{D}(t_i) = \vec{D}(t_{i-1}) + \vec{D}(t_i)\tag{3.15}$$

Walking is usually conducted by swinging one leg while pivoting on the other and maintaining contact with the ground at all times. A very characteristic phenomenon during walking is called the heel strike. Heel strike occurs when the swing leg comes to contact with the ground and becomes the new pivot point for the next stance. Heel strike is usually characterized by a shock on the lower extremities and since BPMS is equipped with accelerometers, the natural step was to observe the acceleration magnitude time sequence and attempt to detect the strike. The accelerations used in this process were extracted from the raw sensor readings of the feet IMUs. The figure below shows a small sample of the acceleration profile during the initial traverse to the geological site. The traverse was mostly on flat irregular ground and from the accelerations, the left and right heel strikes are clearly visible.

From the accelerations profile, we notice two things: The first is that all heel strikes are of magnitude greater than 2.0g and occur at mostly regular intervals (one right and left strike every second hence the stride cycle is very close to 1Hz). We can also see that in most cases, the heel strike shock, propagates through the body and is still registered in the

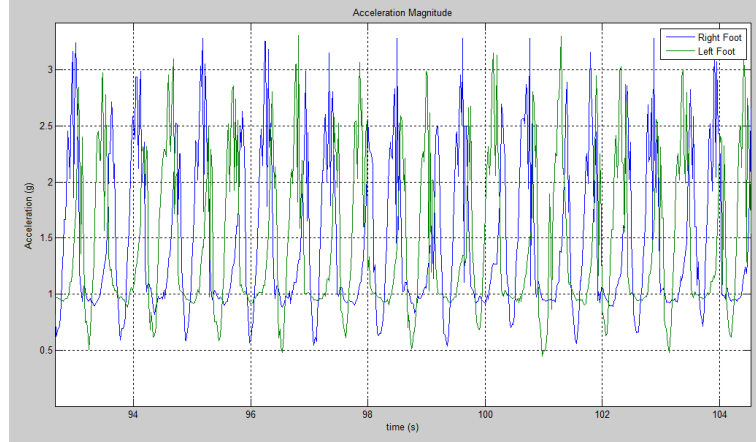


Figure 3.18: Sample of the acceleration magnitude profiles for the right and left foot

opposite foot but the magnitude is usually reduced significantly. In order to extract the heel strike time indexes, we have to find all local maximums of the acceleration function and then verify that we have an alternating sequence of left and right strikes. A matlab script was used to reduce the data for this trial where we make use of the `findpeaks()` function. This function will scan the acceleration profile and return the index of any local peak of magnitude greater than 2.0g that is separated from its previous by at least 0.4 s. Once we obtain the location of the peaks, we can evaluate the distance between the feet at those particular times and build a map by adding the current distance to the previous position. The GPS data on the other hand need to be processed as well. From the GPS logs we obtain the latitude, longitude and altitude of the test subject. This measurement needs to be converted to a flat map expressed in meters and to do so, we use the UTM (Universal Transverse Mercator coordinate system) relations.

From the UTM GPS track, we define the initial location of the subject for the BPMS map but before we proceed with the dead reckoning estimation we must correct for an additional parameter. GPS maps are in reference to true north and not magnetic north.



Figure 3.19: Satellite map of the GPS log, Source:GoogleMaps

BPMS on the other side is only able to detect magnetic north. In order to correct for the magnetic deviation for that area, we resorted to the current aeronautical sectional charts for the area and verified that the magnetic variation in June 2012 over SP crater was 12.4 degrees E. In order to take this in account, we rotated the BPMS map -12.4 degrees and overlaid it on the UTM GPS map. The two tracks are shown in the figures below.

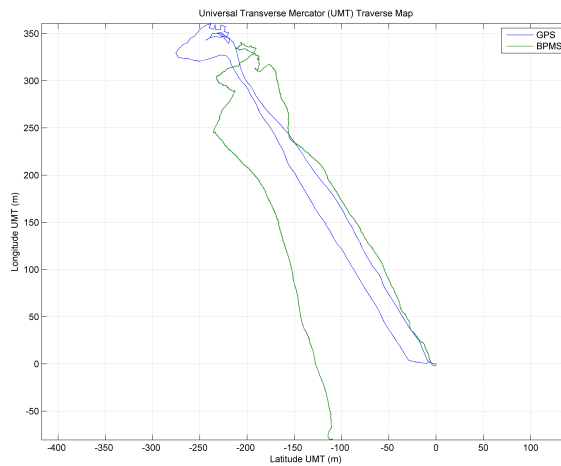


Figure 3.20: BPMS track and GPS track of a simulated geological sortie in the Arizona desert

From the 2D plot above, we can see that the BPMS track follows the GPS track quite well for the first portion of the traverse but errors the add up and during the return path become more and more visible. The main sources of error include numerical errors, position estimate errors and heel strike detection errors. While all those error play an important role, from the data analysis it was clear that heel strike detection was perhaps the most important. As expected, different peak thresholds returned very different results. On a side note, we also must remember that the GPS log is also plagued by errors. The Arizona desert is not the most covered area when it comes to GPS satellites and from the logs we saw that the accuracy of the GPS was usually between 5 and 30m. Also, we know that commercial GPS systems for private use, are not that accurate when it comes down to measuring altitude. During the traverse, the subject followed a slightly downhill path in his outbound portion, then climbed over the lava flow and finally returned to base following almost the same outbound track therefore going uphill. From the previous plot it is not possible to see the elevation changes therefore the following 3D plot was produced to highlight the ability of BPMS to map the altitude profile quite nicely. From the 3D path we can see that the BPMS track resembles much more the actual followed path than the GPS track when it comes to elevation.

3.4.2.1 Discussion

From the analysis above, it is clear that with BPMS as with any dead reckoning mapping strategy, we will have drift over time on our estimates, but we can also see that perhaps we gain a much higher resolution map when compared to GPS systems for small distances (where we can assume that the error build-up is small). In order to mitigate the drift issue, we could combine the two strategies and use the GPS to re-calibrate the

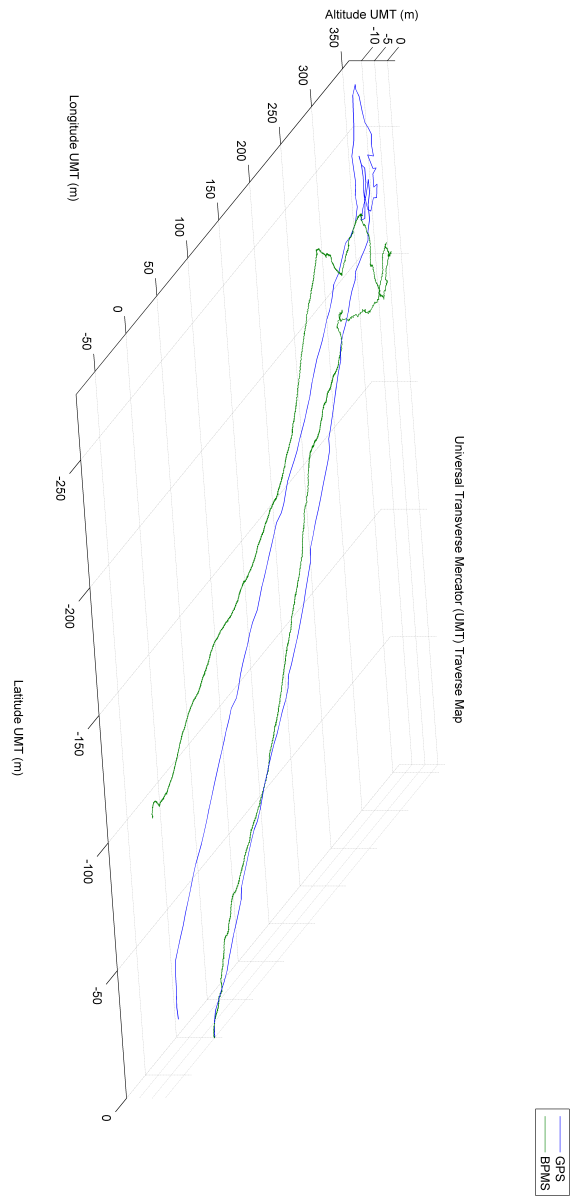


Figure 3.21: BPMS 3D track and GPS 3D track of a simulated geological sortie in the Arizona desert

BPMS map at regular intervals and obtain a better estimate. From the data above we can attempt to give a metric for the drift over time, since the sortie was begun and ended in the same exact location, by comparing the initial and final position of the BPMS track and dividing it over time, we can have a tentative estimate of average drift during the sortie.

$$\begin{aligned}
 error_{Lat} &= 79.23m \\
 error_{Lon} &= 110.39m \\
 error_{Alt} &= 2.69m \\
 T_{tot} &= 1830.5s
 \end{aligned}
 \tag{3.16}$$

$$\overline{Drift}_{Lat} = \frac{error_{Lat}}{T_{tot}} = 0.0433m/s$$

$$\overline{Drift}_{Lon} = \frac{error_{Lon}}{T_{tot}} = 0.0603m/s$$

$$\overline{Drift}_{Alt} = \frac{error_{Alt}}{T_{tot}} = 0.0015m/s$$

During the debrief of the test subjects they were asked whether they felt any mobility constraints induced by the system and the general consensus was that the BPMS garment didn't noticeably alter their motions but it did impact their body temperature. As expected, a black full body garment in the Arizona desert during the day is not the ideal choice when it comes to heat rejection. Other than the heat issue, the subjects were very responsive to the system and didn't have any complaints in matters of comfort and range of motion.

Chapter 4

Future Work and Conclusions

4.1 Future Work and Potential Applications for BPMS

In this work, BPMS has demonstrated the ability to measure body angles and human body pose inside a pressure suit. The natural extension of this work would be to move beyond the use of BPMS to being a measurement tool, adopting it instead as the basis for an advanced control interface. Although current pressure suits are far from being mechanized, BPMS offers a viable starting point for future power-augmented suits by providing a feedback system for future exoskeletal systems, or as a control interface for other robotic systems such as rovers and robotic manipulators. Integrating BPMS on current pressure suits or liquid cooling garments is a very real possibility. BPMS could be initially implemented as a standalone secondary system and, given its working principles and properties, would not pose any additional risks or mobility constraints. Although this first iteration of the system has been developed exclusively for terrestrial applications, it is fully capable of being qualified for use in an oxygen environment, which would allow it to be used in orbit, on the Moon, or Mars. The following section will present the concepts that have been envisioned for the future of BPMS, starting from modifications and optimization of the internal software attitude filters, to electronics upgrades and a series of potential applications.

4.1.1 Attitude Estimation Filters

In this work the attitude estimation filter was arbitrarily selected[16], and was tuned but not optimized for this specific application. This generic complimentary filter proved to be sufficient for this application, but it is clear from the previous data analysis that better filters can be implemented. In order to develop a better filter for a system like BPMS, the first step would be to include a dynamic nonlinear model tuned specifically for each sensor[45] in the garment that is calibrated for each individual test subject (link lengths and therefore the read dynamics vary between test subjects and sensor location on the body.) This approach alone would reduce cross axis errors and increase the convergence speed of the filters, as well as deliver a more accurate estimate of the body pose[36]. This is perhaps the most important research point for future development, since it significantly affects the quality of the output data from BPMS. In order to accomplish such a task, a series of simulated IMU experiments[55, 57] must be run, and various filters implemented and compared against each other.

4.1.2 Electronics upgrades

The current BPMS prototype is the result of several design iterations; since the sensor selection and architecture definition was identified early in the process, the sensor technology selected is now obsolescent, and incorporates a suboptimal wireless interface. BPMS could largely benefit from the current generation MEMS sensors that are quickly becoming smaller, more integrated, and much more capable. Better sensors for BPMS could lead to better tracking accuracy and overall better system performance. As time progresses, we will see more and more single chip IMUs like the Invensense MPU-9150, which will also reduce the mass and power requirements of inertial motion capture systems;

by simplifying the overall architecture, the system reliability can be increased. Other technologies that could very well improve the future iterations of BPMS include elastic electrical cables. BPMS today uses standard electrical connections; as was observed during the various trials, the donning and doffing phases are the two most critical moments in which the system can be damaged. By adopting elastic cables, BPMS would be able to endure many more don and doff cycles, as well as increasing the system comfort for the wearer. Unfortunately, though, current technology elastic cables have their drawbacks, which must be evaluated extensively before integration. Increased wire resistance and variable cable length could impinge on the high speed data communications between the sensors and the various microcontrollers and therefore make the modification infeasible.

Last but not least, the wireless communication interface on BPMS will need a revamp. As of now, Class 1 Bluetooth transmitters and receivers allow us to interface BPMS with virtually any device (desktops, notebooks, PDAs, smartphones, etc.) but this comes at a high price: very limited range. The first step in the upgrade would be to move the transmitter from the chest to a more visible location such as on top of the head of the user or external to the suit, such as on the helmet. Due to the low power, 2.4 GHz link frequency, Bluetooth is very susceptible to any obstacle that contains water, and the human body is a perfect shield. By placing the antenna on the chest of the user, if the test subject is not facing the receiver element, the connection is lost within 15m or less. By placing the antenna on a more visible location, we can regain the lost range and have a maximum range of 100-150m in ideal conditions. The small size and weight of the interface allows such a modification without inducing any noticeable drawbacks. A simple extension to BPMS would be to allow for a modular radio port, where different transmitters could be used based on operational needs. In case BPMS needs to be used with

portable devices such as a smart phone or a PDA, it could be equipped with the canonical Bluetooth radio (low power, fast and reliable communications, but short range), or in case long range comms are required, several other options are available. Long range wireless interfaces usually use frequencies between 433 MHz and 900 MHz, allowing for line of sight high speed comms up to 1 km which are much less susceptible to obstacles. The price for such interfaces is mostly power. Longer range requires more transmission power; this result in larger batteries if one is to maintain the same operational endurance of the system. Alternatively, a modular transmitter interface can be easily converted to a wired connection to an onboard computer, thereby increasing the reliability of the system, as well as expanding the wireless communications possibilities; if the computer is connected to a 3G, LTE or satellite mobile radio, the range can be considered unlimited.

Other possible system upgrades include an on-board GPS receiver, a data storage unit, and an embedded heart rate sensor. As of now those two sensors have been incorporated by using a PDA that connects to both BPMS and a Bluetooth heart rate monitor simultaneously, as well as using its internal GPS receiver and memory. By embedding those devices, BPMS will become a more standalone platform. In addition to the sensors above, BPMS can also be equipped with a network of piezoelectric bend sensors to monitor finger motions, as well as a series of pressure sensors to monitor force distribution on key portions of the body such as on the palm and fingers. Prototypes of this kind have been developed in the past at the SSL, but have not yet been integrated yet on BPMS.

The hardware modifications to the existing system as of now are only limited by the budget for the current project, and by the designer's imagination and time. As this research effort progresses we will see a more capable system at each iteration that will hopefully lead to a reliable, accurate and self-contained biomechanical harness capable of

measuring the kinematics, dynamics and biometrics of a test subject under any operational scenario.

4.1.3 Operations in Microgravity Environments

If BPMS were to be deployed as-is in a micro-gravity environment, the first thing that would be noticed is unreliable attitude measurements. The filtering strategies adopted on Earth cannot be used in this very different scenario for two major reasons. First of all, the gravitational acceleration will be absent, and therefore the reference ground plane will be undetermined. Also, the magnetic field will vary considerably during the orbit, rather than the effectively static magnetic field which obtains for field testing. In order to mitigate this effect it is possible to define new nonlinear models in the complimentary filters that take in account for this variation. The system in any case would still rely heavily on rate measurements which are known to induce drift in the attitude estimates. On Earth, magnetometer and acceleration readings can correct gyro drift, but in microgravity, those two sensors would become less useful, and a different approach would be required. First of all, an accurate knowledge of the current orbit and magnetic field of the earth are required and need to be modeled in the filter, as well as knowledge of the current position of the BPMS wearer in the orbit. In addition to the above, in order to be able to use the acceleration readings, each sensor will be required to have a specific internal model based on its location on the human body. By doing so we can use the accelerations to estimate angular rotations and fuse the measurement with the gyro measurement. At this point in time this approach is very speculative and it requires extensive development and testing before it can be considered.

4.1.4 Operations in Partial-Gravity Environments in the Absence of a Planetary Magnetic Field

Partial gravity operations can be easily accommodated by BPMS with very few modifications to the filters themselves. In order to use the current filters in this new scenario, the accelerometers need to be re-calibrated for the new gravitational environment. This is true if the planetary body in question presents a strong enough magnetic field. In the very likely case that the celestial body doesn't possess a magnetic field (for example, the Moon), a different strategy is required. Although BPMS will still be able to track changes in yaw, they will be relative to the initial position (gyro based measurements) and will be affected by drift. To mitigate this, a possible strategy includes the use of a known magnetic field source (electromagnet or permanent magnet) positioned in a known location on the wearer. The magnetometers at that point would detect this artificial magnetic field and reference all the measurements to that source. The filter internal models need to take this in account; if the magnetic source field amplitude is known and mappable, by evaluating the magnitude of the magnetic field sensed by each of the sensors, not only do we obtain the attitude of the sensor but also an estimate of its distance from the source. (In the past this was the strategy adopted by the Flock Of Birds system.) This approach is necessary when a magnetic field is not present, and will still return the pose of the wearer in relation to the artificial magnetic reference, and not in reference to an inertial frame. This approach is very interesting, since it can possibly lead to more accurate relative position measurement not only in the absence of a magnetic field. If this approach is used on Earth or in any other scenario, it would require a magnetic source powerful enough to shadow the existing magnetic field; the Earth's magnetic field is very weak, therefore very hard to accurately detect and easy to shadow. In addition to having a constant magnetic

source, if we were to pulse, sync and control the sensor acquisitions, we would be able not only to obtain the relative pose and position through magnetometer readings, but we would also be able to detect the existing magnetic field, and therefore obtain attitude in reference to it as well. Once again this is still a very speculative concept that hopefully through extensive research can improve the capabilities and operational scenarios where BPMS could be deployed.

4.1.5 Human factors systems integration evaluations

As of today, rigorous systems integration evaluations are very difficult to conduct since the measurement tools used are not intrinsically compatible with the type of investigation. Today, if we were to measure the body pose of an astronaut while entering and exiting a spacecraft, we would have to resort to low fidelity transparent mockups of the spacecraft in order to allow an optical motion capture system to detect the markers on the astronaut's body. This strategy is used today for lack of a better system that doesn't suffer from the marker occlusion problem. BPMS in this case would be the ideal choice since it would not require any additional lower fidelity mockups development and higher fidelity systems could be used while still obtaining the same but cleaner and more usable data. BPMS could be used to study human motion in suited (pressurized and unpressurized) and unsuited configurations with virtually no additional requirements.

4.1.6 Exoskeletal suits feedback interface

Exoskeletal suits have been in space suit designer minds and roboticists for quite some time [33, 30, 44, 47, 72, 73], but there are many challenges that need to be addressed before even attempting to produce a prototype. First and foremost such a critical system

must be capable at least to move the suit out of the way of the wearer; therefore knowledge of the current and desired pose of the human inside the suit is a requirement [33]. Although several strategies have been conceived (positive force feedback, proximity sensors, etc.) none included inertial, in suit, motion capture. BPMS can provide the information required to close the control feedback loops and if paired with an analogue system on the exterior of the suit, it could also provide state information for the exoskeletal system. In principle BPMS can measure the body pose of the wearer, convert the pose in body joints and be mapped to the exoskeleton joints through a model that not only ensures desired end effector position, but also overall pose compatibility between the test subject and the suit. BPMS is just a very small piece of the puzzle because many other issues need to be resolved such as; reliability and fail safe properties of the system, mechanized joint compatibility with a pressure suit, mobility impingement (work envelope) and many more.

4.1.7 Pressure suit based robot teleoperations

Space robotic systems as of today are either teleoperated from the ground or from an IVA crew, but there has not been any history of a teleoperations task where a human in a pressure suit while in EVA has taken control of a robotic manipulator. This could be enabled by using BPMS as a master interface. The control strategies possible are several; The first and most simple (implementation wise) would be joint by joint control, where human joint angles are directly mapped to the robotic manipulator joints. A slightly more complex strategy could be to map a single human limb to the end effector. This can be achieved in several ways, but possibly an example will serve far better in explaining the concept. Through BPMS we have access to the pose and position of all the major long

bones and extremities of the body. The most natural way of interacting with everyday objects for humans is their hands hence, by mapping the attitude and position of the user's right or left hand to the robot end effector could be a very viable and efficient alternative. Also, given the fact that humans are widely capable of coordinating two arms at the same time, BPMS teleoperations capabilities could be used to control two robotic manipulators at the same time. Lastly BPMS allows for head tracking which, alternatively could be used to control a third arm with a camera that is fed to a head or helmet mounted display in the suit. The first contrary argument to this possible application is that is that this task is far more simply conducted while in IVA and there would be no need for it to be executed while in EVA. As it has been stated many times, BPMS's uses are not confined to just pressure suits. Therefore the same strategy could be used while in IVA. On the other side, as future space missions move further away from earth and astronaut preparation and rehearsal becomes more generic, having the capability to conduct such tasks, might end up finding its place. An example of robotic teleoperation that is very useful in EVA, is the ability to control a rover through gestural control. In summary, if we assume we are using BPMS in a pressure suit for monitoring or as a feedback interface for a robotically augmented space suit, we have a new control interface available that could be used without any modification to existing systems. On the other side, in order to reach this level of reliability, BPMS must undergo a rigorous reliability evaluation and the filtering algorithms must be robust and accurate and must ensure safe operations. As MEMS technology progresses, more accurate and smaller sensors will become available along with higher performance filters that if merged in a system like BPMS could enable an even wider array of possibilities.

4.1.8 BPMS as an interactive interface

As everyday technology shifts to a more user oriented philosophy where the human is the control interface, it is to be expected that it will eventually find its way in the very resilient field of pressure suit and IVA space operations. Today's smart phones and tablets choose touch screens as the principal interaction interface which is great for a 2D synthetic planar interface, unfortunately the real world is not bi-dimensional and as technology progresses it is very likely that a lot of effort will be given in closing the gap. 3D user interfaces are slowly making their way across consumer devices and hopefully in the non distant future they will also be implemented in pressure suits. Head mounted and helmet mounted displays are slowly finding their way in current pressure suits. Several prototypes have been proposed and are currently being evaluated, but once those systems are deployed, a new interactive interface will have to be implemented. Mouse and keyboard in a pressure suit will most likely not be the optimal choice. Therefore a more natural interface could be gesture recognition. BPMS provides the ability to sense the pose and position of the user's limbs (hands or fingers) in 3D space and therefore allow a natural human-machine interface. Extensive work needs to be done in terms of defining the interaction methodology and the user interfaces especially because such interfaces are as of now, part of the science fiction world rather than reality and plenty of questions still need answers. BPMS in this case is possibly only a small piece of a much bigger puzzle. Lastly a very interesting potential application is astronaut training and virtual reality. Astronauts frequently train in virtual environments for their tasks. BPMS could be used as the input interface for the astronaut's avatar and allow the visualization and interaction of the user with the virtual environment.

4.1.9 Non pressure suit related applications

The applications for BPMS are not confined exclusively to pressure suits, in fact, its ability to be used in such a constraining environment makes the system even more appealing for all those other less stringing scenarios. BPMS once refined and if reproduced in large numbers, could lead to a low cost high performance measurement system capable of granting investigators the bio-mechanical data they need from the real scenarios they wish to observe. The medical world could highly benefit from such a system to either monitor or diagnose patients. A simple example can be seen in the orthopedics field, where BPMS could allow doctors to monitor patients during their daily activities and identify which movements are being executed in a harmful way and help the patient correct them, or to diagnose possible illnesses by observing the same data. BPMS would also be ideal in the professional sports field, where monitoring and observing the athlete's motion could be a very valuable piece of information, especially when fine tuning and overall bio-mechanical measurements of motion are important. Today, this type of analysis is confined to a laboratory setup and therefore it allows the athlete to acquire information only during simulated activities. If this information would be available to the athlete continuously and in real-time during his training, it could potentially increase its effectiveness and maybe his overall performance. BPMS even in its prototype stage is able to grant all this information.

4.2 Conclusions

This work has just scratched the surface of the possibilities enabled by a system such as BPMS; although many improvements on the system can still be made, this work hopefully has set a basis from which future iterations and applications can start building.

A comprehensive literature search was performed in order to understand current measurement methodologies and pressure suit evaluation techniques. Based on the current gaps in the literature, it was conceived that inertial motion capture could come in handy to answer several questions that plague the field of space suit design and operations.

A novel inertial motion capture system capable of measuring the pose of a person inside a pressure suit was designed, built and tested. The electronics and software required to acquire, visualize, and record the data was developed and tested, and the performance metrics of the systems were also determined. BPMS is able to measure a person's body pose with equivalent performance to optical motion capture systems, but without requiring a massive equipment overhead or line of sight with the sensors.

In a multi-IMU system like BPMS, the various sensor attitudes must be measured in the appropriate frames. In this work a single point calibration strategy has been implemented and evaluated to define the initial reference frames to which the sensor are referred. This approach provides the system the ability to map, based on a given model, the IMU's orientation to a specific initial pose, therefore allowing the system to reconstruct the the body pose of a test subject from arbitrary initial sensor attitudes.

Following the calibration strategy, kinematic evaluations of humans inside three different pressure suits at different pressures were conducted. The scope of these tests was to address the effect of internal pressure on range of motion for each specific suit, and then compare the results against the unsuited reference case. For the range of motion evaluation, a methodology to evaluate human joint angles from IMU angles was required. By using a series of geometrical relationships between the inertial sensors and the constraints of the human body joint, we are able to translate the IMU sensors attitudes to the canonical set of human body joints, and therefore compare the measurements obtained

with BPMS against the current literature.

Lastly, a dead reckoning algorithm to reconstruct the path followed by a field geologist during a simulated sortie was conceived. In order to demonstrate that BPMS's capabilities are not constrained to pressure suits or to laboratory settings, as a target of opportunity BPMS was used to record the pose of a field geologist during a simulated sortie in the Arizona desert. The test was mainly aimed to show that significant data can be acquired by BPMS, even in extreme environments, without affecting the mobility or capabilities of the test subjects. By being able to acquire the kinematics of a test subject in both suited and unsuited configurations while conducting a simulated sortie in an analogue environments, BPMS can be used in the future to better inform the design of the next generation of planetary space suits.

Appendix A

Early BPMS Prototypes

A.1 Instrumented Space Suit Glove

NASAs current vision for space exploration will include possibly going back to the moon or mars. New challenges will arise and astronauts will be required a much higher level of autonomy than what was done in the past. In order to aid this research effort, the Space Systems Laboratory at the University of Maryland has dedicated its attention to astronaut-assist rovers and developed Raven (Shown in the figure below along with MX-Alpha, space suit test-bed). In this work we will investigate the development of an instrumented glove to remotely or locally control such rovers using in this particular implementation, gestural control. Gestural control is a non-invasive way to interact with synthetic elements but it requires the control system to be compatible with the human hand.

Also to add an additional layer of complexity, the system will also have to be compatible with the space suit environment since such system will be necessary for the astronaut survival in the harsh and unforgiving environment of the moon, mars or deep space. In order to achieve those objectives a non-pressurized glove prototype has been developed and has been instrumented to allow us to identify hand poses and grasps. Through the use of an array of bend sensors and MEMS devices, we



Figure A.1: MX-A and Raven

are able to identify each fingers bend angle and the attitude of the right hand, translate those measurements in poses and use the analog signals to control the rovers movements. This project is a small part of a much greater effort that is currently undergoing and that will produce a space suit mockup that will allow the wearer to seamlessly interact with synthetic elements via gestural control, speech recognition, digital I/O and it will also allow the wearer to increase his situational awareness through the use of synthetic environments projected inside the helmet (Head-Up-Display). An instrumented glove will also enable us to investigate pressure glove performance by allowing us to compare the difference in the hand workspace when pressurized versus free hand performance; also it is the first step towards the implementation of a robotically actuated glove where the measurement system would function as a feedback system for the control loop. This system will also allow us to manipulate virtual objects in the virtual or augmented reality environments, etc.

A.1.1 Background

A.1.1.1 The human hand, and the space suit glove

In order to design a human compatible system we have to begin by understanding the human system we wish to interact with and also keep in mind the external constraints to which it is subject. The human hand is a very complex limb that enables us to interact with our environment. There have been extensive studies in regard but there still is no agreement on the significant degrees of freedom of the hand. In this work we will use the following definition:

The hand can be viewed from a robotics prospective as a combination of serial manipulators all having rotary joints some of which co-located. The reference model that we will use is illustrated in the figure on the side and as we can see, it counts 24 degrees of freedom. Since the biomechanics of the hand are out of scope in this work, we will just say that not all of them are independent. Each finger is actuated by two sets of antagonistic muscles located in the forearm and by a few other smaller muscle groups in the palm. Since we do not have a direct mapping of the actuator force on the joints, a few degrees of freedom are not independent reducing the independent degrees of freedom of the hand to 20 (as long as we are bound to a small range of motion around the neutral pose).

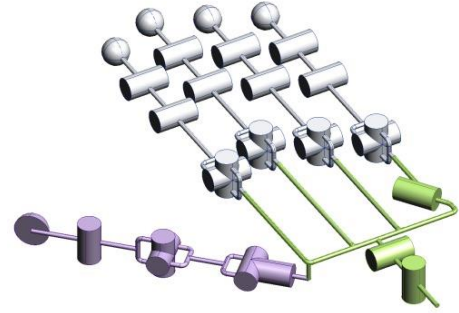


Figure A.2: The human hand in the robot perspective

Lastly a few words should be dedicated to the biological sensor called the muscle spindle which serves as a position feedback system within a muscle group and that allows the brain to be aware of the current pose of the hand. The muscle spindle is part of the neuro-muscular systems and it senses the muscle state (how much it is contracted). We are able to translate the muscle state to limb position and attitude from experience (by building the coefficients of the relative transfer function in our mind) and we usually refer to it as physical coordination. Given the hand system we have to introduce another element, the



Figure A.3: Phase VI IMU glove and power-glove without TMG

Table A.1: Finger Torque on a phase VI IMU glove at 4.3psi

Torques (lbs-in)	Without TMG	With TMG
Index	0.6	1.5
Middle	0.7	1.9
Ring	0.6	1.5
Little	0.6	1.7
Thumb	0.8	2.1

space suit glove, or pressurized glove. Since the early days of flight as soon as men began to explore the higher layers of our atmosphere weather it was with hot air balloons or early aircrafts, they began to experience a series of issues at high altitudes that often lead to their death or just to the loss of their equipment if they were lucky. The phenomenon they were experiencing is today referred to as Hypoxia or lack of oxygen. It is not the point of this work to go deep in the analysis of such phenomenon but we can say that this was the driver force for building the first pressure suits. Pressure suits allow humans to survive in environments that would be otherwise deadly by recreating a human compatible, portable atmosphere around them. The space suit is a pressure suit and it is perhaps the most complex human centered spacecraft. It includes all the traditional systems that you would find in a satellite, but it also has to enable the mobility of the subject within it. The state of the art in pressure suits hasn't advanced significantly since the 50s and their mobility is very limited restricting significantly both the user's range of motion and his ability to perform work. It is estimated that about 75% of the work an astronaut does in an EVA (Extra-Vehicular Activity) is used to move the suit around him, and the remaining is actually used towards anything useful.

The space suit glove is no exception and as we can see from the table below, it induces significant loads just to move it from its neutral position (the table below shows

just the average loads to move each finger on a Phase VI IMU glove pressurized at 4.3psi. Loads in general are a function of the bend angle and are directly proportional to it. In the table above we introduce the TMG or Thermal Micro-Meteorite Garment, which is the most external layer of a space suit and also a significant contributor to the performance reduction as we can see from the table. We can see the TMG in the glove on the left in the picture above, while on the right we can see a similar glove without the TMG. Lastly given the complexity of the human hand, as of today, only 9 of the original 24 degrees of freedom are enabled.

A.1.1.2 State of the art in hand performance measuring systems

The body of the literature on instrumented gloves was built in the late 50s with the upcoming of computers and Virtual reality. Since then people have tried to make computer interfaces as intuitive and as simple as possible by attempting to enable the interaction between the machine and the human as human centered as possible. We live in a real world and computer and synthetic environments are usually used to aid the introduction to reality of ideas. In order to do so, since the main mean of physical interaction with the environment is the hand, why not allow it to be the main way to interact with a computer system.

Instrumented gloves were initially introduced in the gaming industry with the Nintendo Power Glove. This system used piezo-resistive bend sensors on the fingers and it relied on an ultrasonic system to determine the position of the glove in the monitor reference frame. The low computational power capabilities of the Nintendo systems and the poor performance of the glove and of the software that was developed for it made it a huge flop. Lately Shape hand was developed but this time the interested industry

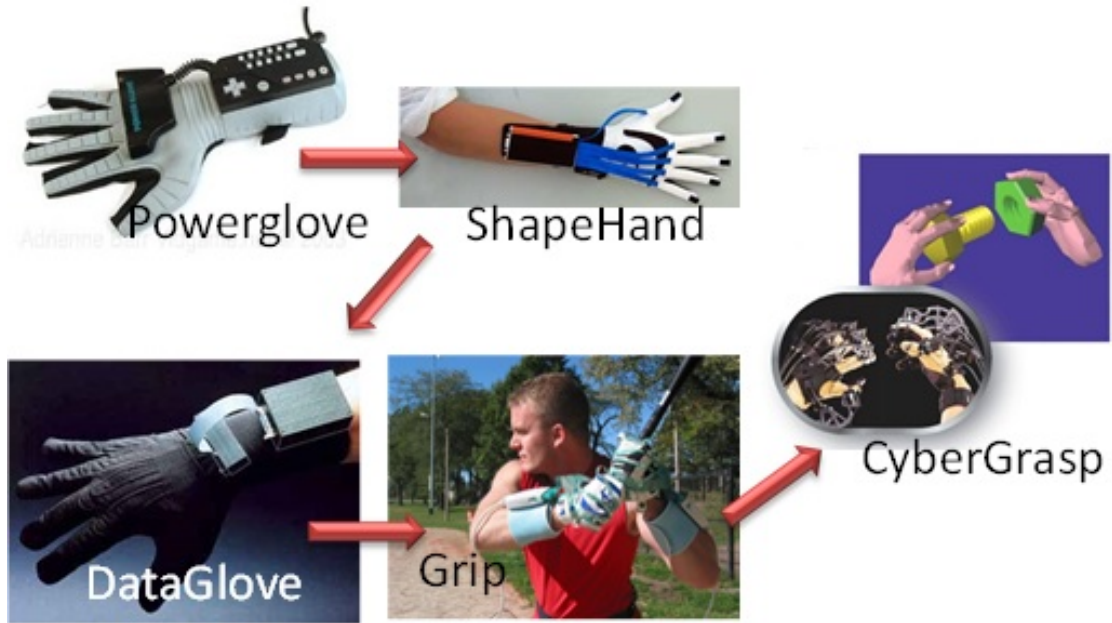


Figure A.4: Instrumented gloves technology in the years

was different. The scientific community began to gain interest in the biomechanics of the hand and this system was developed for research purposes. Shape hand is a simple system that relies on bend sensors exclusively to determine the hand grasp. It has no external attitude or global positioning determination but it introduced the concept of a wireless sensor to reduce limitations associated with cable routing. With the introduction of fiber-optics, bend sensors were improved over piezo-resistive and an analog system to the Shape Hand that adopted this new technology is the Data Glove. Data glove is as of today the industry standard in terms of wireless, grasp and attitude hand measurement system. There are several versions produced some of which use either magnetic position and attitude measurement systems or use MEMS based devices for attitude readings. So far we have talked about acquiring hand joint angles and position. A parallel effort has

been conducted in the determination of the hand dynamics, as in the forces that we can apply on an object with it. Grip is a system that introduces piezo-resistive arrays to map the pressure profiles in the hand, allowing the investigator to measure the pressure points (contact points) and allow the evaluation of the work that the hand is producing and the energy expenditure associated with it. Lastly with the introduction of CAD (Computer Aided Design) cyber presence was also being explored. Cyber presence is the ability to feel virtual elements as if they were real (think of is as surround sound, stereoscopic imaging, force feedback, etc.). In the glove realm, Cyber grasp is the only system that I know of that allows the user to feel the presence of a virtual object by using a robotically actuated glove to apply the reaction forces that one would experience when manipulating a real object. In order to achieve its goal it relies on a closed loop control system that reads pressures from various points on the fingers, and palm of the hand, measures the hand position through a series of fiber optic bend sensors and determines the hand attitude with a 3 axis accelerometer and corrects it though an infrared vision positioning system. This last system, if mignaturizable would have immediate application in the space suit system. It would be able to eliminate the torques introduced by the glove and restore free hand performance. It could also, if capable, augment human performance and would introduce the concept of a robotically actuated space suit (Robotic human compatible Exoskeleton). This is the final goal of my research.

A.1.2 Design Phase: Trade Studies and System Specifications

A brief bibliographical search was conducted to identify possible candidates for this specific application, and several alternatives were found. In this section we will cover the various candidates that were considered, the selection is by far not complete nevertheless

it includes a wide enough variety to satisfy the requirements for this work. The system is required to accomplish two main functions; the first is to measure finger bend angles, while the second is to determine the attitude of the glove in 3D space. Given those two requirements we already see a separation of functions and therefore the investigation of two different groups of elements. We will begin with the Relative finger Position Detection. Three sensing techniques were considered and include IPMCs, Piezo-electric/resistive sensors and Optical sensors. IPMCs are a very attractive candidate especially for the long term since they enable both sensing and actuation at the same time and are very compliant and have high elongations and human compatible driving currents and tensions. The only downside for these systems is the intrinsic cost and overhead electronics to control and measure from the device, also its intrinsic non-linear behavior make up for its downsides. Optical sensing is also very interesting for its resolution and sensitivity characteristics but as with IPMCs cost and overhead electronics make it prohibitive in this phase of the project. This leaves us with piezoelectric sensors that are commonly spread and inexpensive. They also require very simple supporting electronics and although we can still consider piezo materials as actuators, the magnitude of the displacements and forces they produce are negligible in the macroscopic scale that we are working in. For this last reason, it was decided to use Piezo-Resistive Flex sensors (Datasheets can be gathered on the web. Links are provided in the Reference section).

The flex sensors used in this case are 4.5 in long and have a 4in sensing section. Those sensors vary their resistance linearly with bend angle. In order to increase

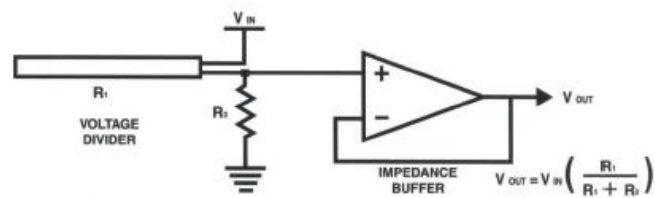


Figure A.5: Voltage divider and operational amplifier

the sensor sensitivity, a voltage divider circuit was used and R2 was identified experimentally for each finger since we wanted to maximize the variation in output voltage within each fingers flexion range. A potentiometer was used instead of R2 and a series of trials identified the optimal value to be around R2= 5K @5Vin . Later the sensitivity of the circuit was evaluated and was found to be:

$$S = \frac{V_{in}R_2}{(R_1 + R_2)^2} V/\Omega \quad (A.1)$$

Where we can see that if we see how the sensitivity varies with R2 and if we plot the variation of the Vout over the range of the sensor we get:

Where we see that; if we consider the resistance range of each finger and not the resistance range from the sensor specifications, we probably would end up with a value closer to the experimental 5K which is still in the high sensitivity region even for the full sensing range of the sensor.

The second trade that was conducted involves Local Position and Attitude Detection. Several sensing techniques have been used in the past and include Ultrasonics, Magnetic Detection, Computer Vision and Inertial Measurements. Given the material covered in the class I decided to opt for a MEMS IMU in order to have the chance to gain experience with those devices. MEMS IMUs have many more ad-

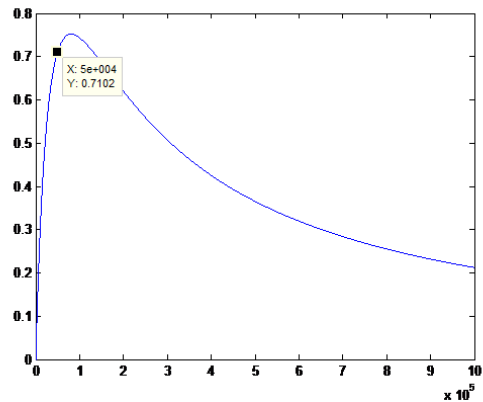


Figure A.6: Sensitivity as a function of voltage divider resistance R2

advantages to the previously listed techniques such as compactness, limited support electronics demand, and high sensitivity and low cost. The only down side is the incapacity to determine absolute position.

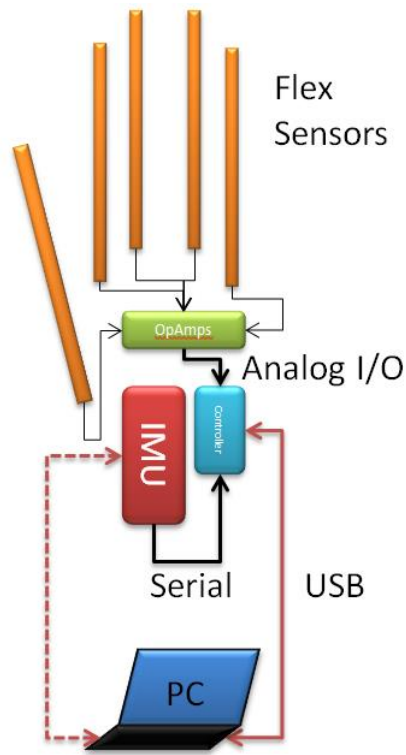


Figure A.7: Instrumented hand schematic

The reason for this is the sensing techniques used. In this implementation we will be using a three axis accelerometer, magnetometer and gyro. All of these sensors give us relative readings therefore absolute measurement are very hard to derive accurately and are affected by drift which can become very significant over time. (Datasheets for all the components of the IMU can be gathered on the web. Links are provided in the Reference section). Lastly the overall architecture of the system that allows us to acquire the data from the various sensors is depicted on the side. Such architecture relies on two separate microcontrollers one for the IMU and one for the bend sensors. Ideally we would like to merge the two on a single controller for several reasons

like interface simplicity and communications as well as power requirements. But for the moment having the two separated grants us to better understand and isolate possible issues in the software debugging phase.

A.1.3 Manufacturing Process and Software Development

The prototype is based on a leather work glove that has been outfitted with a coarse weave nylon sleeve in order to mount the supporting electronics. The bend sensors have been mounted on the vertical symmetry axis of each finger by sewing a nylon sleeve on each finger. The pocket for the sensor allows for an effective and uniform compliance of the sensor with the finger. A few preliminary test showed that the contact end of the sensor tended to slide out of the sleeve when fingers were bent. In order to mitigate this behavior, the contact pads were sewn in the sleeve, constraining the sensor. Once the bend sensors were mounted, they were initially connected to the support electronics via jumper cables which tended to get disconnected in the highly dynamic hand environment and imposed suboptimal loads on the sensor contacts end. For this reason, soldering the wires directly on the contacts was preferred. The bend sensors and the piezo-resistive film and its contacts are very susceptible to heat and the soldering process was complicated by the burning of one of the sensor (ring finger) contact paths. The sensor was pseudo repaired by using a clip to close the circuit where the path was burned out. A second sensor (index) also begun to randomly fail, and it is probably due to the initial stresses induced in the first tests with the jumper wires. Subsequently the IMU was mounted on the back of the glove by sewing it on the fabric and pre-tensioning the it in order to minimize slack in the orientation of the IMU reducing the noise in the measurements due to random vibration of the board. Also an elastic band was introduced around the body of the hand to reduce slack between the glove and the IMU. Data cables were routed around the lines of non-extension on the had to reduce the impact on mobility. Finally the main arduino board was mounted on the sleeve and a mini breadboard was mounted with double sided tape on the board itself. Once the electronics were all wired and working, the firmware for

both the arduino and the IMU was written and loaded on the microcontrollers. Appendix 1 includes all the code and comments explaining the functions of each command.

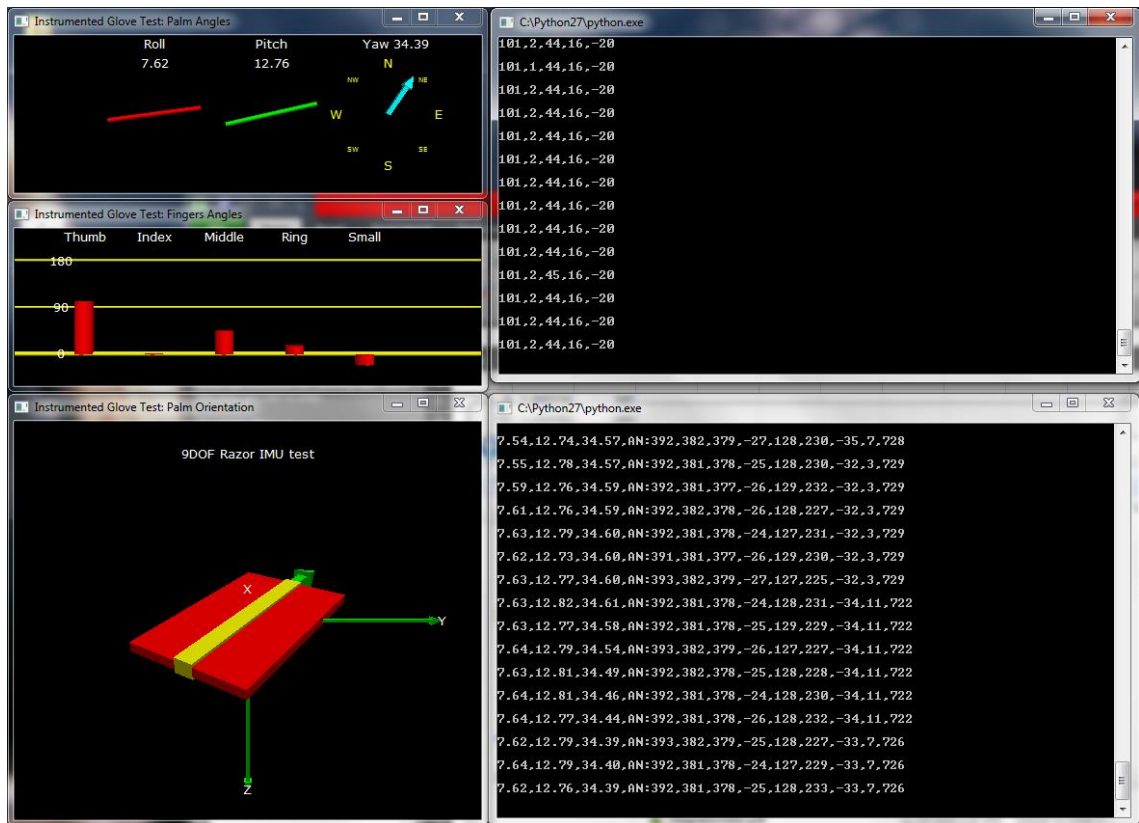


Figure A.8: Instrumented glove GUI

The final step was the development of a Graphical User Interface (GUI) for initial monitoring and evaluation of the system. Python was chosen for simplicity and VPython enabled 3D representations of the state of the hand in the virtual environment. Appendix 1 includes all the code and comments. The GUI is shown below and it comprises of 5 windows . The first gives a numerical summary of the attitude of the palm while the second shows vertical bars for the bend of each finger. The third view shows a 3D graphic view of the IMU. An updated version is currently under development where the fingers are

shown on the 3D view appropriately animated to reflect the pose and grasp of the hand. The last two terminal views show the raw data incoming from the two devices and the program logs it in a text file and allows for post processing and diagnostics of the system.

A.1.4 Preliminary Testing and System Performance Evaluation

The testing phase focused mainly in verifying the assumption made during the design phase and was triggered to determine the reliability of readings and their repeatability. The first thing that was done as soon as the system was operational was to derive calibration curves and it was soon seen that they had to be evaluated for each sensor specifically. The resistance tolerances from the manufacturer are such that a general calibration was not possible. In order to calibrate the bend sensors, the glove was donned and the hand laid flat on a flat surface. Sensor output was recorded and then each finger was bent to its maximum deflection. As we can see from the graph below the maximum deflection for each finger is different. These two points were recorded, plotted and connected with a straight line. The equation for each line was computed and used as a calibration curve.

The calibration curves were then verified by bending each finger at known angles (20-45-90-110,) and verified the reading from each sensor. Results were within 2 degrees from the expected and showed a very linear behavior. It was of interest to verify whether it was then possible to distinguish various types of grasps. Below are a few samples of different grasps of interest where we can see a very explicit difference between them.

The next phase will include the mapping of such grasps for commanding and controlling external devices or even just for gestural recognition.

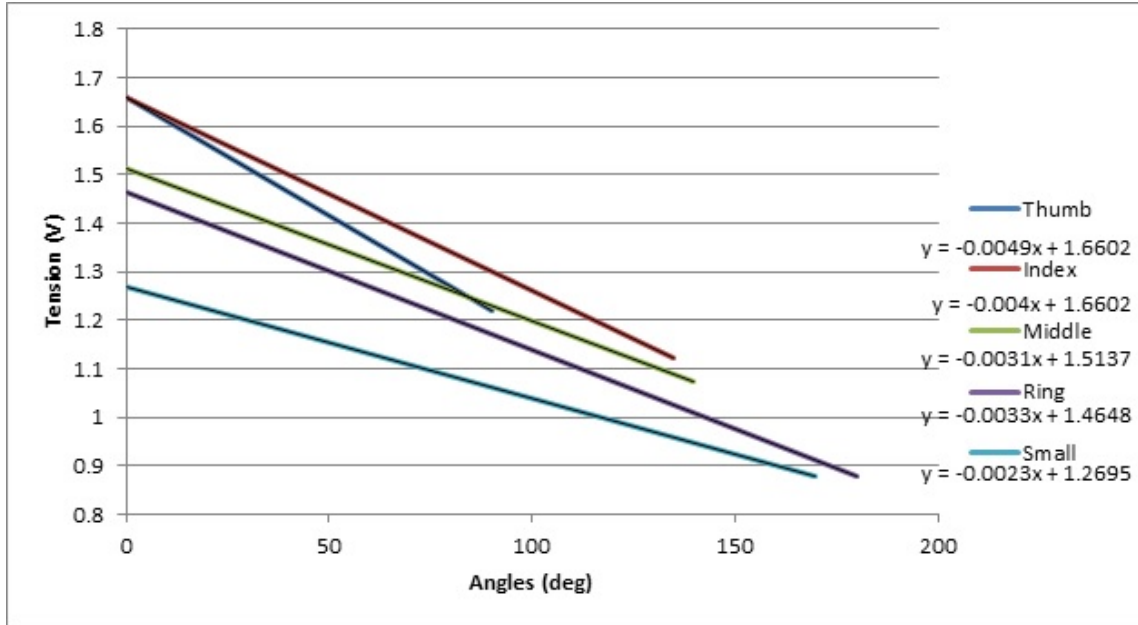


Figure A.9: Instrumented glove calibration curves

A.1.5 Conclusions

With this project we have barely scratched the surface of a much bigger research effort. We have demonstrated the ability of piezo-resistive bend sensor to measure finger bend angles and we were able to measure the attitude of the hand in a bio-compatible package. From the preliminary performance evaluation section, we have determined that results are repeatable and decoupled (Applying perpendicular pressure on the bend sensors doesn't produce a significant variation in the measured output). Sensitivity of the sensors is compatible with our application but several improvements are possible to make the system response more effective with little effort. Although in this preliminary prototyping phase we were not able to use more expensive and interesting sensors, we understand their potential and their integration in the system. If funding will eventually allow for

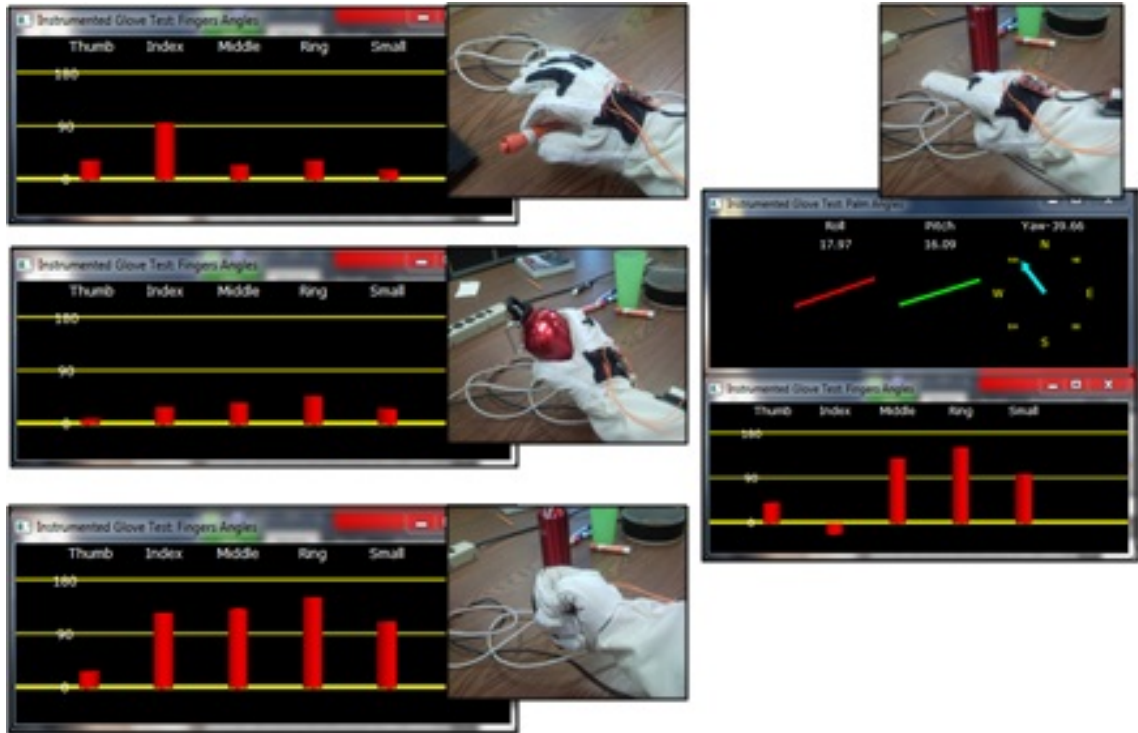


Figure A.10: Instrumented glove sample hand poses with data view

their evaluation the transition will be relatively seamless and would allow for a much more extensive comparison between architectures. In conclusion with this project we have enabled a new array of research fields that could potentially lead to a significant contribution in the space suit field.

A.1.6 Future Work

As previously mentioned, a lot of work has yet to be done on the system and it will include: Introducing piezo-resistive pressure pads on the fingers and palm to measure grasp forces. This will allow simultaneous measurement of both angles and forces and allow biometric workload and fatigue measurements if coupled with EMG measurements on the

forearm to derive metabolic workload. Integrate the system in a pressure suit glove and introduce an integrated battery and a wireless (X-BEE) transmitter to make the system wireless and standalone. Adding filtering to the IMU would lead to more accurate and less noisy measurements. It would also be very interesting to integrate accelerations to gather an estimate of the position of the hand in 3D space in a global reference frame. This will require other means of compensating for drift and will require the evaluation of its impact on the measurements and their reliability. Lastly once measurements are available we would have to translate them in useful commands. This will lead to the development of software that translates gestural commands in actions. The immediate purpose for this system is to control a remote or local vehicle, command and control in suit avionics and eventually serve as feedback control of a robotically actuated pressure glove. This last implementation will be divided in two phases; initially it will serve to offload the loads induced by the pressure glove and later enhance human performance.

A.2 Concepts for advanced pressure suit controls and displays

In August 2010, the University of Maryland and Arizona State University were awarded a grant under the NASA Lunar Advanced Science and Exploration Research (LASER) program. This four-year program is meant to enable the development and testing of both robotic technologies and operating protocols to increase the ability of humans to perform extended science missions on the Moon and Mars. This paper will exclusively focus on the development and initial evaluation of the human interfaces that enabled the above research, while leaving to a separate manuscript [61] to expand on the field trials.

We will describe in detail the MX-A and MX-B unpressurized space suit simulators, their avionics and capabilities, and how they contribute to the wider research scope. We will give an overview of all the interfaces that were developed, and describe the experimental procedures that were used to evaluate their performance and user response. Lastly we will cover plans for future work and experimentation, and will discuss observations and conclusions on the current state of the art in EVA/robotic surface scientific exploration.

A.2.1 MX-A/B Unpressurized Space Suit Simulators

In the last year and a half, two space suit simulators were designed and built at the University of Maryland with the goal of producing a simple system that would simulate the mobility constraints of a planetary pressure suit. The extensive history of space suit design and testing at the SSL with the Maryland Experimental (MX-x) series of suits[68] was branched in two different directions. The first, numbered series, refers to pressurized space suit analogues (MX-1, MX-2) while the phonetic series refers to unpressurized space suit simulators (MX-A, MX-B). The ratio-

nale behind the branching was to avoid confusion given the significant difference in the two architectures. In this work we will focus on the phonetic series of suits, leaving the pressurized suit analogues for future publications. It is our vision to converge back to a single branch and produce MX-3, a high mobility planetary space suit analogue that will be the result of all the advancements made in both current branches.

The unpressurized series of suit simulators is envisioned as a simple platform that enables the rapid prototyping of advanced controls, displays, measurement, and monitoring systems for field evaluation in an environment as close as possible to the pressurized suit environment, but without the overhead complexity associated with pressure suits. By



Figure A.11: MX-Alpha

adding physical bulk around the wearer and providing a realistic helmet and ventilation system, we are able to constrain their range of motion, tactile feedback and visual perception in a manner similar to current pressure suits. These unpressurized suit simulators will be used in this study to perform field tests of planetary surface geological exploration with and without robotic support; in a larger sense, they will also be used to identify functional requirements for future pressurized space suit systems under development in the SSL.

MX-A and MX-B were designed to accommodate a very wide range of subjects, mainly with the intent of enabling test subjects from very different backgrounds without incurring strict mission or safety protocols. It is our assumption that future planetary EVAs will inevitably be highly autonomous and unchoreographed. By allowing subjects to follow their normal field study procedures during our trials, we obtain a better un-

derstanding on what a scientist might do or need to do in planetary surface exploration activities. This approach allows us to add important operational requirements to our design processes, as well as informing future scientists on the the current technological limitations that are associated with planetary EVA.



Figure A.12: MX-Bravo

In the following sections we will describe in detail all the various subsystems that make up both MX-A and MX-B. These two suits were equipped with a backpack incorporating an on-board computer that would interface with any enabled surrounding systems via WiFi such as astronaut assistant rovers, control stations, other suits, or robotic manipulators.

At the same time they provide the user with cartographic elevation GPS-base moving maps, and voice feedback when commands are executed. The suits also include a two way radio for communications with the ground station, a digital microscope and a helmet mounted camera whose video output can be streamed over the network. A high definition helmet mounted video camera for capturing the user workspace for post analysis of the simulated EVAs was also provided, along with a head tracking system, a gestural recognition system, a speech recognition interface, and a visual feedback wristpad controller.

A.2.1.1 Pressure Garment Simulators

MX-A and MX-B share the same design approach when it comes to the pressure garment system. An outer nylon garment simulates the external micrometeoroid and

thermal insulation layer, while an internal layer acts as both a comfort layer and a backing plane for an interstitial layer of polyester stuffing. The rigidity of the external and internal nylon layers, along with the interstitial stuffing, provides a significant mobility constraint. We do not claim a high fidelity emulation of the constraints associated with pressure suits, but merely attempt to simulate the concept. The purpose of these suit simulators is not to evaluate the performance of the mobility system but, as stated before, to provide a platform for avionics and user perception evaluation.

The pressure garment simulator is also the main interface for cable routing and the backpack connection interface. The main difference between the first and second iteration of this series resides in the entry type and size. MX-A is a waist entry suit, divided in two separate sections.



Figure A.13: Donning MX-B

MX-A was our first attempt in CAD assisted design, and early versions of design techniques resulted in a smaller suit than initially anticipated. MX-B, on the other hand, is a body seal closure suit; resulting from the experience gained from its predecessor, it was redesigned completely and is able to accommodate a much wider range of subjects. The change in entry type was the result of the fact that with MX-Alpha we were unable to achieve autonomous donning and doffing, especially when it came down to closing the two zippers. With MX-B we were able to significantly increase the ease of donning and doffing of the suit, allowing for semi-autonomous operations and reducing entry time.

A.2.1.2 On-Board Computer and Audio Loop

Both suit simulators are equipped with an ASUS EeePC 1215N netbook that serves as the main processing unit in the avionics package. As we learned from the MX-2 pressurized suit, by delegating all the crucial but simple processing to embedded



systems, we can ensure a much higher level of reliability of the overall system. In the suit simulators we have taken the same approach, leaving to the on-board computer only those tasks that are either graphics intensive or that require more computational power. The netbook in these two cases serves as the main driver for the head mounted display, GPS cartographic moving map, video interface for the digital microscope and the helmet mounted camera, as well as the receiver for any other streaming video feed. It also functions as the main network hub, since all the embedded electronics are connected to it. The on-board computer parses the data from all the connected devices, translates it into commands, and sends those commands to the appropriate system be it a rover, a control station, or a robotic arm. In addition to the above features, the on-board computer is tied in the suit audio loop and can function as an audio recording system; via text-to-speech technology it also gives audio feedback on the current suit systems status and on commands received or sent. As part of the top-level avionics, MX-B also includes an advance audio loop that, through the use of a 4 channel active mixer, was able to feed to the user headset all audio transmissions from the on-board two way radio, on-board computer, speech recognition board, and audio feedback from the suit subject microphone. The other side of the audio loop includes a

4-way audio splitter that splits the microphone input to the on-board radio, computer and speech recognition board, and through a line amplifier feedback to the headset mixer loop. All audio inputs and outputs have volume controls to equalize all the audio streams.

A.2.1.3 Helmet Assembly

Helmet assemblies play a critical role in enabling a high-fidelity suit simulation, by constraining the user's head mobility and field of view. MX-A uses an 1/8 inch initial thickness vacuum formed polycarbonate ellipsoidal bubble, 6 inches tall with an 8 inch semi-major and a 6 inch semi-minor axis . Over the main pressure bubble, the assembly includes a second clear



Figure A.15: Helmet Assembly

visor, a simulated sun shield and a outer fabric cover. The main purpose of the second clear visor is to enable the projection of see-through synthetic images, and is currently under development. Given the dimensions of the desired bubble, we were unable to acquire a COTS element; therefore we were forced to build our own from scratch, resulting in a usable but optically suboptimal element. The helmet assembly encloses the user's head, but does not seal it from the external atmosphere. The helmet does however increase the risk of CO₂ pooling. In order to mitigate this issue, a set of box fans was mounted inside the helmet which ensured adequate airflow to both reduce the likelihood of CO₂ pooling and fogging of the visor. The helmet assembly severely impinges the ability to conduct unaided communications with the suit user, therefore a communication headset

was developed which will be described later.

As of now the two simulator suits share the same helmet assembly. The SSL is currently engaged in the design and assembly process of the MX-B helmet assembly, which process and description will be included in future publications.

A.2.1.4 Backpack Assembly

MX-A used a backpack assembly that was made from two acrylic bins that were cut to the desired size and joined together via a wood cross beam assembly. Adjustable shoulder straps were connected to a second wood cross beam structure that also prevented the open side of the larger, lower bin from deforming under load. The resultant shell served as mounting frame for the embedded suit avionics, and also included an outer fabric layer for aesthetic purposes. The backpack assembly was designed to be lightweight and impact resistant to protect the electronics within. The outer fabric layer included two zippers and a series of Velcro pads that covered the access ports of the backpack. From initial testing we immediately noticed that the lack of an access port wide enough to access the suit computer when the backpack was being worn was a significant design flaw. The use of two different non-connected volumes in the backpack was also inconvenient for debugging purposes. Lastly, the suit's exteriorized shoulder straps made donning and doffing the backpack very easy, but were aesthetically displeasing since they induced strange bulging of the stuffing in the lower torso and shoulder sections.

These design flaws in the first iteration were the main driver for the design of the MX-B backpack. This implementation uses a 1/2 inch PVC pipe structure that defines the backpack envelope and serves as a mounting frame for all the embedded electronics and adjustable shoulder straps. It is covered by a fabric layer that includes two main

access panels to the interior, one on the back and one on the front of the unit. The access windows allow access to the entire content of the backpack even when the backpack is in use, mitigating one of the mayor design flaws of its predecessor. Secondly it allows for quick connect and disconnect of the adjustable shoulder harness that is interior to the suit simulator. Both backpacks also served as the mounting point for the helmet assembly. This last feature will likely be removed from future implemmentations, given the constraints it imposes on the alignment and position of the suit. In later iterations, the helmet assembly will be integral to the suit and not to the backpack.

A.2.2 Advanced Controls

In the past micro-controllers and embedded systems were reserved to limited niches due to their complexity, cost and size. Today’s micro-controller based systems are smaller, cheaper and easier to use by allowing for more intuitive development environments and standardized communication protocols. These systems allow re-

searchers, engineers and students to implement rapidly and efficiently a wide array of devices that are limited only by one’s imagination and experience. MX-A was our first attempt at suit-integrated embedded micro-electronics; although very successful, MX-A’s avionics were not very reliable, and had limited capabilities. MX-B was the result of this experience, where a much better understanding of the Arduino microcontroller platform and its IDE (Integrated Development Environment) resulted in more complex, stable and

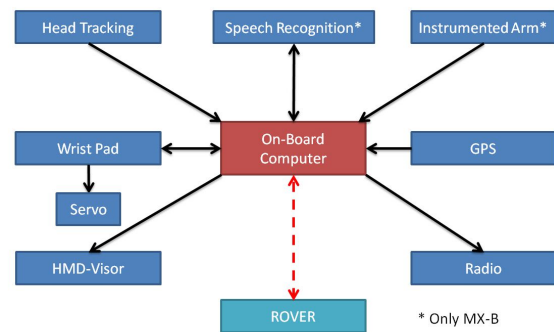


Figure A.16: Avionics Architecture

performing interfaces. This section will describe in the detail the interfaces that were developed, integrated and tested in the MX-Alpha, and Bravo suit simulators. Figure A.16 gives a general overview of the two suits' avionics architecture. The figure identifies additional functionality incorporated into the MX-B avionics package; we expect this trend to continue in the future.

A.2.2.1 Multi-Functional Wrist Pad

The main control interface for both the Alpha and Bravo platforms resided in a wrist mounted button pad. The first iteration of the wristpad included a series of conductive pads connected to the digital readouts of an Arduino Mega board. To limit noise and false positives, all the in-

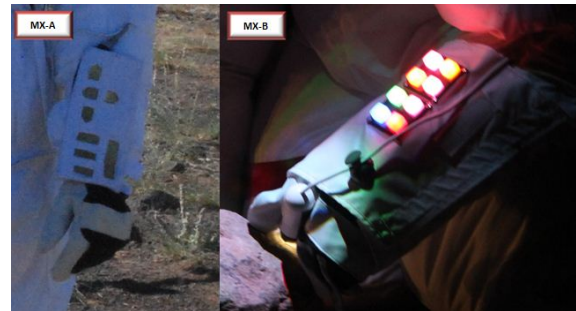


Figure A.17: MX-A and MX-B Wristpads

puts were grounded through pull-down resistors. The wristpad was mounted on the left wrist, while a modified right glove included two conductive pads on the index and middle fingers connected to two different digital output lines. Pressing a finger on a button would close a specific digital circuit, allowing the Arduino to identify if a button was pressed, and with which finger it was pressed. Finger differentiation was obtained by pulsing one finger high and the other low, and reading from the digital inputs of the board. Our implementation was restricted to two fingers since there was no need for more input commands in our specific application, but the principle applies to any arbitrary number of sensed fingers, and is limited only by the desired sampling rate and microcontroller performance. The system had its advantages and disadvantages: it is a very reliable concept, since it

does not allow for accidental button press, and due to its simplicity it can be packaged in a very low profile wristpad. On the other side, it does not allow any sort of visual feedback, and it is sensitive to dust and corrosion since the contacts are exposed and the finger contacts are located right on the finger tips, which are usually subject to wear and contamination. This last issue was the main driver for moving away from the concept. During field trials we experienced several failures of the finger contacts which reduced the reliability and user confidence in the system.

For MX-B, we expanded on the initial wristpad concept by building a much more advanced interface that includes an analog thumbpad, nine buttons, eight of which include an RGB LED; these can be commanded to any desired color, allowing for visual feedback of the button press. The ability to color code each button independently opened the doors for a layered user interface, allowing a much broader control authority. In the second iteration wristpad, accidental button presses are possible, but they are mitigated on the procedural level. The wristpad layered interface requires a button sequence to be pressed in order to execute a command; following each button press, the user receives an audio message that relays what button was pressed and what it does, along with visual feedback on the wrist pad. For example, initially the wristpad is in the top menu layer with all the buttons lit white, except for the stop command, which is a dedicated button and is always red. The user can now select the a mode of operation; in this example, we will assume the user wishes to teleoperate a rover via the embedded thumb pad. The user will select the thumb pad button and the computer will feed to the local audio loop a synthesized voice saying “Thumb pad mode enabled”. A series of buttons will light up. In this specific case, the pressed button will turn green, the “send commands” button will turn orange and blink while the “exit” command will turn blue and the “stop” command will remain red. The

remaining buttons will not be lit, indicating that no functions are associated with those in the currently active mode. At this point, no commands are sent to any paired system; in order to begin transmission, the user will have to press the “send commands” button, at which point that button will turn off and the “stop sending commands” button will start blinking orange. The user will be notified by an audio message that reads “Thumb pad mode active”. From now on commands are sent to the paired system until either the “exit” command or “stop” command is pressed. Two buttons never change location: those are the “exit” and the “stop” buttons. The “stop” command is the only button that doesn’t require any secondary press; regardless of the mode, it immediately sends a signal to all paired systems to stop. Audio feedback is also provided. The “exit” command on the other side is enabled only after a mode has been selected; when pressed, it returns the wrist pad to the top level layer and interrupts any transmission of commands. This approach was very successful in the field, since any accidental button press was readily recognized; there were no instances of unwanted commands being sent to any paired system. It was also observed that the intrinsic stiffness of the buttons, along with the position of the wristpad, made accidental presses infrequent.

A.2.2.2 Head Tracking

MX-A had a very basic head tracking interface that relied on a single two-axis MEMS accelerometer mounted on the communication and visor headset. The accelerometer allowed us to measure pitch and roll angles of the user’s head, and translate those to forward and lateral velocities that were then used to control rovers. In MX-B we have brought the concept further by replacing the accelerometer with a 9 DOF MEMS IMU (3-axis accelerometer, magnetometer and angular rate sensors). The IMU includes a mi-

controller that filters the data from the three sensors via an extended Kalman filter, and then communicates over serial to the MX-Bravo's on-board computer. The filter and the dual redundant sensors allow us to

have a much smoother and more accurate reading of the pose of the user's head in the Earth's magnetic reference frame. The IMU is also able to evaluate the pose of the head in a highly dynamic environment; it allows the user to access readings from the three sensors separately (before filtering and mixing is executed) allowing for a rough estimate of the dynamics of the user's head.



Figure A.18: MX-A and MX-B head Tracking System

A.2.2.3 Gestural Control Interface

One of the current research topics at the SSL is robot-actuated space suits. The first step towards the implementation of an active exoskeleton or “powered suit” is to better understand the dynamics of the human inside the space suit, the limitations that the suit imposes on the person, etc. Current measurement systems that allow for such an analysis either require an extensive infrastructure (e.g., arrays of motion tracking cameras), and/or do not grant accurate and repeatable measurements (e.g., resistive bend sensors). In a novel approach to this problem, the SSL has developed a mea-



Figure A.19: Arm pose measurement System

suring system capable of determining the attitude of the human arm and the bend angle of each finger on the hand while still remaining portable and nonintrusive. This initial prototype is purposely restricted to the arm for simplicity, but will be extended to the entire body once the initial evaluation of the concept has been conducted and the appropriate revisions on the embedded electronics are finalized. As of now the measurement system includes a microcontroller, three 9 DOF MEMS IMUs located each on the back of the hand, forearm and upper arm, and 10 piezo-resistive bend sensors (two for each finger). By initially measuring the lengths of the subject's arm sections we are able to determine the position and pose in earth-magnetic reference frame of the subject's arm and hand. Currently, the embedded micro-controller is only responsible for parsing the data from the various sensors and sending it to the MX-B on-board computer, which executes filtering and data manipulation.

A.2.2.4 Speech Recognition

Speech recognition was not implemented on the MX-A platform, but it was initially introduced earlier in MX-2. In MX-B we added an embedded speech recognition board that communicates to both the embedded on-board microcontroller and suit computer. Its main purpose is to provide the user with a more natural primary control interface while still allowing the wristpad to be used as a redundant interface. This subsystem is still in the development and evaluation phase.

A.2.3 Advanced Displays

MX-A and MX-B shared the same head-mounted display. Based on a stripped down z800 visor from eMagin, we adapted the visor mount to include a small radio controlled

model servo to swivel the visor in and out of the user's field of view. To actuate the servo, the user is required to input a button sequence on the wristpad, and the Arduino microcontroller commands the servo. The user also has the ability to incrementally tilt the visor up and down to adapt it to his or her preference. The same commands have been implemented via speech recognition, therefore allowing the user to adjust the head-mounted display without pressing any buttons.

The visor is able to display a maximum resolution of 800x600 pixels at 60Hz; although compact, the visor assembly does not always fit inside the helmet depending on the suit subject. This interface is extremely useful, but it is not likely to be an option in future implementations. The visor, when down, completely obstructs the user's field of view, resulting in an unacceptable loss of situational awareness.

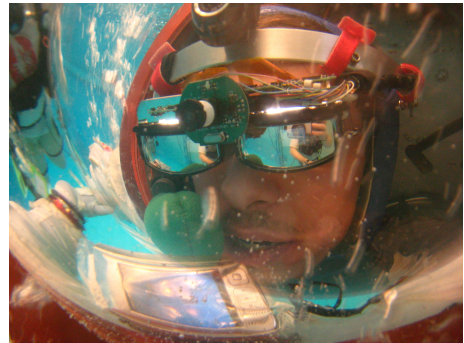


Figure A.20: Modified z800 visor

Also, if for any reason the visor swivel would fail to function, the visor would remain locked in its last position, possibly compromising the astronaut's ability to see outside. Current effort is concentrated in the development of a similarly capable system, but with a different approach. Instead of using a head mounted display, we are in the process of testing an in-helmet projection system that would enable the astronaut to pull down an external helmet visor to see the projected images, while at the same time retaining the ability to see outside. The synthetic images would be displayed as overlays on the physical world, therefore limiting the overall distraction and loss of situational awareness. The authors are currently engaged in developing this two-way mirror and micro-projector based head-up display that will be integrated in the next iteration helmet assembly.

A.2.3.1 Night Operations and Lighting

If we were to consider extended lunar missions, we would have to take into account that for half of the time the crew will be required to operate in the dark during the 14 days of lunar night. That would be true as well for Mars, although Mars has a day/night cycle much more similar to the familiar Earth cycle. In order to enable night operations with MX-B, we incorporated helmet mounted light sources and a flashing beacon. We evaluated this



system in a night run in the Arizona desert, where a moonless night allowed us to verify that the brightness levels provided by our LED-based system were

Figure A.21: Night Operations and lighting system

sufficient to enable science operations in the dark. The MX-B lighting system is composed of two modules, each including an array of three high intensity RGB LEDs. By turning on all three LEDs we obtained a cold white color light beam approximately 40 degrees wide. The lights are enabled, disabled and color tuned from the wristpad or speech recognition system. When the lights are enabled, a beacon located on the top of the backpack also begins to flash; the flash color and frequency can be modified, to allow an easy identification of the crew members in the dark. Lastly, the light modules are housed on both sides of the helmet on a swivel mechanism, allowing the user to turn the beam wherever necessary. The right light module also includes the two helmet mounted cameras which are rigidly mounted on the assembly, therefore providing light for the video devices as well.

A.2.4 Rover Control Interface Evaluation

During the latest Desert Field Lessons in Engineering and Science (Desert FLEAS) field trials, we had the opportunity to run a pilot study to address the usability of each rover control mode that MX-Bravo offered. A simple course, depicted in figure A.22 was set up in the Arizona desert where eight subjects participated in the experiment. The RAVEN rover was the controlled platform to be run across the course. Measurements taken were course completion time, videos and user feedback. Each subject was briefed on a single control interface, and then asked to perform the course as fast as comfortable. Once each subject ran the course in a given control mode, we would repeat for the next control mode.

Three control modes were tested in sequence. The first mode was the wristpad, the second was head tracking, and the third was gestural control. No additional training was provided for any of the control modes, so each subject was at a similar point in the learning process. The figure below shows the completion times that were recorded during this preliminary test. All test subjects recorded higher

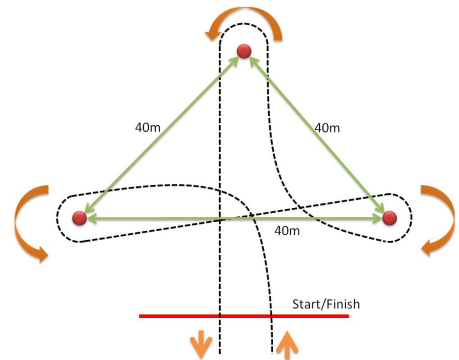


Figure A.22: Course diagram

times in the initial wrist pad run, and on average, their times were better with the head tracking than with the gestural control. We believe that the learning process played a significant role in this preliminary test; therefore randomizing the controller sequence and granting some familiarization time with all the interfaces will be a must in future implementations of the test. Statistically we are not able, with this preliminary data set, to draw any significant conclusions; qualitatively the rover was easy to control in all three

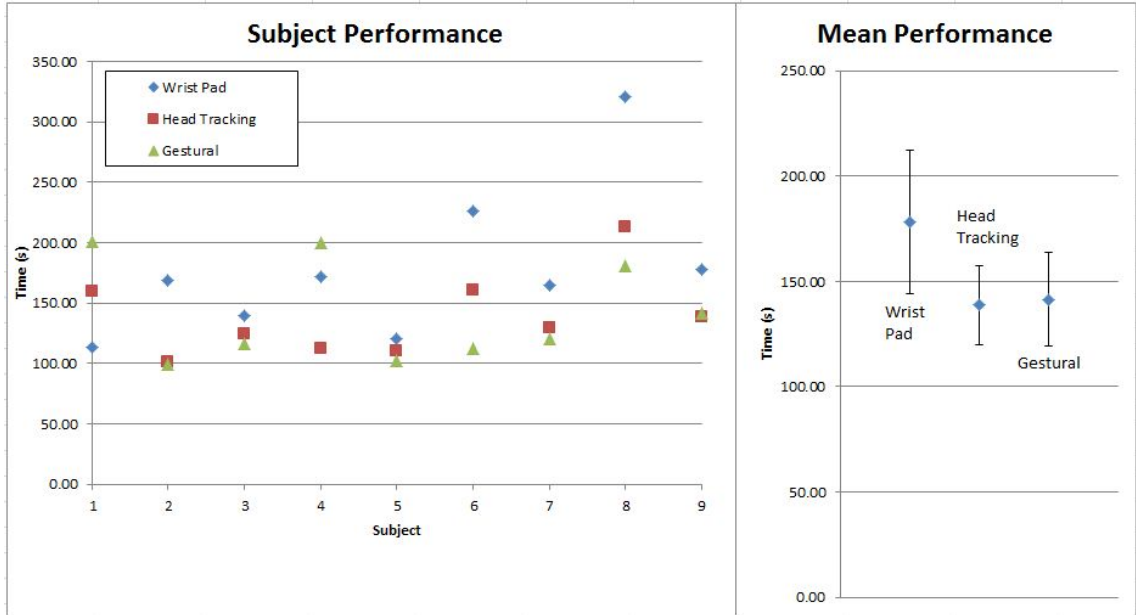


Figure A.23: Experimental results: Execution times

modes tested.

User feedback was perhaps the most valuable of all the collected data elements. Users were very helpful in giving us feedback on the various control modes. In summary, the general consensus was that all three modes were perfectly feasible, but subjects tended to like the gestural control strategy best, followed by head tracking and (lastly) the wristpad. Subjects thought that the commanded output from the wristpad was very different from the one in the latter two modes. From a software perspective, the output commands followed the same translation process for each interface, and the control range for each interface was also similar. Following this initial pilot test, a more extensive study was conducted. 15 test subjects, 12 males and 3 females participated in a more rigorous study where the three interfaces (wristpad, gestural and head control) were compared. Test subjects had an average age of 22.2 and an average experience with videogames of 4.2 on a scale from 1 to 7. All were currently enrolled students.

A similar course to the previous test was implemented at the University of Maryland. Three obstacles were placed in a corridor and a slalom course was identified by arrows taped on the corridor floor A.24. Subjects were asked to run the course as fast as comfortable while standing in a predetermined location near the control station. Since the RAVEN platform was not available, a simpler but similar rover was built. Mini-RAVEN is a three wheel vehicle with two powered front wheels and a free-swiveling third wheel. The rover uses an Arduino based micro-



Figure A.24: Indoor test course and mini-RAVEN

controller with an X-Bee module for wireless serial communications. The rover was programmed to emulate the input-output data structure used on the RAVEN platform, therefore eliminating the need to modify the interfaces control software. Its speed and handling performance are comparable to it's "bigger brother", but since the data acquired during this test is not going to be compared to the data acquired in the field trials, the difference in platform will not impact our conclusions. The data acquired included course completion times, obstacles hit (either corridor walls or markers), a Cooper-Harper evaluation, a NASA Task Load Index (TLX) survey, and finally an analytical hierarchy process (AHP) survey to rank pair-wise comparisons. As learned from the previous test, the interface sequence was randomized between subjects; all subjects were asked to perform up to three

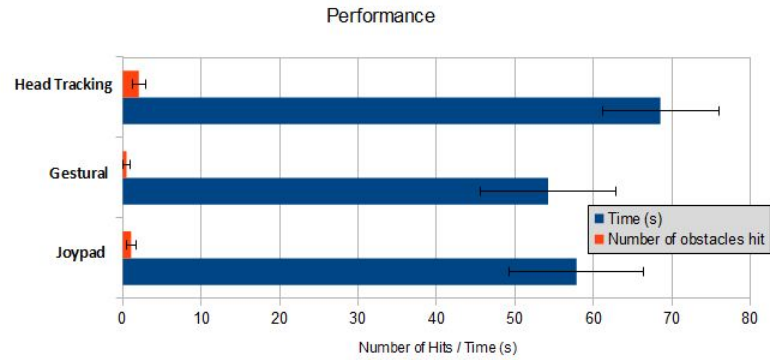


Figure A.25: Mean course performance

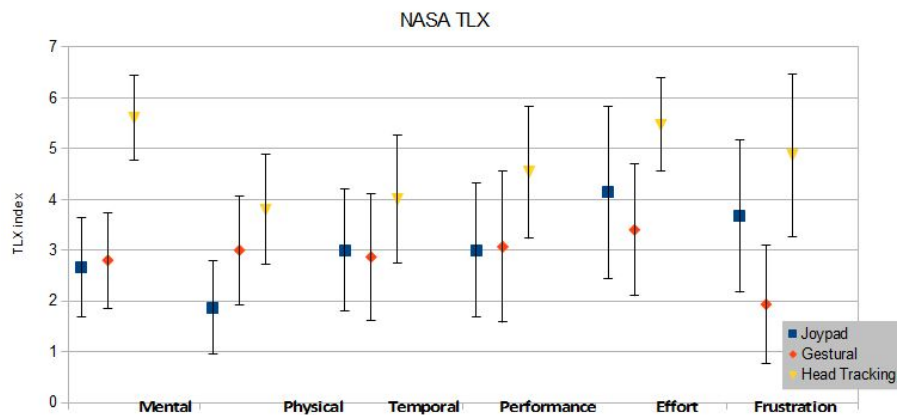


Figure A.26: NASA TLX test results

runs with each interface to mitigate learning curve and order bias on the acquired data.

Data was acquired for all runs, and outliers were excluded from the post processing. The results from the test are summarized in the figures A.25, A.26 and A.27 below.

The last metric that was evaluated was a simple AHP analysis that, through three pair-wise comparisons, gave a ranking of the three interfaces. Subjects were asked which interface they liked better overall, and the results show a wide preference for the gestural control (60.88%), followed by head tracking (25.68%), and lastly by the joy pad (13.44%).

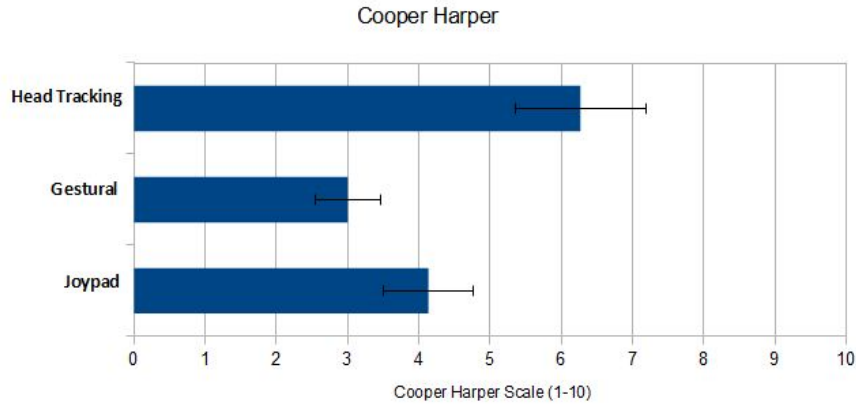


Figure A.27: Cooper Harper

The result is probably biased by the novelty of the interfaces over the more standard joypad.

Otherwise, from the summary above, we can see that overall the users preferred and had better performance with the gestural control interface, followed by the joypad, and finally the head tracking system. All the metrics except for the AHP show this trend. Unfortunately, the limited population of test subjects and relatively wide spread of data points does not allow us to make a statistically conclusive assessment. It is however safe to state that the user response to the gestural control interface was very positive, and at least comparable to the more standard joypad, and that such control interfaces can be very promising for future applications.

A.2.5 Conclusion and Future Work

This work has greatly expanded our capabilities in, and understanding of, space suit avionics and command and control interfaces. The next step forward to continue this effort will be modifying the MX-B platform for neutral buoyancy operations for micro- and partial-gravity simulations in the SSL Neutral Buoyancy Research Facility, as well

as finally continuing the development of the third iteration of our pressurized series of suits. The birth of the MX-3 space suit analogue will not imply the conclusion of our work with our simulated suits branch; instead both will progress in parallel given our firm belief that there are a series of scenarios where the two suits branches can contribute significantly in different ways. MX-A and MX-B are ideal platforms for an initial concept test; they allow for a very rapid turn-around time when developing innovative avionics and user interfaces, as well as simple integration of subsystems. They open the doors for more extensive experimental studies where we do not have to worry about custom fitting a pressure suit on every potential trained subject, and all the safety concerns associated with pressurized suit operations. On the other side, their reduced fidelity does not allow us to draw significant conclusions when it comes down to scenarios where suit pressurization is a critical factor. We will compensate for the reduction of fidelity by providing a more complex, pressurized platform that will fill in the gap in the previous studies.

MX-3 is expected to be operational by the end of 2011, and will be designed to be a highly instrumented, lightweight planetary space suit analogue to be used in our future Desert FLEAS studies, simulating lunar and Mars EVAs in Earth analogue sites like parts of the Arizona desert. MX-3 has also been envisioned to be compatible with neutral buoyancy operations, although this latter feature might require a few modifications on the platform, such as incorporation of ballast systems. In future research, we plan to incorporate into MX-3 all the embedded systems that are currently prototyped in the MX-Bravo platform, and to expand upon the IMU based measurement system with extended metabolic workload studies, by adding on electromyographic (EMG) sensors, oxygen uptake (VO₂) measurements, a heart rate monitor and pressure sensors on the hands and contact points between the suit and subject's limbs. The proposed measurement



Figure A.28: The SSL 2011 Human Factors Team

system may very likely open the doors for a more in-depth understanding of suit-demanded workload on astronauts in a high fidelity simulation environment. Three more Desert FLEAS test series are planned in Arizona analogue sites through 2014, and the SSL is always receptive to additional opportunities for analogue studies.

A.2.6 Acknowledgments

We gratefully acknowledge the support of the NASA LASER program, NNX11AB42G. The authors wish to thank Dr. Kip Hodges, Dr. Srikanth Saripalli, Kelsey Young and all the ASU SESE students that participated to the 2010 and 2011 D-FLEAS tests. Our most sincere thanks also go to the entire SSL, especially William Cannan and Justin Brannan for their hard work in this research.

Appendix B

Concepts for advanced high mobility pressure suits

B.1 Introduction

Since the 1970's, the Space Systems Laboratory (SSL) has been involved in research into advanced EVA technologies and operations. Throughout the 1980's and the first half of the 90's, much of this research was performed at the NASA Marshall Neutral Buoyancy Simulator (NBS) using Apollo-era A7L-B and Shuttle EMU suits. Results from this research included quantification of EVA capabilities in structural assembly, leading to the Experimental Assembly of Structures in EVA (EASE) flight experiment in 1985.[5] Subsequent research focused on the interactions between EVA crew and robotic devices, including assembly aids, free-flying support platforms, and full telerobotic assembly and servicing systems.[85]

Although the University of Maryland opened the Neutral Buoyancy Research Facility (NBRF) on campus in 1992, experimental activities still went on at NASA Marshall to get access to suits for advanced EVA research, including biomedical instrumentation and human/robot cooperation. With the closing of the NBS in 1997, access to NASA space suits became impossible for research purposes, and the SSL was faced with the choice of dropping this area of research or developing independent capability in pressure suit design, fabrication, and operations. Despite the formidable challenges, the choice was made to develop an in-house pressure suit to enable continued SSL EVA research.

The first Maryland experimental suit (MX-1) was conceptually a scuba dry suit with a hard torso and attached helmet. The dry interior allowed the development and

testing of biomedical instrumentation and advanced controls and displays. A hard upper torso (HUT) was designed and laid-up by hand out of room-temperature curing epoxy and woven fiberglass cloth. Soft goods, for the arms and lower torso assembly, were special-ordered using dry suit technology from sealed neoprene fabric. A rear entry architecture was adopted for entry and egress, along with sufficient helmet volume to accommodate helmet-mounted and head-mounted displays.

The use of neoprene for the soft goods precluded pressurization above vent pressure, as the low elastic modulus of the material allowed substantial inflation under positive pressurization. Experiments indicated that even the normal water pressure gradient of 70 Pa/m (0.44 psi/foot) provided more than enough pressure differential to inflate neoprene soft goods above the level of the suit back-pressure regulator near the subject's waist. This substantial inflation volume led in turn to uncontrolled shifts in suit buoyancy and balance, proving to be difficult to overcome without elaborate work-arounds affecting suit comfort and safety.

These early results resulted in a redesign, incorporating traditional pressure suit soft goods concepts in the second-generation MX-2 suit. A three-layer system was adopted, using heat-sealed urethane-coated nylon fabric as the innermost layer as pressure bladder, rip-stop sewed nylon fabric as a restraint layer, incorporating convolutes to allow free motion at the wearer's joints, and an outer layer of ripstop nylon fabric. The outermost-layer, which would be the thermal/micrometeoroid garment (TMG) in a flight pressure suit, was renamed as the integrated ballast garment (IBG) for the MX-series of suits as it incorporated a number of external pockets with restraint straps to allow the insertion and extraction of lead ballast masses to reach neutral buoyancy for underwater testing.

The MX-2 reached operational status in 2005, and has been used extensively for

neutral buoyancy simulations at UMD until the current time. Its operational utility has been hampered by a number of factors, including design errors which placed the HUT's scye bearings too far apart for comfortable dual-handed operations, and a HUT lower extension which is too long to support adequate waist flexibility in the suit. Also, as a suit intended from the outset to be dedicated to neutral buoyancy research, the overall mass of MX-2 with an average test subject is of the order of 700 pounds, which is clearly infeasible for 1-g laboratory or field testing.

For all of these reasons, the MX-2 has been retired in favor of the next-generation MX-3 pressure suit, currently under development in the SSL at UMD. The overarching goal of MX-3 is to provide comfortable operations through resizing of suit components while maintaining an overall weight low enough to allow extended testing in 1-G operations. In parallel, a new series of suit simulators, starting with MX- α , are being developed to support outreach and extended surface test operations which do not require full pressurization, to facilitate greater access to testing with reduced infrastructure requirements.

B.2 MX-3 Architecture

B.2.0.1 Soft Versus Hard Joints

Spacesuit joints have historically been constructed using some combination of flexible fabric elements and hard, bearing-interfaced elements, each of which carry unique advantages and disadvantages. Hard joints typically allow for greater mobility and lower joint torques, as suit volume remains constant throughout the full joint range of motion. However, hard suits are susceptible to programming issues, meaning that the location and orientation of bearings enforce a preference for certain paths between various suit configurations, which may not be optimal for the performance of certain tasks. Fabric

joints typically tend to produce higher joint torques, which results in greater exertion on the part of the suit occupant, and may preclude the occupant from moving into certain configurations. Soft suits also tend to be lower mass than hard suits, and their collapsible nature allows for expanded storage options. Soft suits are favored for launch and entry, as they are more conformal and mobile than hard suits in depressurized operation.

Because of the desire to minimize mass and to create a suit which will be equally appropriate for all roles, including launch and entry, the MX-3 will be an all-soft suit. The MX-3 will utilize unique two-degree-of-freedom joints in order to minimize joint torques, with the goal of achieving mobility and range of motion comparable to hard suits.

B.2.0.2 Joint Degrees of Freedom

Traditional fabric suit joints are typically single degree-of-freedom, and fabric joints in deployed suits have been limited to so-called "degree-and-a-half" of freedom, in which a joint is supplemented with a sleeved cable. This creates a continuous line of non-extension passing over the joint, allowing the joint to bend side-to-side while maintaining a constant-length centerline. Such degree-and-a-half joints were used on the Apollo A7LB suit, the modified Mark V, and the 1994 Sokol suit [86]. The joint torques in this second, cable-permitted degree of freedom are significantly higher than those in the primary, gore-convolute axis; hence the "degree-and-a-half" designation.

The MX-3 will instead use a unique joint design, which allows for two full-performance degrees of freedom in the shoulder, elbow, wrist, hip, knee and ankle.

B.2.0.3 MX-3 Advanced Mobility System

The top-level design requirements generated for the MX-3 are:

- MX-3 shall be an all-soft suit capable of being deployed in:
 - Planetary Surface Operations
 - Microgravity Operations
 - Launch and Entry Operations
- MX-3 must eliminate joint programming
- MX-3 must be modular and sizable to accommodate occupants ranging from the 95th percentile American male to the 5th percentile American female

Additionally, MX-3 must be low cost, and easily implemented and tested, in order to accelerate the MX-series suit development cycle.

In attempting to identify the type of soft joint best suited to use in the MX-3, a series of existing and new, hybrid joint types were prototyped and tested in-house to subjectively assess joint torque and range of motion. Each joint was constructed using a traditional, two-layer (pressure bladder and restraint layer) design with the exception of one iteration of the toroidal joint which utilized a Single Wall Laminate layer. The following joint types were tested:

- Convolute
- All-Soft Toroidal
- Sliding Cable Toroidal
- Dual Sliding Cable Toroidal
- Helicoidal Restraint Toroidal
- Gathered Fabric Helicoidal



Figure B.1: Examples of Experimental Arm Sections

These joints types were prototyped and tested in a glove box, where preliminary qualitative data was acquired, but no rigorous torque measurements were taken. By testing the various joints in the glove box and in a custom-built test stand, we were able to determine, at least at a subjective level, whether each joint type would be a suitable candidate. Below are summarized the notes acquired in preliminary testing of the various joint types.

Convolute This was the first type of joint that was tested. With heritage from the A7LB and Shuttle EMU, it presented low torques and limited hysteresis, and is relatively

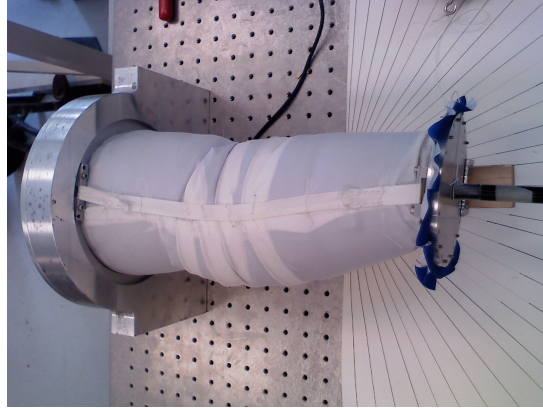


Figure B.2: Convolute

simple to manufacture. The main concern regarding this type of joint was that it is not axially symmetric; therefore, it is unlikely to be a good candidate for a two-DOF joint. This joint test also highlighted the importance of closely indexing the pressure bladder and the restraint layer, since most of the torques and hysteresis were due to a misalignment of the two layers.

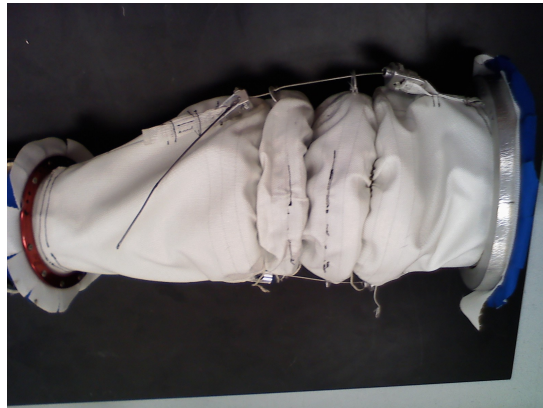


Figure B.3: All-Soft Toroidal joint

All-Soft Toroidal, Sliding Cable Toroidal and Dual Sliding Cable Toroidal

The toroidal joint is based on Elkin's design for a high mobility soft joint [76]. Our

implementation simplified his design by eliminating the hard rings that are characteristic of this type of joint. The hard rings were replaced by a series of circumferential mylar cables that constrain the geometry of the joint by recreating virtual small hard rings when the joint is pressurized. The cable pass-throughs were sewed at half length between the mylar restraints. The body of the joint started from a flat rectangular section that was then constrained to its final shape through the use of cables.

This joint was particularly appealing, since it is axially symmetric, and, through the implementation of a sliding cable system, it delivers an additional limited degree of freedom. In the first implementation of the joint, considerable asymmetry in performance was noted between the two rotation axes (motion on the secondary axis was enabled by means of a bicycle brake cable, and presented noticeably higher torques than the primary axis). Secondly, the joint was unstable. The gores tended to assume a bent state, even when the joint was kept straight. This led to the insight that in order to achieve a high-performance joint, the joint had to be constructed in a marginally stable regime in order to minimize joint torque. Overall, our implementation of this joint did not approach the performance of the Shuttle EMU arm, and further modifications were required.

The second prototype of the toroidal joint attempted to mitigate wear on the cable restraints and the sliding cable. In testing this prototype, friction in the cable sleeve was clearly the predominant factor in the higher joint torque of the secondary degree of freedom. However, this friction appeared to mitigate joint instability, and hysteresis was minimal. Indexing once again was a major reason for the instability and poor performance of the joint. It was at this point that a single-wall laminate design came under consideration, as a means of eliminating the indexing issue.

A single-wall laminate toroidal joint was prototyped next. The same cable restraint

system as before was implemented over a two-layer, heat-sealed urethane-backed nylon fabric construction. This design resolved the indexing issue, but made it difficult to restrain the sliding cable. Sewing into this single wall laminate design was not an option due to its pressure-bladder role; therefore, other solutions had to be investigated. Two additional circumferential mylar cables were introduced, and the sliding cable was mounted on these two extremity restraints. This implementation mitigated some of the torque issues, although the joint still did not approach the performance of the EMU arm, especially in rotation where the sliding cable was engaged.

A final joint concept that was not implemented in testing was a dual sliding cable toroidal. With this design, we expect to see symmetric performance on the two degrees of freedom, as well as reduced friction in the cables, as the tension on each cable will be halved.

Helicoidal Restraint Toroidal and Gathered Fabric Helicoidal

From the testing performed on the various iterations of toroidal joints, it became apparent that sliding cable systems present an inherent limitation to the mobility and reliability of suit joints. Sliding cables apply a great deal of wear and tear to soft goods and restraints. In order to eliminate the sliding cable, while still achieving a secondary degree of freedom, a different solution had to be implemented. Basic structures concepts can be used to anticipate how pressure loads will be transferred to the restraint layer on a straight pressurized cylinder:

$$\sigma_l t 2\pi r = p \pi r^2 \rightarrow \sigma_l = \frac{pr}{2t} \quad (\text{B.1})$$

$$2\sigma_h t dx = p 2r dx \rightarrow \sigma_h = \frac{pr}{t} \quad (\text{B.2})$$

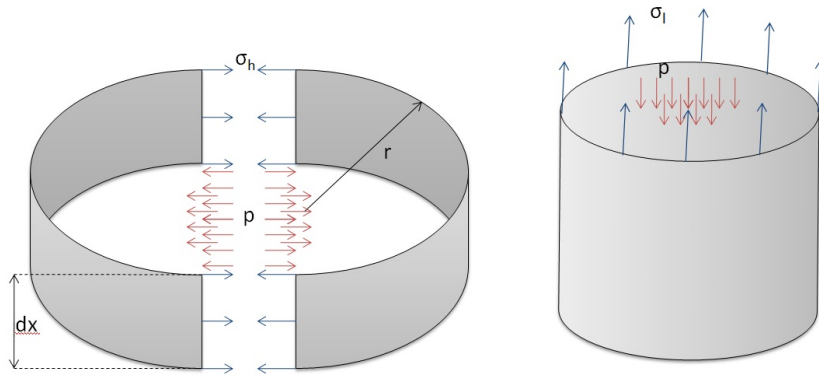


Figure B.4: Thin wall pressurized Cylinder

Where:

p =net pressure

r =radius of the cylinder

t =thickness ($r > 5t$)

σ_l =longitudinal stress

σ_h =hoop stress

Clearly, real joints are not straight cylindrical sections, and the presence of gores, convolutes, and other patterns adds complexity to the physical system, which is difficult to accurately capture in a model. However, this simple analysis makes it clear that there are several alternative ways of longitudinally constraining the joint. A pressurized joint reacts with both circumferential and longitudinal tension. While restraint lines are traditionally straight lines along the sides of the joint and intersect the constrained bending axis of a single-DOF joint, a joint does not necessarily need to be restrained in this way. A helical restraint line, if constrained from "unwinding", can be just as effectively employed to constrain the length of a limb segment; moreover, such as helical restraint will allow for the excess fabric required by a joint to bend with approximately constant volume. Unlike

straight restraint lines, however, a helical restraint line does not inherently constrain the joint to a single degree of freedom – an attractive feature allowing for flexibility in the location and number of degrees of freedom offered by the suit.

Based on this experimental approach, an architecture relying on two degree of freedom joints was conceived. Considering the anatomy of the human arm, seven rotational degrees of freedom are apparent: three in the shoulder, one in the elbow, one roll degree in the forearm, and two in the wrist. This represents a redundant manipulator: one more degree of freedom is present than is strictly necessary to determine the position and orientation of the hand in space, thus allowing the human arm a freedom to reach the same "end-effector" position and orientation through multiple configurations, and eliminating singularities in the kinematics of the manipulator. The kinematic analysis of the human arm can be applied to the leg as well without loss of generality, as the leg possesses the same degrees of freedom and a similar morphology, albeit with different joint limits.

To replicate the DOFs of the human limb anatomy in a pressure suit, an equal number of mobility elements are necessary. The traditional approach to meeting this requirement relies on a series of soft joints and hard rotation bearings. Such hard bearings, especially in the shoulders, present a problem for launch-and-entry suits, which must be comfortable when worn in 1 g in a supine position for long periods of time, and through higher g -loads during launch. Additionally, hard bearings tend to introduce a degree of programming to suits, as the occupant may need to first rotate the bearing through a large angle in order to bring a primary degree of freedom into alignment with the desired direction of motion. While programming and other limitations of traditional suit design have not been especially problematic during the extensively-rehearsed and deliberate EVAs of the Shuttle era, suit limitations were seen to impair mobility during the Apollo lunar

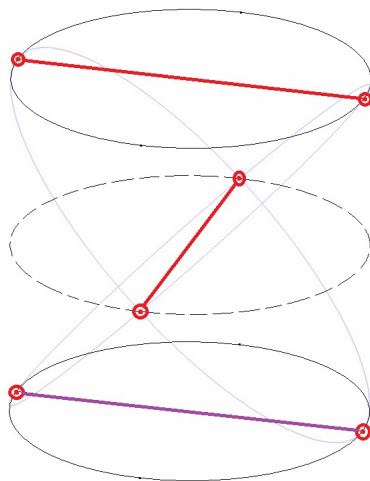


Figure B.5: Helicoidal Restraint Diagram

surface EVAs. (It has been argued that walking is not the optimal gait for humans in lunar gravity, even in a "shirtsleeve" environment[87], although the observed preference for two-legged hopping and loping over running is likely a result of the damping and spring-like effect of the A7L suits, as much as a result of the gravitational load.)

The MX-3 will introduce the use of two-DOF mobility units, constructed using warped helicoidal lines of non-extension. This constraint scheme will take advantage of the suit pressure differential in order to maintain the desired shape by using the radial and longitudinal pressure forces to "push out" the restraint lines. The helicoidal restraint lines depicted in the diagram are hinged in the highlighted locations, and allow two symmetric preferential directions of motion on the topmost ring. This particular type of joint will be symmetric over the plane passing through the centerline of the joint and the two hinged locations on the rings.

As the diagram in Figure B.2.0.3 illustrates, we can identify the two extremity rings and a "virtual" ring in the vertical center of the joint that virtually divides the

joint in 4 identical sections. Those identical sections are offset 90° from each in pairs, and mirrored over the vertical plane. The restraint of the joint is a fundamental element of this concept, but the pattern design is most likely equally important. Several attempts have been implemented and more are currently being considered for the joint soft pattern; the toroidal joint pattern or a modified convolute are two examples of implemented solutions. The initial review of our prototypes pointed out a few interesting behaviors of this type of joint restraint, such as the geometrical constraints that restraint lines must satisfy in order to assume the desired shape and behavior. We believe there is a relation between the joint pattern and the joint radius versus joint length that needs to be determined and that will be the key for implementing such restraint lines. Other items of concern that will need to be addressed are pattern stability and range of motion performance.

Rather than provide three degrees of freedom at the shoulder and one at the elbow, mimicking the anatomical degrees of freedom, the MX-3 will provide two degrees of bending freedom at the shoulder, and two at the elbow and wrist, supplemented with a wrist roll DOF provided by a rotary bearing. As long as the suit occupant can rotate their upper arm axially within the pressure garment, this combination of joint locations can provide for the full range of mobility anatomically allowed. An equivalent set of joints will be implemented at the hip, knee, and ankle.

Given this initial architecture concept and the results of preliminary testing, an extensive investigation of pattern geometries and helicoidal restraint lines is currently being undertaken. The final goal of this investigation is to produce a single two-DOF mobility element sized for an standard elbow, evaluate its performance, and compare the torque and range of motion of each DOF with the comparable DOF on the Shuttle EMU arm. Pending the success of this assessment, a complete seven DOF arm segment and

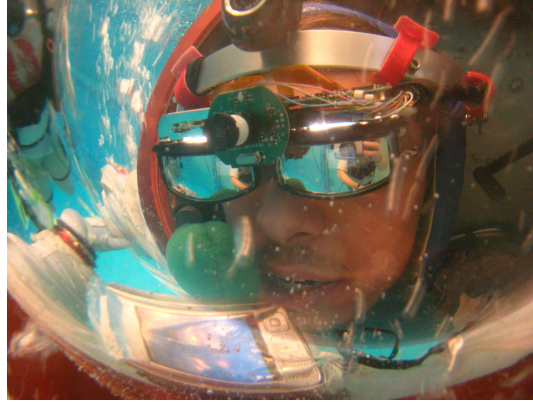


Figure B.6: MX-2 AR visor and Chest Mounted Display

ultimately a full set of soft-goods limbs will be constructed based on the concept.

The torso section of the MX-3 will make use of a modified version of the Morphing Upper Torso (MUT) Phase I concept[89, 18, 90], sized to the MX-3 design requirements to accommodate different sized helmet assemblies, not necessarily based on the ILC Dover I-Suit helmet for which the MUT was originally designed.

B.2.1 Integrated Advanced Controls and Displays

Advanced controls and displays have been a focus of ongoing research at the SSL for several years. Clearly, the availability of relevant information plays a critical role, even in seemingly non-technical, non-time-critical aspects of modern everyday life. Between smart phones and omni-present 3G networks, we have become accustomed to having the sum of human knowledge available at our fingertips whenever and wherever we need it. Such access to information will be especially critical in future human space exploration, where communication with Mission Control may be intolerably delayed, and EVA procedures will be much more ad hoc than in the Shuttle era.

Speech Recognition Advanced controls and displays research at the SSL has focused largely on the subject's means of interaction with the information display system. Unlike in typical experience on Earth, where keyboards, keypads, and mice are the primary tools for interaction with our information systems, EVA suits do not readily allow for quick tactile interaction without interfering with other EVA activities, and using fine control of the fingers in order to navigate an information display structure would likely be quite difficult and tiring in the suit. Therefore, voice recognition is being explored as potentially the best method of interaction with the information display system. The acoustic environment of the EVA suit presents an interesting challenge for this approach, although prior research has demonstrated that this challenge can be overcome [88]. MX-3 will make use of the knowledge gained in its predecessors, although a fresh start is envisioned in the software and hardware base. MX-3 will attempt to rely almost exclusively on open source software that is actively developed by the online community.

In-Helmet Displays The display system itself will consist of both visual and aural components, with the former providing more complex information, including maps and task instruction sets, and the latter reserved for quick, status-type information, especially in response to verbal queries. A translucent display will be integral to the extravehicular visor assembly (EVVA). The SSL has investigated a series of information displays, such as chest-mounted and wrist-mounted touch screens, as well as in-helmet translucent (see-through) helmet and head mounted displays.[88, 51]. In the current approach, the information display will be closely integrated into the helmet assembly, and a virtual image will be projected onto a partially-reflective pull-down visor.

The image is achieved through the use of a two-way mirror-coated visor and a

micro-projector. The micro-projector shines an image onto an opaque portion of the inner pressure helmet, and that image is then reflected on the two-way mirror-coated visor, enlarging it and causing it to appear at a virtual depth that is consistent with minimization of eye strain.

The intrinsic benefits of this system lie in the fact that the display can easily be cleared from view by the astronaut, either by giving a voice command that turns off the projector, or by simply pulling up the reflective visor. The reflective visor can also be used as a sun visor to reduce glare. A final decision on the MX-3 helmet shape has not been made, but two alternatives are currently being evaluated: the first uses the I-Suit-style elliptical bubble, while the second uses the more common hemispherical visor. The two geometries are being evaluated in terms of enabled visibility, projection system performance, available space for in-helmet systems, and ease of manufacturing.

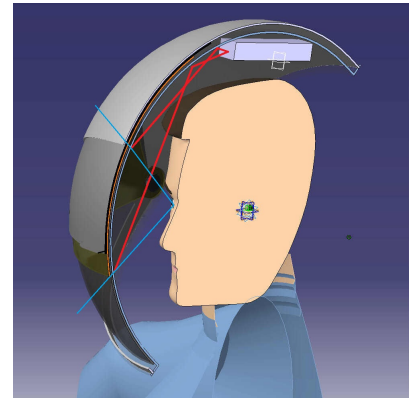


Figure B.7: In-Helmet AR Projection System Concept

B.2.2 Entry Type

In MX-B we have brought the concept further by replacing the accelerometer with a 9 DOF MEMS IMU (3-axis accelerometer, magnetometer and angular rate sensors). The IMU includes a microcontroller that filters the data from the three sensors via an extended Kalman filter, and then communicates over serial to the MX-Bravo's on-board computer. The filter and the dual redundant sensors allow us to

Although the suit entry method for the MX-3 has yet to be defined, a series of entry types, including rear entry, body seal closure, and neck entry[60], are currently under investigation. Some entry types, such as dual-plane body seal closures, were eliminated from consideration, as they are typically used only with hard-torso suits.

Testing will consist of the evaluation of each entry type on MX- α , an unpressurized mock-up of the MX-3, on which the various entry types can be quickly and easily implemented and tested for ease of use in both 1g and neutral buoyancy microgravity-simulation environments.

The type and complexity of support equipment necessary, donning and doffing envelopes, donning and doffing time, and subjective difficulty assessments will be considered when selecting the suit entry method to be employed on MX-3. Time permitting, the metabolic workload associated with suit donning and doffing will be examined, as previous research indicated that this represented a significant portion of the metabolic expenditure associated with EVA[60].



Figure B.8: NBRF Neck Entry
Concept Testing

B.3 Operational Environment and Scenarios

Operationally, the primary objective for MX-3 development is to create a pressure suit that is applicable to any and all forms of Earth-based EVA simulation, including

- 1-g field testing
- neutral buoyancy simulation of microgravity

- suspension harness simulation of partial gravity
- ballasted underwater simulation of partial gravity
- parabolic flight simulations

Of these, perhaps the most limiting is the noncounterweighted 1-g field trials, as the overall suit weight must be highly constrained to prevent unrealistic crew fatigue in Earth trials. By minimizing hard components, including bearings in the arms and legs, suit weight can be minimized, while a soft suit is more readily adjustable for critical dimensions of the wearer.

Given a functional lightweight pressure suit, external mounting garments can be added for neutral buoyancy ballast, or for the additional ballast required to simulate lunar or martian operations in the underwater environment. While the effects of viscous drag must be taken into account in simulation design, the underwater environment can provide a significantly more accurate simulation of partial gravity than would be available from counterweight suspension systems.

While realistic operational restrictions are best provided by an actual pressurized suit, some operations do not require the additional complexity and safety impacts of a pressurized suit. In line with the goal of providing appropriate levels of fidelity while maximizing simulation opportunities, a parallel effort is being placed on developing and fielding the first in a series



Figure B.9: Moonyard Operations and MX- α

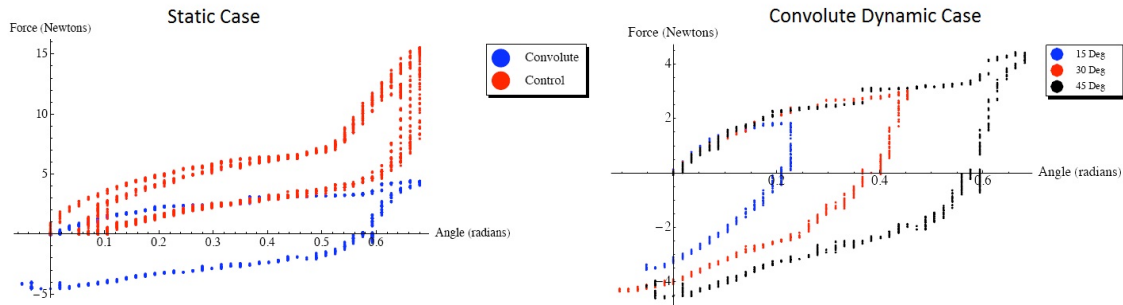


Figure B.10: Joint Torques Vs Angle Curves

of nonpressurized simulated suits, starting with the MX- α shown in Figure B.3. This series of suits, which represent the bulk and stiffness of a pressurized suit through fabric bulk, will be used for early development testing, operations which require human interfaces but not full fidelity suit function, and outreach activities wherein experimental subjects may not be qualified for operations under suit pressure.

B.4 Testing and Performance Evaluation

The success of an EVA depends directly on the ability of the astronaut to use their hands and arms. A spacesuit arm must therefore be produced which will allow the greatest possible range of motion with minimal induced torque. Three elbow joints were tested by measuring the force required to bend each joint through its entire range of motion. A test stand was produced which enabled a quantitative comparison of different elbow joint designs. This was accomplished by fabricating three sets of flanges, or circular interfaces, for each end of each of the three arm segments. This permitted the use of a single test stand to take data for all three segments.

The first joint tested was a baseline model made of a cylindrical tube of pressure

bladder material (urethane-backed nylon) wrapped in a cylinder of restraint layer material (nylon). The pressure bladder material was heat-sealed to make an airtight seal. The restraint layer was sewn together, and was designed to be slightly smaller than the pressure bladder, thereby taking the radial pressure loads. Axial pressure loads were taken by high-strength nylon restraint lines, which ran axially along the lines of non-extension.

The second design, the flat panel convolute joint, also consisted of an internal pressure bladder and an external restraint layer, but in this case each layer contained a series of three almond-shaped panels on the outer, elbow side, and two almond-shaped panels on the inner side of the joint, forming a traditional gore/convolute joint. In this joint design, the extra fabric on the extension side of the joint expands when the arm is bent and contracts when the arm is extended, allowing the joint to retain a constant volume throughout the range of motion, and minimizing the work required to move the joint.

The third joint tested was the toroidal joint, constructed from an alternating series of two sizes of concentric fabric rings. The larger rings were bonded to the inside of the joint, and attached to restraint lines through small clamps, while the smaller rings were bonded to the outside of the joint. Upon bending the elbow joint inwards, the toroidal joint folds up similarly to an accordion. The smaller rings travel toward the outer side of the joint, because the change in shape of the joint required by the bending motion forces an increase in fabric area on the outer part of the joint. This forces the larger rings to converge toward the inner side of the elbow, forming what, from a top view of the joint structure, resembles long isosceles trapezoid segments. This trapezoid shape provides a coincident center of pressure and center of restraint. The astronaut therefore has to overcome a lower induced joint torque and spring return force.

An angle sensor was used with a force gauge to calculate the dynamic force and angle

correspondence while each arm was slowly bent from its fully extended, resting position of 0° to its fully bent state of approximately 120° . Examining the static and dynamic moment curves allows a quantitative comparison of mobility effectiveness between joint types. As predicted, both the convolute and toroidal joints were much more effective designs than the plain tube. Future research will compare other types of joints, including the rolling convolute joint and the cardonic joint.

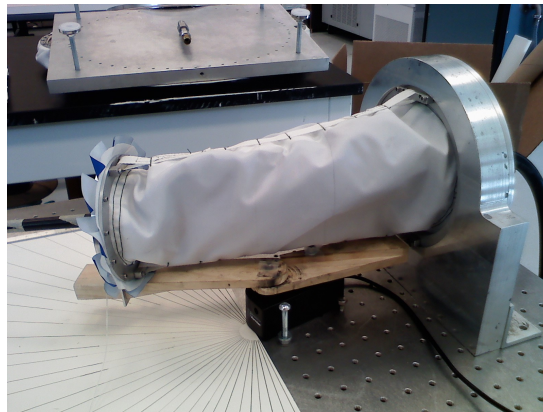


Figure B.11: Joint Torque Vs Angle Measurement Testbed

B.4.0.1 Destructive Hydrostatic Testing

During the Fall 2009 semester, a group of six freshman aerospace students at the University of Maryland built and tested nine straight arm segments (lacking joint articulation elements) of three different constructions, in order to determine the reliability of pressure suit manufacturing techniques, through finding the burst pressure, failure location and type of the burst of the three construction types. The construction types included a single-layer urethane-coated nylon single-wall laminate, a dual-layer urethane-coated nylon single wall laminate, and a traditional urethane-coated nylon pressure bladder with nylon restraint layer. The three sections were manufactured and preliminarily tested for

leaks using a partial-pressure glove box. In this preliminary testing phase, the sections underwent patching and verification in order to achieve a leak rate of less than a 0.05 psi/min in pressure drop. Once all the sections were leak checked, hydrostatic testing began. The goal of hydrostatic testing was to identify the failure modes and locations for each construction type, as well as assessing the repeatability of the manufacturing techniques used.



Figure B.12: Water Filled Pressurized Section: Leaks

Hydrostatic testing was conducted through the use of a custom-made testbed. A 1/2" hose was connected via piping from a water faucet to the arm section. The faucet provided water at 80 psi. The water flow and pressure was regulated by the faucet valve. This set-up was adopted to compensate for lost water pressure due to leaks in the arm. A pressure sensor was connected to the pipe and wired to an NI-9002 USB A/D acquisition device. Using Labview, the burst pressure and pressure profile in the arm over time were measured. Once each arm segment was filled with water, a screw plug on the extremity of the arm section wrist plate was sealed, and the arm was immersed in water to limit spills. Once the arm was in position, the faucet valve was slowly opened, increasing pressure



Figure B.13: Burst Test Section

in the arm segment, and data was acquired until a sudden pressure drop was detected. During the test, before the pressure drop, the test sections were taken out of the water periodically, in order to visually inspect for leaks and assess the status of the segment structure.

Results: The single-layer arms, on average, held a maximum of 7.96 psi before failure. The dual-layer arms held an average of 22.67 psi. However, this mean was affected by the low performance of test article DL-2, which was likely a result of a manufacturing defect rather than a reflection of the design. The standard pressure bladder/restraint layer arms held 27.67 psi on average.

All sections during the testing showed that leaks originate at the interfaces until the pressure loads originate a tear, usually on the axial sewing lines or heat seals (as expected) that then rapidly propagated to a catastrophic failure. In conclusion, the standard arm design was deemed the most effective and reliable design from a structural point of view. Also to note is the intrinsic safety factor that those manufacturing techniques deliver if

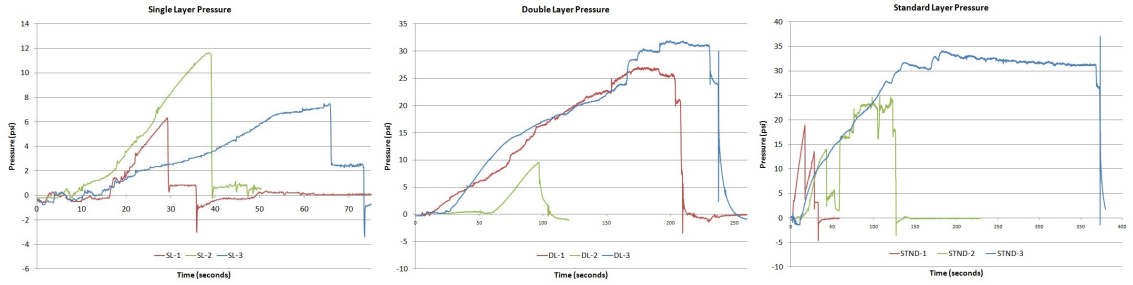


Figure B.14: Hydrostatic Test Results

we consider a nominal operating pressure of 3.5 psi.

B.5 Conclusions

The advantage of an independently-developed pressure suit is that it serves as a testbed across a wide variety of potential research fields. Thus far, this paper has focused on pressure suit technologies to be incorporated into the first production version of the MX-3. However, in the test bed mode, a number of alternative designs could be created for testing in a variety of application domains. For example, the SSL has a long-standing experience base in robotically-augmented suit components, ranging from the elbow to the hand itself. Related tests involved the incorporation of a full-scale robotic manipulation onto the suit backpack for use in assessing the benefits and limitations to robotic augmentation in a microgravity environment. By keeping the MX-3 modular and reconfigurable, it will become a workhorse in advanced EVA technology studies.

B.6 Acknowledgments

The authors wish to thank the students of the 2009 ENAE100 Human Factors team for their great work in building, testing and evaluating manufacturing techniques for space suit soft joints. Our most sincere thanks also go to the entire SSL personnel for giving us an incredible work and social environment where no idea is ever too crazy or "impossible", and where help is always at hand.

Appendix C

IMU Class source code

```
import visual, visual.controls, numpy
from math import sin, asin, cos, atan2, sqrt, pi

def init_rotation_matrix(yaw, pitch, roll):
    m = [[0.,0.,0.],[0.,0.,0.],[0.,0.,0.]]
    c1 = cos(roll)
    s1 = sin(roll)
    c2 = cos(pitch)
    s2 = sin(pitch)
    c3 = cos(yaw)
    s3 = sin(yaw)
    m[0][0] = c2 * c3
    m[0][1] = c3 * s1 * s2 - c1 * s3
    m[0][2] = s1 * s3 + c1 * c3 * s2
    m[1][0] = c2 * s3
    m[1][1] = c1 * c3 + s1 * s2 * s3
    m[1][2] = c1 * s2 * s3 - c3 * s1
    m[2][0] = -s2
    m[2][1] = c2 * s1
    m[2][2] = c1 * c2
    return m

def init_rotation_matrix3D(yaw, pitch, roll):
    m = [[0.,0.,0.],[0.,0.,0.],[0.,0.,0.]]
    c1 = cos(roll)
    s1 = sin(roll)
    c2 = cos(pitch)
    s2 = sin(pitch)
    c3 = cos(yaw)
    s3 = sin(yaw)
    m[0][0] = c2 * c3
    m[0][1] = c3 * s1 * s2 - c1 * s3
    m[0][2] = s1 * s3 + c1 * c3 * s2
    m[1][0] = -c2 * s3
    m[1][1] = -c1 * c3 - s1 * s2 * s3
    m[1][2] = -c1 * s2 * s3 + c3 * s1
    m[2][0] = s2
    m[2][1] = -c2 * s1
    m[2][2] = -c1 * c2
    return m

def DCM2Euler(DCM):
    pitch = -asin(DCM[2][0])
    roll = atan2(DCM[2][1],DCM[2][2])
    yaw = atan2(DCM[1][0],DCM[0][0])
    EulerAngles = [yaw, pitch, roll]
    return EulerAngles

class IMU(object):
    #-----
    def ResetCalibration(self,GRAVITY, ACCEL_X_MIN, ACCEL_X_MAX, ACCEL_Y_MIN,
        ACCEL_Y_MAX, ACCEL_Z_MIN, ACCEL_Z_MAX,MAGN_X_MIN,
        MAGN_X_MAX, MAGN_Y_MIN, MAGN_Y_MAX, MAGN_Z_MIN,
        MAGN_Z_MAX,GYRO_AVERAGE_OFFSET_X,
        GYRO_AVERAGE_OFFSET_Y, GYRO_AVERAGE_OFFSET_Z):
        self.GRAVITY = GRAVITY
        self.GainGyro = 0.06957*pi/180
        self.GainMag = 0.000917431192661
        self.ACCEL_X_MIN = ACCEL_X_MIN
        self.ACCEL_X_MAX = ACCEL_X_MAX
        self.ACCEL_Y_MIN = ACCEL_Y_MIN
        self.ACCEL_Y_MAX = ACCEL_Y_MAX
        self.ACCEL_Z_MIN = ACCEL_Z_MIN
        self.ACCEL_Z_MAX = ACCEL_Z_MAX
        self.MAGN_X_MIN = MAGN_X_MIN
        self.MAGN_X_MAX = MAGN_X_MAX
        self.MAGN_Y_MIN = MAGN_Y_MIN
        self.MAGN_Y_MAX = MAGN_Y_MAX
        self.MAGN_Z_MIN = MAGN_Z_MIN
        self.MAGN_Z_MAX = MAGN_Z_MAX
        self.GYRO_AVERAGE_OFFSET_X = GYRO_AVERAGE_OFFSET_X
        self.GYRO_AVERAGE_OFFSET_Y = GYRO_AVERAGE_OFFSET_Y
        self.GYRO_AVERAGE_OFFSET_Z = GYRO_AVERAGE_OFFSET_Z
        self.ACCEL_X_OFFSET = (self.ACCEL_X_MIN + self.ACCEL_X_MAX) / 2.0
        self.ACCEL_Y_OFFSET = (self.ACCEL_Y_MIN + self.ACCEL_Y_MAX) / 2.0
        self.ACCEL_Z_OFFSET = (self.ACCEL_Z_MIN + self.ACCEL_Z_MAX) / 2.0
        self.ACCEL_X_SCALE = ( self.GRAVITY / (self.ACCEL_X_MAX -
            self.ACCEL_X_OFFSET))
        self.ACCEL_Y_SCALE = ( self.GRAVITY / (self.ACCEL_Y_MAX -
```

```

        self.ACCEL_Y_OFFSET))
self.ACCEL_Z_SCALE = ( self.GRAVITY / (self.ACCEL_Z_MAX -
self.ACCEL_Z_OFFSET))
self.MAGN_X_OFFSET = (self.MAGN_X_MIN + self.MAGN_X_MAX) / 2.0
self.MAGN_Y_OFFSET = (self.MAGN_Y_MIN + self.MAGN_Y_MAX) / 2.0
self.MAGN_Z_OFFSET = (self.MAGN_Z_MIN + self.MAGN_Z_MAX) / 2.0
self.MAGN_X_SCALE = 100.0 / (self.MAGN_X_MAX - self.MAGN_X_OFFSET)
self.MAGN_Y_SCALE = 100.0 / (self.MAGN_Y_MAX - self.MAGN_Y_OFFSET)
self.MAGN_Z_SCALE = 100.0 / (self.MAGN_Z_MAX - self.MAGN_Z_OFFSET)
#-----
def __init__(self, ID, GRAVITY, ACCEL_X_MIN, ACCEL_X_MAX, ACCEL_Y_MIN,
ACCEL_Y_MAX, ACCEL_Z_MIN, ACCEL_Z_MAX, MAGN_X_MIN,
MAGN_X_MAX, MAGN_Y_MIN, MAGN_Y_MAX, MAGN_Z_MIN,
MAGN_Z_MAX, GYRO_AVERAGE_OFFSET_X, GYRO_AVERAGE_OFFSET_Y,
GYRO_AVERAGE_OFFSET_Z):
    self.x = ID
    self.M = init_rotation_matrix(0, pi/2, 0)
    if self.x == 0: self.Name = "Right Leg"
    if self.x == 1:
        self.Name = "Right Foot"
        self.M = numpy.eye(3)
    if self.x == 2: self.Name = "RightShank"
    if self.x == 3: self.Name = "Right Hand"
    if self.x == 4: self.Name = "Right Arm"
    if self.x == 5: self.Name = "Right Fore Arm"
    if self.x == 6: self.Name = "Hip"
    if self.x == 7: self.Name = "Neck"
    if self.x == 8: self.Name = "Spine"
    if self.x == 9: self.Name = "Left Hand"
    if self.x == 10: self.Name = "Left Arm"
    if self.x == 11: self.Name = "Left Fore Arm"
    if self.x == 12: self.Name = "Left Leg"
    if self.x == 13:
        self.Name = "Left Foot"
        self.M = numpy.eye(3)
    if self.x == 14: self.Name = "Left Shank"
    if self.x == 15:
        self.Name = "Right Shoulder"
        self.M = init_rotation_matrix(0, 0, pi/2)
    if self.x == 16:
        self.Name = "Left Shoulder"
        self.M = init_rotation_matrix(0, 0, pi/2)
    if self.x == 17:
        self.Name = "Head"
        self.M = numpy.eye(3)

self.RealIMU = False
self.mode3D = "none"
self.ResetCalibration(GRAVITY, ACCEL_X_MIN, ACCEL_X_MAX, ACCEL_Y_MIN,
ACCEL_Y_MAX, ACCEL_Z_MIN, ACCEL_Z_MAX, MAGN_X_MIN,
MAGN_X_MAX, MAGN_Y_MIN, MAGN_Y_MAX, MAGN_Z_MIN,
MAGN_Z_MAX, GYRO_AVERAGE_OFFSET_X,
GYRO_AVERAGE_OFFSET_Y, GYRO_AVERAGE_OFFSET_Z)

self.G_Dt = 0.02
self.DCM_Matrix = [[1., 0., 0.], [0., 1., 0.], [0., 0., 1.]]
self.OffsetMat = [[1., 0., 0.], [0., 1., 0.], [0., 0., 1.]]

self.accel = visual.vector(0,0,0)
self.gyro = visual.vector(0,0,0)
self.magnetom = visual.vector(0,0,0)

# Mahony Algorithm
self.Ki_ROLLPITCH = 0.00002
self.Ki_YAW = 0.00002
self.Kp_ROLLPITCH = 0.02
self.Kp_YAW = 1.2
self.roll = 0.
self.pitch = 0.
self.yaw = 0.
self.Euler = [self.roll, self.pitch, self.yaw]
self.Omega_I = visual.vector(0,0,0)
self.Omega_P = visual.vector(0,0,0)

#-----
def read_sensors(self, data):
    ax, ay, az, mx, my, mz, gx, gy, gz = data
    if self.RealIMU == False:
        gx = -gx
        gy = -gy
        gz = -gz
        my = -my
        mz = -mz
    Taccel = visual.vector((ax-self.ACCEL_X_OFFSET)*self.ACCEL_X_SCALE,
(ay-self.ACCEL_Y_OFFSET)*self.ACCEL_Y_SCALE,
(az-self.ACCEL_Z_OFFSET)*self.ACCEL_Z_SCALE)
    Tgyro = visual.vector(visual.dot((gx+self.GYRO_AVERAGE_OFFSET_X),
(gy+self.GYRO_AVERAGE_OFFSET_Y),
(gz+self.GYRO_AVERAGE_OFFSET_Z)), self.GainGyro))
    Tmagnetom = visual.vector((mx+self.MAGN_X_OFFSET)*self.MAGN_X_SCALE,

```

```

(my+self.MAGN_Y_OFFSET)*self.MAGN_Y_SCALE,
(mz+self.MAGN_Z_OFFSET)*self.MAGN_Z_SCALE)

self.accel = numpy.dot(self.M,Taccel)
self.gyro = numpy.dot(self.M,Tgyro)
self.magnetom = numpy.dot(self.M,Tmagnetom)

#-----
def Mahony(self,data):
    self.read_sensors(data)
    cos_roll = cos(self.roll)
    sin_roll = sin(self.roll)
    cos_pitch = cos(self.pitch)
    sin_pitch = sin(self.pitch)
    mag_x = self.magnetom[0]*cos_pitch +
            self.magnetom[1]*sin_roll*sin_pitch +
            self.magnetom[2]*cos_roll*sin_pitch
    mag_y = self.magnetom[1]*cos_roll - self.magnetom[2]*sin_roll
    self.MAG_Heading = atan2(-mag_y, mag_x)
    Omega = self.gyro + self.Omega_I
    Omega_Vector = Omega + self.Omega_P
    Update_Matrix = [[0,-self.G_Dt*Omega_Vector[2],
                    ,self.G_Dt*Omega_Vector[1]],
                    [self.G_Dt*Omega_Vector[2],0,-
                    self.G_Dt*Omega_Vector[0]],
                    [-self.G_Dt*Omega_Vector[1],
                    self.G_Dt*Omega_Vector[0],0]]
    Temporary_Matrix = visual.dot(self.DCM_Matrix,Update_Matrix)
    self.DCM_Matrix = self.DCM_Matrix + Temporary_Matrix
    X = [self.DCM_Matrix[0][0],self.DCM_Matrix[1][0],self.DCM_Matrix[2][0]]
    Y = [self.DCM_Matrix[0][1],self.DCM_Matrix[1][1],self.DCM_Matrix[2][1]]
    error = -visual.dot(X,Y)/2.
    Xo = X + visual.dot(Y,error)
    Yo = Y + visual.dot(X,error)
    Zo = visual.cross(Xo,Yo)
    m0 = visual.vector(visual.dot(visual.dot(3-visual.dot(Xo,Xo),Xo),.5))
    m1 = visual.vector(visual.dot(visual.dot(3-visual.dot(Yo,Yo),Yo),.5))
    m2 = visual.vector(visual.dot(visual.dot(3-visual.dot(Zo,Zo),Zo),.5))
    self.DCM_Matrix = numpy.transpose([m0,m1,m2])
    Accel_magnitude = visual.mag(self.accel) / self.GRAVITY
    Accel_weight = 1 - 2.*abs(1 - Accel_magnitude)
    if Accel_weight <= 0: Accel_weight = 0.
    elif Accel_weight >= 1: Accel_weight = 1.
    errorRollPitch = visual.cross(self.accel,self.DCM_Matrix[2])
    self.Omega_P = visual.vector(visual.dot(errorRollPitch,
            self.Kp_ROLLPITCH*Accel_weight))
    Scaled_Omega_I = visual.vector(visual.dot(errorRollPitch,
            self.Ki_ROLLPITCH*Accel_weight))
    self.Omega_I = self.Omega_I + Scaled_Omega_I
    mag_heading_x = cos(self.MAG_Heading)
    mag_heading_y = sin(self.MAG_Heading)
    errorCourse = (self.DCM_Matrix[0][0]*mag_heading_y) -
            (self.DCM_Matrix[1][0]*mag_heading_x)
    errorYaw = visual.vector(visual.dot(self.DCM_Matrix[2],errorCourse))
    Scaled_Omega_P = visual.vector(visual.dot(errorYaw,self.Kp_YAW))
    self.Omega_P = self.Omega_P + Scaled_Omega_P
    Scaled_Omega_I = visual.vector(visual.dot(errorYaw,self.Ki_YAW))
    self.Omega_I = self.Omega_I + Scaled_Omega_I

    self.pitch = -asin(self.DCM_Matrix[2][0])
    self.roll = atan2(self.DCM_Matrix[2][1],self.DCM_Matrix[2][2])
    self.yaw = atan2(self.DCM_Matrix[1][0],self.DCM_Matrix[0][0])
    self.Euler = visual.vector([self.roll,self.pitch,self.yaw])

    self.Accel_m_s = visual.dot(self.accel,9.81/self.GRAVITY)
    self.Mag_Gauss = visual.dot(self.magnetom, self.GainMag)

def Reset_Mahony(self,data):
    self.Omega_I = visual.vector(0,0,0)
    self.Omega_P = visual.vector(0,0,0)
    self.read_sensors(data)
    self.pitch = -atan2(self.accel[0], sqrt(self.accel[1]**2 +
            self.accel[2]**2))
    xAxis = [1.0, 0.0, 0.0]
    temp1 = visual.cross(self.accel, xAxis)
    temp2 = visual.cross(xAxis, temp1)
    self.roll = atan2(temp2[1], temp2[2])
    cos_roll = cos(self.roll)
    sin_roll = sin(self.roll)
    cos_pitch = cos(self.pitch)
    sin_pitch = sin(self.pitch)
    mag_x = self.magnetom[0]*cos_pitch +
            self.magnetom[1]*sin_roll*sin_pitch +
            self.magnetom[2]*cos_roll*sin_pitch
    mag_y = self.magnetom[1]*cos_roll - self.magnetom[2]*sin_roll
    self.MAG_Heading = atan2(-mag_y, mag_x)
    self.yaw = self.MAG_Heading
    self.DCM_Matrix = init_rotation_matrix(self.yaw, self.pitch, self.roll)
    self.Euler = [self.roll,self.pitch,self.yaw]
#-----Single IMU 3D Visual -----
#-----
def SingleSensor3DViewInit(self,mode):

```

```

self.mode = mode
if self.mode == "axis" or self.mode == "box":
    self.Display = visual.display(title=self.Name +
        " IMU(id=" + str(self.x) +
        ") 3D View",x=self.x*200-int(self.x/6)*1200,
        y=int(self.x/6)*200, width=200, height=200,
        center=(0,0,0), forward=(-.5,-.5,-.5),
        up = (0,0,1), background=(0,0,0),range=(2,2,2))
    self.Display.select()
    visual.arrow(color=visual.color.green,axis=(1,0,0),
        shaftwidth=0.02, fixedwidth=1)
    visual.arrow(color=visual.color.green,axis=(0,1,0),
        shaftwidth=0.02, fixedwidth=1)
    visual.arrow(color=visual.color.green,axis=(0,0,1),
        shaftwidth=0.02, fixedwidth=1)
    visual.label(pos=(0,0,0.8),text=self.Name + " IMU(id=" +
        str(self.x) + ")",box=0,opacity=0)
    visual.label(pos=(1,0,0),text="X",box=0,opacity=0)
    visual.label(pos=(0,1,0),text="Y",box=0,opacity=0)
    visual.label(pos=(0,0,1),text="Z",box=0,opacity=0)
if self.mode == "axis":
    self.Xaxis = visual.arrow(color=visual.color.red,axis=(1,0,0),
        shaftwidth=0.02, fixedwidth=1)
    self.Yaxis = visual.arrow(color=visual.color.blue,axis=(0,1,0),
        shaftwidth=0.02, fixedwidth=1)
    self.Zaxis = visual.arrow(color=visual.color.yellow,axis=(0,0,1),
        shaftwidth=0.02, fixedwidth=1)
    self.XaxisLab = visual.label(pos=(1,0,0),
        text="X IMU",box=0,opacity=0)
    self.YaxisLab = visual.label(pos=(0,1,0),
        text="Y IMU",box=0,opacity=0)
    self.ZaxisLab = visual.label(pos=(0,0,1),
        text="Z IMU",box=0,opacity=0)
elif self.mode == "box":
    self.IMU3DObject = visual.frame()
    visual.box(frame=self.IMU3DObject,length=1, height=0.05,
        width=.5, color=visual.color.red)
    visual.box(frame=self.IMU3DObject,length=1,height=0.08,
        width=0.1,color=visual.color.yellow)
    visual.arrow(frame=self.IMU3DObject,color=visual.color.yellow,
        axis=(.8,0,0), shaftwidth=0.06, fixedwidth=1)

def SingleSensor3DViewUpdate(self):
    M = numpy.dot(self.OffsetMat,numpy.transpose([self.DCM_Matrix[0],
        -self.DCM_Matrix[1],-self.DCM_Matrix[2]]))
    if self.mode == "axis":
        self.Xaxis.axis = M[0]
        self.Yaxis.axis = M[1]
        self.Zaxis.axis = M[2]
        self.XaxisLab.pos = self.Xaxis.axis
        self.YaxisLab.pos = self.Yaxis.axis
        self.ZaxisLab.pos = self.Zaxis.axis
    elif self.mode == "box":
        self.IMU3DObject.axis = M[0]
        self.IMU3DObject.up = M[2]
    elif self.mode == "none":
        pass
    if self.Display.kb.keys:
        k = self.Display.kb.getkey()
        if k == "c":
            self.OffsetMat = [self.DCM_Matrix[0],
                -self.DCM_Matrix[1],-self.DCM_Matrix[2]]
        if k == "x":
            self.OffsetMat = [[1,0,0],[0,1,0],[0,0,1]]

#####
#-----Body 3D Visual -----
#-----
#####
debug = True

HeadRadius = 0.15
SpineHLength = 0.235
SpineMLength = 0.25
SpineLLength = 0.3
HipLength = 0.2
ShoulderLength = 0.3
ArmLength = 0.25
ForeArmLength = 0.3
HandLength = 0.15
HandWidth = 0.1
HandHeight = 0.03
LegLength = 0.5
ShankLength = 0.5
FootLength = 0.25
FootWidth = 0.1
FootHeight = 0.03
Height = SpineHLength + SpineMLength + SpineLLength + LegLength +
ShankLength + FootHeight/2
DCM0=numpy.zeros([18,3,3])
for i in xrange(18):
    DCM0[i]=numpy.eye(3)

```

```

class grid():
    '''Ground grid of the whole scene'''
    def __init__(self, size=100, small_interval=1, big_interval=2):
        self.frame = visual.frame(pos=(0,0,0))
        for i in range(-size, size+small_interval, small_interval):
            if i % big_interval == 0:
                c = visual.color.gray(0.65)
            else:
                c = visual.color.gray(0.25)
            visual.curve(frame=self.frame, pos=[(size,i,0),
                (-size,i,0)], color=c)
            visual.curve(frame=self.frame, pos=[(i,size,0),
                (i,-size,0)], color=c)

def SliderPos(Slider):
    Slider.value

def StartStopAction(b): # Called by controls when button clicked
    if b.text == 'Start':
        b.text = 'Stop'
    else:
        b.text = 'Start'

def Body3DInit():
    global StickMan, Slider, StartStop,
        debug, f, Head, SpineH, SpineM, JSM, SpineL, JSL,
        ShoulderL, JSHL, ShoulderR, JSHR, ArML, JAL, ForeArML, JFAL, HandL, ArmR, JAR,
        ForeArmR, JFAR, HandR, HipL, JHL, HipR, JHR, LegR, JLR, ShankR, JShkR,
        FootR, LegL, JLL, ShankL, JShkL, FootL
    HeadPosition = -(SpineHLength + SpineMLength + SpineLLength +
        LegLength + ShankLength + FootHeight/2)
    SegmentsR = 0.015
    JSphereR = 0.02
    ObjectColorR = (0,1,0)
    ObjectColorL = (1,0,0)
    JSphereColor = (0,0,1)
    ControlsWindow = visual.controls.controls(title='Command Window',
        x=1200, y=600, width=400, height=200, range=100)
    Slider = visual.controls.slider(pos=(-90,-25), width=7,
        length=180, action=lambda: SliderPos(Slider))
    StartStop = visual.controls.button(text='Start', pos=(-70,10),
        width=40, height=40, action=lambda: StartStopAction(StartStop))

    StickMan = visual.display(title="3D Body View", x=1200, y=0,
        width=400, height=600, center=(0,0,0),
        forward=(-.5, .5, .5), up = (0,0,-1),
        background=(0,0,0), range=(3,3,3))
    StickMan.select()
    StickMan.lights = [visual.distant_light(direction=(0.22, 0.44,
        -0.88), color=visual.color.gray(0.8)),
        visual.distant_light(direction=(-0.88, -0.22, 0.44),
        color=visual.color.gray(0.3))]
    grid(50)
    visual.arrow(pos=(0,0,0), axis=(.4,0,0), shaftwidth=0.015, color=(1,0,0))
    visual.label(pos=(.5,0,0), text='X', box=False, opacity=0)
    visual.arrow(pos=(0,0,0), axis=(0,-.4,0), shaftwidth=0.015, color=(0,1,0))
    visual.label(pos=(0,-.5,0), text='Y', box=False, opacity=0)
    visual.arrow(pos=(0,0,0), axis=(0,0,-.4), shaftwidth=0.015, color=(0,0,1))
    visual.label(pos=(0,0,-.5), text='Z', box=False, opacity=0)
    f = visual.frame()
    Head = visual.ellipsoid(frame=f, pos=(0,0,HeadPosition),
        axis=(0,0,2*HeadRadius), length=2*HeadRadius,
        height=HeadRadius*1.5, width=HeadRadius*1.5,
        color=ObjectColorR)
    SpineH = visual.cylinder(frame=f, pos=Head.pos, axis=(0,0,SpineHLength),
        radius=SegmentsR, color=ObjectColorR)
    SpineM = visual.cylinder(frame=f, pos=SpineH.pos+SpineH.axis,
        axis=(0,0,SpineMLength), radius=SegmentsR,
        color=ObjectColorR)
    JSM=visual.sphere(frame=f, pos=SpineM.pos, radius=JSphereR,
        color=JSphereColor)
    SpineL = visual.cylinder(frame=f, pos=SpineM.pos+SpineM.axis,
        axis=(0,0,SpineLLength), radius=SegmentsR,
        color=ObjectColorR)
    JSL=visual.sphere(frame=f, pos=SpineL.pos,
        radius=JSphereR, color=JSphereColor)
    ShoulderL = visual.cylinder(frame=f, pos=SpineM.pos,
        axis=(0,-ShoulderLength,0),
        radius=SegmentsR, color=ObjectColorL)
    JSHL=visual.sphere(frame=f, pos=ShoulderL.pos+ShoulderL.axis,
        radius=JSphereR, color=JSphereColor)
    ShoulderR = visual.cylinder(frame=f, pos=SpineM.pos,
        axis=(0,ShoulderLength,0), radius=SegmentsR,
        color=ObjectColorR)
    JSHR=visual.sphere(frame=f, pos=ShoulderR.pos+ShoulderR.axis,
        radius=JSphereR, color=JSphereColor)
    ArML = visual.cylinder(frame=f, pos=ShoulderL.pos+ShoulderL.axis,
        axis=(0,0,ArMLength), radius=SegmentsR,
        color=ObjectColorL)
    JAL=visual.sphere(frame=f, pos=visual.vector(ArML.pos+ArML.axis),
        radius=JSphereR, color=JSphereColor)
    ForeArML = visual.cylinder(frame=f, pos=visual.vector(ArML.pos+ArML.axis),
        axis=(0,0,ForeArMLength), radius=SegmentsR,

```

```

    color=ObjectColorL)
JFAL=visual.sphere(frame=f,pos=visual.vector(ForeArmL.pos+ForeArmL.axis),
    radius=JSphereR, color=JSphereColor)
HandL = visual.box(frame=f,pos=visual.vector(ForeArmL.pos+ForeArmL.axis),
    axis=(0,0,HandLength), height=HandHeight,
    width=HandWidth, color=ObjectColorL)
HandL.pos = HandL.pos+visual.norm(HandL.axis)*HandLength/2
ArmR = visual.cylinder(frame=f,pos=ShoulderR.pos+ShoulderR.axis,
    axis=(0,0,ArmLength), radius=SegmentsR,
    color=ObjectColorR)
JAR=visual.sphere(frame=f,pos=visual.vector(ArmR.pos+ArmR.axis),
    radius=JSphereR, color=JSphereColor)
ForeArmR = visual.cylinder(frame=f,pos=visual.vector(ArmR.pos+ArmR.axis),
    axis=(0,0,ForeArmLength), radius=SegmentsR,
    color=ObjectColorR)
JFAR=visual.sphere(frame=f,pos=visual.vector(ForeArmR.pos+ForeArmR.axis),
    radius=JSphereR, color=JSphereColor)
HandR = visual.box(frame=f,pos=visual.vector(ForeArmR.pos+ForeArmR.axis),
    axis=(0,0,HandLength), height=HandHeight,
    width=HandWidth, color=ObjectColorR)
HandR.pos = HandR.pos+visual.norm(HandR.axis)*HandLength/2
HipL = visual.cylinder(frame=f,pos=SpineL.pos+SpineL.axis,
    axis=(0,-HipLength,0), radius=SegmentsR,
    color=ObjectColorL)
JHL=visual.sphere(frame=f,pos=HipL.pos+HipL.axis, radius=JSphereR,
    color=JSphereColor)
HipR = visual.cylinder(frame=f,pos=SpineL.pos+SpineL.axis, axis=-HipL.axis,
    radius=SegmentsR, color=ObjectColorR)
JHR=visual.sphere(frame=f,pos=HipR.pos+HipR.axis, radius=JSphereR,
    color=JSphereColor)
LegR = visual.cylinder(frame=f,pos=HipR.pos+HipR.axis,
    axis=(0,0,LegLength), radius=SegmentsR,
    color=ObjectColorR)
JLR=visual.sphere(frame=f,pos=visual.vector(LegR.pos+LegR.axis),
    radius=JSphereR, color=JSphereColor)
ShankR = visual.cylinder(frame=f,pos=visual.vector(LegR.pos+LegR.axis),
    axis=(0,0,ShankLength), radius=SegmentsR,
    color=ObjectColorR)
JShkR=visual.sphere(frame=f,pos=visual.vector(ShankR.pos+ShankR.axis),
    radius=JSphereR, color=JSphereColor)
FootR = visual.box(frame=f,pos=visual.vector(ShankR.pos+ShankR.axis),
    axis=(FootLength,0,0), up =(0,0,1), height=FootHeight,
    width=FootWidth, color=ObjectColorR)
FootR.pos = FootR.pos+visual.norm(FootR.axis)*FootLength/3
LegL = visual.cylinder(frame=f,pos=HipL.pos+HipL.axis,
    axis=(0,0,LegLength), radius=SegmentsR,
    color=ObjectColorL)
JLL=visual.sphere(frame=f,pos=visual.vector(LegL.pos+LegL.axis),
    radius=JSphereR, color=JSphereColor)
ShankL = visual.cylinder(frame=f,pos=visual.vector(LegL.pos+LegL.axis),
    axis=(0,0,ShankLength), radius=SegmentsR,
    color=ObjectColorL)
JShkL=visual.sphere(frame=f,pos=visual.vector(ShankL.pos+ShankL.axis),
    radius=JSphereR, color=JSphereColor)
FootL = visual.box(frame=f,pos=visual.vector(ShankL.pos+ShankL.axis),
    axis=(FootLength,0,0), up =(0,0,1), height=FootHeight,
    width=FootWidth, color=ObjectColorL)
FootL.pos = FootL.pos+visual.norm(FootL.axis)*FootLength/3
for x in xrange(18):
    globals()["Axis" + str(x)+"X"] = visual.arrow(frame=f,
        pos=(0,0,0), axis=(.1,0,0),
        shaftwidth=.01, color = (1,0,0))
    globals()["Axis" + str(x)+"Y"] = visual.arrow(frame=f,
        pos=(0,0,0), axis=(0,.1,0),
        shaftwidth=.01, color = (0,1,0))
    globals()["Axis" + str(x)+"Z"] = visual.arrow(frame=f,
        pos=(0,0,0), axis=(0,0,.1),
        shaftwidth=.01, color = (0,0,1))
    globals()["Axis" + str(x)+"X"].visible = debug
    globals()["Axis" + str(x)+"Y"].visible = debug
    globals()["Axis" + str(x)+"Z"].visible = debug

def Body3DUpdate(DCMinput):
    global StickMan,f,DCMO,debug, Head,SpineH,SpineM,JSM,SpineL,JSL,
    ShoulderL,JShL,ShoulderR,JShR,ArmL,JAL,ForeArmL,
    JFAL,HandL,ArmR,JAR,ForeArmR,JFAR,HandR,HipL,JHL,
    HipR,JHR,LegR,JLR,ShankR,JShkR, FootR,LegL,JLL,
    ShankL,JShkL,FootL
    DCM=numpy.zeros([18,3,3])
    for i in xrange(18):
        #DCM[i] = numpy.dot(DCMO[i],numpy.transpose([DCMinput[i][0],
        -DCMinput[i][1],-DCMinput[i][2]]))
        DCM[i] = numpy.dot(DCMO[i],numpy.transpose(DCMinput[i]))
    BD = 17 # Head
    M = DCM[BD]
    Head.axis = visual.norm(M[2]) * 2 *HeadRadius
    Head.pos=(0,0,0)
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.2)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.2)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.2)
        globals()["Axis" + str(BD)+"X"].pos = Head.pos

```

```

        globals()["Axis" + str(BD)+"Y"].pos = Head.pos
        globals()["Axis" + str(BD)+"Z"].pos = Head.pos
    BD = 7 # SpineH
    M = DCM[BD]
    SpineH.axis = -visual.norm(M[2]) * SpineHLength
    SpineH.pos = Head.pos
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = SpineH.pos + SpineH.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = SpineH.pos + SpineH.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = SpineH.pos + SpineH.axis/2
    BD = 8 # SpineM
    M = DCM[BD]
    SpineM.axis = -visual.norm(M[2]) * SpineMLength
    SpineM.pos = SpineH.pos + SpineH.axis
    JSM.pos = SpineM.pos
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = SpineM.pos + SpineM.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = SpineM.pos + SpineM.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = SpineM.pos + SpineM.axis/2
    BD = 6 # SpineL + Hips
    M = DCM[BD]
    SpineL.axis = -visual.norm(M[2]) * SpineLLength
    SpineL.pos = SpineM.pos + SpineM.axis
    JSL.pos = SpineL.pos
    HipL.axis = visual.norm(M[1]) * HipLength
    HipL.pos = SpineL.pos + SpineL.axis
    JHL.pos = HipL.pos + HipL.axis
    HipR.axis = -HipL.axis
    HipR.pos = SpineL.pos + SpineL.axis
    JHR.pos = HipR.pos + HipR.axis
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = SpineL.pos + SpineL.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = SpineL.pos + SpineL.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = SpineL.pos + SpineL.axis/2
    BD = 16 # ShoulderL
    M = DCM[BD]
    ShoulderL.axis = visual.norm(M[1]) * ShoulderLength
    ShoulderL.pos = SpineM.pos
    JShL.pos = ShoulderL.pos + ShoulderL.axis
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = SpineM.pos + ShoulderL.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = SpineM.pos + ShoulderL.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = SpineM.pos + ShoulderL.axis/2
    BD = 15 # ShoulderR
    M = DCM[BD]
    ShoulderR.axis = -visual.norm(M[1]) * ShoulderLength
    ShoulderR.pos = SpineM.pos
    JShR.pos = ShoulderR.pos + ShoulderR.axis
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = SpineM.pos + ShoulderR.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = SpineM.pos + ShoulderR.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = SpineM.pos + ShoulderR.axis/2
    BD = 10 # ArmL
    M = DCM[BD]
    ArmL.axis = -visual.norm(M[2]) * ArmLength
    ArmL.pos = ShoulderL.pos + ShoulderL.axis
    JAL.pos = visual.vector(ArmL.pos + ArmL.axis)
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = ArmL.pos + ArmL.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = ArmL.pos + ArmL.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = ArmL.pos + ArmL.axis/2
    BD = 11 # ForeArmL
    M = DCM[BD]
    ForeArmL.axis = -visual.norm(M[2]) * ForeArmLength
    ForeArmL.pos = visual.vector(ArmL.pos + ArmL.axis)
    JFAL.pos = visual.vector(ForeArmL.pos + ForeArmL.axis)
    if debug:
        globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
        globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
        globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
        globals()["Axis" + str(BD)+"X"].pos = ForeArmL.pos + ForeArmL.axis/2
        globals()["Axis" + str(BD)+"Y"].pos = ForeArmL.pos + ForeArmL.axis/2
        globals()["Axis" + str(BD)+"Z"].pos = ForeArmL.pos + ForeArmL.axis/2
    BD = 9 # HandL

```

```

M = DCM[BD]
HandL.axis = -visual.norm(M[2]) * HandLength
HandL.up = M[1]
HandL.pos = visual.vector(ForeArmL.pos + ForeArmL.axis)
HandL.pos = HandL.pos + visual.norm(HandL.axis) * HandLength/2
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = HandL.pos
    globals()["Axis" + str(BD)+"Y"].pos = HandL.pos
    globals()["Axis" + str(BD)+"Z"].pos = HandL.pos
BD = 4 # ArmR
M = DCM[BD]
ArmR.axis = -visual.norm(M[2]) * ArmLength
ArmR.pos = ShoulderR.pos + ShoulderR.axis
JAR.pos = visual.vector(ArmR.pos + ArmR.axis)
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = ArmR.pos + ArmR.axis/2
    globals()["Axis" + str(BD)+"Y"].pos = ArmR.pos + ArmR.axis/2
    globals()["Axis" + str(BD)+"Z"].pos = ArmR.pos + ArmR.axis/2
BD = 5 # ForeArmR
M = DCM[BD]
ForeArmR.axis = -visual.norm(M[2]) * ForeArmLength
ForeArmR.pos = visual.vector(ArmR.pos + ArmR.axis)
JFAR.pos = visual.vector(ForeArmR.pos + ForeArmR.axis)
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = ForeArmR.pos + ForeArmR.axis/2
    globals()["Axis" + str(BD)+"Y"].pos = ForeArmR.pos + ForeArmR.axis/2
    globals()["Axis" + str(BD)+"Z"].pos = ForeArmR.pos + ForeArmR.axis/2
BD = 3 # HandR
M = DCM[BD]
HandR.axis = -visual.norm(M[2]) * HandLength
HandR.up = M[1]
HandR.pos = visual.vector(ForeArmR.pos + ForeArmR.axis)
HandR.pos = HandR.pos + visual.norm(HandR.axis) * HandLength/2
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = HandR.pos
    globals()["Axis" + str(BD)+"Y"].pos = HandR.pos
    globals()["Axis" + str(BD)+"Z"].pos = HandR.pos
BD = 12 # LegL
M = DCM[BD]
LegL.axis = -visual.norm(M[2]) * LegLength
LegL.pos = HipL.pos + HipL.axis
JLL.pos = visual.vector(LegL.pos + LegL.axis)
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = LegL.pos + LegL.axis/2
    globals()["Axis" + str(BD)+"Y"].pos = LegL.pos + LegL.axis/2
    globals()["Axis" + str(BD)+"Z"].pos = LegL.pos + LegL.axis/2
BD = 14 # ShankL
M = DCM[BD]
ShankL.axis = -visual.norm(M[2]) * ShankLength
ShankL.pos = visual.vector(LegL.pos + LegL.axis)
JShkL.pos = visual.vector(ShankL.pos + ShankL.axis)
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = ShankL.pos + ShankL.axis/2
    globals()["Axis" + str(BD)+"Y"].pos = ShankL.pos + ShankL.axis/2
    globals()["Axis" + str(BD)+"Z"].pos = ShankL.pos + ShankL.axis/2
BD = 13 # FootL
M = DCM[BD]
FootL.axis = visual.norm(M[0]) * FootLength
FootL.up = M[2]
FootL.pos = visual.vector(ShankL.pos + ShankL.axis)
FootL.pos = FootL.pos+visual.norm(FootL.axis) * FootLength/3
if debug:
    globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
    globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
    globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
    globals()["Axis" + str(BD)+"X"].pos = FootL.pos
    globals()["Axis" + str(BD)+"Y"].pos = FootL.pos
    globals()["Axis" + str(BD)+"Z"].pos = FootL.pos
BD = 0 # LegR
M = DCM[BD]
LegR.axis = -visual.norm(M[2]) * LegLength
LegR.pos = HipR.pos + HipR.axis
JLR.pos = visual.vector(LegR.pos + LegR.axis)
if debug:

```



```

globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
globals()["Axis" + str(BD)+"X"].pos = LegR.pos + LegR.axis/2
globals()["Axis" + str(BD)+"Y"].pos = LegR.pos + LegR.axis/2
globals()["Axis" + str(BD)+"Z"].pos = LegR.pos + LegR.axis/2
BD = 2 # ShankR
M = DCM[BD]
ShankR.axis = -visual.norm(M[2]) * ShankLength
ShankR.pos = visual.vector(LegR.pos + LegR.axis)
JShkR.pos = visual.vector(ShankR.pos + ShankR.axis)
if debug:
globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
globals()["Axis" + str(BD)+"X"].pos = ShankR.pos + ShankR.axis/2
globals()["Axis" + str(BD)+"Y"].pos = ShankR.pos + ShankR.axis/2
globals()["Axis" + str(BD)+"Z"].pos = ShankR.pos + ShankR.axis/2
BD = 1 # FootR
M = DCM[BD]
FootR.axis = visual.norm(M[0]) * FootLength
FootR.up = M[2]
FootR.pos = visual.vector(ShankR.pos + ShankR.axis)
FootR.pos = FootR.pos + visual.norm(FootR.axis) * FootLength/3
if debug:
globals()["Axis" + str(BD)+"X"].axis = numpy.multiply(M[0],.1)
globals()["Axis" + str(BD)+"Y"].axis = numpy.multiply(M[1],.1)
globals()["Axis" + str(BD)+"Z"].axis = numpy.multiply(M[2],.1)
globals()["Axis" + str(BD)+"X"].pos = FootR.pos
globals()["Axis" + str(BD)+"Y"].pos = FootR.pos
globals()["Axis" + str(BD)+"Z"].pos = FootR.pos
if -asin(DCM[13][2][0])<0:
zL = FootL.pos+visual.norm(FootL.axis) * FootLength/2
else:
zL = FootL.pos-visual.norm(FootL.axis) * FootLength/2
if -asin(DCM[1][2][0])<0:
zR = FootR.pos+visual.norm(FootR.axis) * FootLength/2
else:
zR = FootR.pos-visual.norm(FootR.axis) * FootLength/2
if abs(zL[2])>abs(zR[2]):
f.pos=(-HipL.x,-HipL.y,-zL[2]+FootHeight/2)
else:
f.pos=(-HipR.x,-HipR.y,-zR[2]+FootHeight/2)
if StickMan.kb.keys:
k = StickMan.kb.getkey()
if k == "d":
if debug:
debug = False
for x in xrange(18):
globals()["Axis" + str(x)+"X"].visible = False
globals()["Axis" + str(x)+"Y"].visible = False
globals()["Axis" + str(x)+"Z"].visible = False
else:
debug = True
for x in xrange(18):
globals()["Axis" + str(x)+"X"].visible = True
globals()["Axis" + str(x)+"Y"].visible = True
vars()["Axis" + str(x)+"Z"].visible = True
elif k == "p":
StickMan.center=(0,0,LegLength + ShankLength + FootHeight/2)
elif k == "r":
StickMan.forward = (-1,-1,0.5)
elif k == "q":
StickMan.center = (0,0,LegLength + ShankLength + FootHeight/2)
StickMan.forward = (-1,-1,0.5)
elif k == "up":
StickMan.center -= visual.norm((StickMan.forward[0],
StickMan.forward[1],0))*0.05
elif k == "down":
StickMan.center += visual.norm((StickMan.forward[0],
StickMan.forward[1],0))*0.05
elif k == "left":
StickMan.center += (0,0,0.05)
elif k == "right":
StickMan.center -= (0,0,0.05)
elif k == "x":
StickMan.forward = (-1,0,0)
elif k == "y":
StickMan.forward = (0,-1,0)
elif k == "z":
StickMan.forward = (0,0,1)
elif k == "=":
if StickMan.fov >= pi-pi/180.:
StickMan.fov = pi
else:
StickMan.fov += pi/180.
elif k == "-":
if StickMan.fov <= pi/180.:
StickMan.fov = pi/180.
else:
StickMan.fov -=pi/180.
elif k == "0":

```

```

        StickMan.fov = pi/3.
    elif k == "c":
        CalID = 6 # Head = 17, Hip = 6
        Euler0 = DCM2Euler(DCMinput[CalID])
        DCMO[CalID] = init_rotation_matrix3D(0,Euler0[1],Euler0[2])
        for i in xrange(18):
            if i != CalID:
                Euler = DCM2Euler(DCMinput[i])
                DCMO[i] = init_rotation_matrix3D(
                    Euler[0]-Euler0[0],Euler[1],Euler[2])
    elif k == "v":
        for i in xrange(18):
            DCMO[i]=numpy.eye(3)

#-----
#=====
#-----
#----- Sample Functions to use the class-----
#-----
#=====
def HardwareIMU(view,serport):
    import serial, string, time
    IMUDevice = serial.Serial(serport, baudrate=1000000)
    IMUDevice.readline()
    data = IMUDevice.readline()
    line = string.split(data,",")
    Acc = [float(line[0]), float(line[1]), float(line[2])]
    Mag = [float(line[3]), float(line[4]), float(line[5])]
    Gyro = [float(line[6]), float(line[7]), float(line[8])]
    Euler = [float(line[9]), float(line[10]), float(line[11])]
    data = Acc + Mag + Gyro
    IMU1 = IMU(0, 256.,-250.,250.,-250.,250.,-250.,250.,
        -600.,600.,-600.,600.,-600.,600.,0.,0.,0.)
    IMU1.RealIMU = True
    IMU1.SingleSensor3DViewInit(view)
    IMU1.Reset_Mahony(data)

    while 1:
        data = IMUDevice.readline()
        line = string.split(data,",")
        Acc = [float(line[0]), float(line[1]), float(line[2])]
        Mag = [float(line[3]), float(line[4]), float(line[5])]
        Gyro = [float(line[6]), float(line[7]), float(line[8])]
        Euler = visual.vector([float(line[9]), float(line[10]),
            float(line[11])])
        data = Acc + Mag + Gyro
        IMU1.Mahony(data)
        IMU1.SingleSensor3DViewUpdate()

#-----
def FromFile(SaveFiltered, ViewSingleIMU, ViewBodyModel):
    import PyQt4.QtGui
    import PyQt4.Qt, PyQt4.Qt5
    import string, sys, pickle, struct, time, numpy,thread
    global Slider, StartStop
    timestart = 0
    counter = 0
    app = PyQt4.Qt.QApplication(sys.argv)
    try:
        CalibFileName = "Calib.cfg"
        CalibFile = open(CalibFileName, 'r')
        data = CalibFile.read()
        data = string.split(data, ";")
        for x in xrange(18):
            CalibData = string.split(data[x], ",")
            name = "IMU" + str(x)
            vars()[name]=IMU(x,float(CalibData[0]), float(CalibData[1]),
                float(CalibData[2]), float(CalibData[3]),
                float(CalibData[4]), float(CalibData[5]),
                float(CalibData[6]), float(CalibData[7]),
                float(CalibData[8]),float(CalibData[9]),
                float(CalibData[10]),float(CalibData[11]),
                float(CalibData[12]),float(CalibData[13]),
                float(CalibData[14]),float(CalibData[15]))
            if ViewSingleIMU:
                vars()[name].SingleSensor3DViewInit("box")
        CalibFile.close()
    except:
        CalibFileName = PyQt4.Qt.QFileDialog.getOpenFileName(None,
            "Open BPMS Calibration File", "*.cfg")
        CalibFile = open(CalibFileName, 'r')
        data = CalibFile.read()
        data = string.split(data, ";")
        for x in xrange(18):
            CalibData = string.split(data[x], ",")
            name = "IMU" + str(x)
            vars()[name]=IMU(x,float(CalibData[0]), float(CalibData[1]),
                float(CalibData[2]), float(CalibData[3]),
                float(CalibData[4]), float(CalibData[5]),
                float(CalibData[6]), float(CalibData[7]),
                float(CalibData[8]), float(CalibData[9]),
                float(CalibData[10]), float(CalibData[11]),
                float(CalibData[12]), float(CalibData[13]),

```

```

        float(CalibData[14]), float(CalibData[15]))
    if ViewSingleIMU:
        vars()[name].SingleSensor3DViewInit("box")
    CalibFile.close()
if ViewBodyModel:
    Body3DInit()

FileName = PyQt4.Qt.QFileDialog.getOpenFileName(None,
"Open BPMS Raw Data File", "*.txt")
File = open(FileName, 'r')
print FileName
for i, l in enumerate(File):
    pass
EndOfFile = File.tell()
File.seek(0)
if SaveFiltered:
    FilteredFileName = PyQt4.Qt.QFileDialog.getSaveFileName(None,
"Save Filtered Data as...", ".txt")
    Filtered = open(FilteredFileName, 'w')
print "Filtering..."
i=0
Time = time.time()
eulerRad = numpy.zeros([18,3])
DCM = numpy.zeros([18,3,3])
while 1:
    while File.tell() < EndOfFile:
        if ViewBodyModel:
            while StartStop.text == 'Start':
                File.seek(Slider.value*float(EndOfFile)/100.)

            try:
                timestart = time.time()
                [timeFileOld,line] = pickle.load(File)
                if line[0:2]=="$#" and line[338:]=="#$$":
                    line2Decode = line[2:56] + line[58:112] + line[114:168]+
                    line[170:224]+ line[226:280] +
                    line[282:336]
                    for x in xrange(0,18):
                        name = "IMU" + str(x)
                        if i == 0:
                            vars()[name].Reset_Mahony(struct.unpack(
">hhhhhhhhh", line2Decode[18*x:18*(x+1)]))
                        else:
                            vars()[name].Mahony(struct.unpack(">hhhhhhhhh",
line2Decode[18*x:18*(x+1)]))
                        if ViewSingleIMU:
                            vars()[name].SingleSensor3DViewUpdate()
                            eulerRad[x] = vars()[name].Euler
                            DCM[x] = vars()[name].DCM_Matrix
                    if i == 0: i=1
                if SaveFiltered:
                    data = str(Time) + ","
                    for x in xrange(0,18):
                        data+= str(eulerRad[x][0]) +","+str(eulerRad[x][1])
                        +","+str(eulerRad[x][2])+","
                    data += "\n"
                    Filtered.write(data)
                if ViewBodyModel:
                    thread.start_new_thread(Body3DUpdate,(DCM,))
                Time += 0.02
                if (time.time()-timestart) < 0.02:
                    if ViewBodyModel:
                        time.sleep(0.02-(time.time()-timestart))
            except:
                print "File Error"
        if ViewBodyModel:
            Slider.setvalue(float(File.tell())*100./float(EndOfFile))
        else:
            if int(File.tell()*100./EndOfFile) != counter:
                print int(File.tell()*100./EndOfFile)
                counter = int(File.tell()*100./EndOfFile)
    if ViewBodyModel:
        StartStop.text = 'Start'
    i=0
    if ViewBodyModel:
        Slider.setvalue(0)
        File.seek(0)
    else:
        break
    Filtered.close()
    File.close()
    print "done"

#=====
#=====
#=====
FromFile(SaveFiltered = True, ViewSingleIMU = False, ViewBodyModel = False)

```

Appendix D

Mathematical Background

The following section will briefly introduce the reader to the fundamental mathematical concepts used in this work. While standard symbolism will be used as much as possible, this section will serve as the baseline for the nomenclature and symbolism used. Most of the notation and the definitions have been summarized from [81, 82, 80, 83] Scalar values will be expressed with a lower case Latin letter (i.e. a, b, c, \dots), while vectors will be represented as one-dimensional column arrays:

$$\vec{X} = [a, b, c]^T \quad (\text{D.1})$$

Unit vectors are vectors of norm 1 and will be represented with the following symbolism:

$$\hat{Y} = [a, b, c]^T; \quad \hat{x} = \frac{\vec{X}}{|\vec{X}|}; \quad |\hat{Y}| = 1 \quad (\text{D.2})$$

and the following notation will be used for the following basic vector operations:

$$\vec{A}\vec{B} = |\vec{A}||\vec{B}|\cos\theta = \vec{A}^T\vec{B} = a_xb_x + a_yb_y + a_zb_z \quad (\text{dot product}) \quad (\text{D.3})$$

$$\begin{aligned} \vec{A} \times \vec{B} &= \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = \dots \quad (\text{cross product}) \\ &\dots = (a_yb_z - a_zb_y)\hat{i} - (a_xb_z - a_zb_x)\hat{j} + (a_xb_y - a_yb_x)\hat{k} \end{aligned} \quad (\text{D.4})$$

where the vectors \vec{A} and \vec{B} are represented in Cartesian coordinates as follows:

$$\begin{aligned}\vec{A} &= a_x \hat{i} + a_y \hat{j} + a_z \hat{k} \\ \vec{B} &= a_z \hat{i} + a_y \hat{j} + b_z \hat{k}\end{aligned}\tag{D.5}$$

Matrices on the other hand will be identified by bold capital Latin letters (i.e. **A**, **B**, **C**, ...) and represent multidimensional arrays of scalars. The elements of a matrix will be represented by the lower case Latin letter that represents the matrix name, followed by a series of subscripts that indicate the row and column. In this work we will make extensive use of 2-dimensional matrices for which the items can be represented as follows:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}\tag{D.6}$$

D.0.1 Attitude in 3D space

There are several ways to represent rotations in space such as Euler angles, direction cosine matrices, quaternions and Euler axis rotations. This work will focus almost exclusively on Euler angles and direction cosine matrices.

D.0.1.1 Rotation matrices

A rotation matrix, or direction cosine matrix (DCM), describes the orientation of one coordinate system with respect to another. The columns of the matrix are the unit vectors of one system as seen in the other. Assuming that the first reference system is identified by the unit vectors \hat{x}_1 , \hat{y}_1 , \hat{z}_1 , and the second system by \hat{x}_2 , \hat{y}_2 , \hat{z}_2 , 1_2R represents the DCM from the first system to the second.

$${}^1_2\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (\text{D.7})$$

Where:

$$\begin{aligned} \hat{X}_1 &= [r_{11}, r_{12}, r_{13}]^T \\ \hat{Y}_1 &= [r_{21}, r_{22}, r_{23}]^T \\ \hat{Z}_1 &= [r_{31}, r_{32}, r_{33}]^T \end{aligned} \quad (\text{D.8})$$

$$\begin{aligned} \hat{X}_2 &= [r_{11}, r_{21}, r_{31}]^T \\ \hat{Y}_2 &= [r_{12}, r_{22}, r_{32}]^T \\ \hat{Z}_2 &= [r_{13}, r_{23}, r_{33}]^T \end{aligned} \quad (\text{D.9})$$

The transformation in the reverse direction is accomplished with the inverse of the rotation matrix, which for a DCM is equal to its transpose.

$$\mathbf{R}\mathbf{R}^{-1} = \mathbf{I}; \quad \mathbf{R}^T\mathbf{R} = \mathbf{I}; \quad \mathbf{R}^{-1} = \mathbf{R}^T \quad (\text{D.10})$$

where I is the identity matrix.

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{D.11})$$

In three dimensions, the following properties apply for any rotation matrix $\mathbf{R}_{\mathbf{a},\theta}$ in R^3 , where \mathbf{a} is a rotation axis and θ a rotation angle,

$$\begin{aligned}
\mathbf{R}_{a,\theta}^T &= \mathbf{R}_{a,\theta}^{-1} \text{ (orthonormal)} \\
\det(\mathbf{R}_{a,\theta}^T) &= 1 \\
\mathbf{R}_{a,\theta+r} &= \mathbf{R}_{a,\theta} \mathbf{R}_{a,r} \\
\mathbf{R}_{a,0} &= \mathbf{I} \\
\text{eig}(\mathbf{R}_{a,\theta}^T) &= [1, e^{i\theta}] \\
\text{trace}(\mathbf{R}_{a,\theta}^T) &= 1 + 2 \cos \theta
\end{aligned} \tag{D.12}$$

D.0.2 Single axis rotations

The following represent the canonical set of simple, single axis rotations or fundamental rotations about one of the Cartesian coordinate axis.

$$\mathbf{R}_{\mathbf{x}}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{D.13}$$

$$\mathbf{R}_{\mathbf{y}}(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{D.14}$$

$$\mathbf{R}_{\mathbf{z}}(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{D.15}$$

D.0.3 Arbitrary rotations in 3D space

By combining the fundamental rotations above, we can obtain any arbitrary rotation. In order to do so, we have to execute an ordered sequence of rotations. In general a sequence of three rotations can represent any arbitrary orientation in the 3D space, where:

$$\mathbf{R}_{\mathbf{k}\mathbf{j}\mathbf{i}} = \mathbf{R}_{\mathbf{i}}(\psi)\mathbf{R}_{\mathbf{j}}(\theta)\mathbf{R}_{\mathbf{k}}(\phi) \quad (\text{D.16})$$

Of the 24 possible sequences, the most commonly adopted in the aeronautical world is the so-called 3-2-1 Euler sequence or yaw-pitch-roll sequence. In this work we will exclusively refer to this rotation sequence, which can be expressed as follows:

$$\begin{aligned} \mathbf{R}_{\mathbf{321}} &= \mathbf{R}_{\mathbf{x}}(\psi)\mathbf{R}_{\mathbf{y}}(\theta)\mathbf{R}_{\mathbf{z}}(\phi) = \dots \\ \dots &= \begin{bmatrix} \cos(\theta)\cos(\psi) & \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) \\ \cos(\theta)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) \\ -\sin(\theta) & \sin(\phi)\cos(\theta) & \cos(\phi)\cos(\theta) \end{bmatrix} \end{aligned} \quad (\text{D.17})$$

Where, ψ is the yaw angle, θ is the pitch angle and ϕ is the roll angle[83].

D.0.3.1 Euler Angles and Gimbal Lock

Although DCMs are a very convenient way of representing rotations, it is not uncommon to want to reduce the 3x3 matrix above in something simpler like the original Euler angles. In order to do so, we select a key elements of the $\mathbf{R}_{\mathbf{321}}$ matrix and invert the relationship as follows:

$$\begin{aligned} \psi &= \tan^{-1}(r_{12}/r_{11}) = \tan^{-1}\left(\frac{\cos(\theta)\sin(\psi)}{\cos(\theta)\cos(\psi)}\right) \quad (\text{Yaw}) \\ \theta &= -\sin^{-1}(r_{13}) = -\sin^{-1}(-\sin(\theta)) \quad (\text{Pitch}) \\ \phi &= \tan^{-1}(r_{23}/r_{33}) = \tan^{-1}\left(\frac{\sin(\phi)\cos(\theta)}{\cos(\phi)\cos(\theta)}\right) \quad (\text{Roll}) \end{aligned} \quad (\text{D.18})$$

From the relation above, we immediately see that for some angles the results are going to be not what we expect. The phenomenon is called gimbal lock and it occurs when $\cos(\theta) = 0 \rightarrow \theta = \pm\pi/2$. When this condition is verified, we have a zero in the denominator of both the yaw and pitch \tan^{-1} argument, and therefore we have an

indeterminate function on both angles. In other words, we are rotating the new coordinate frame twice about the same axis [84]. In a 321 rotation sequence, the first rotation axis is the initial z axis, followed by a rotation on the new y' axis and finally the last rotation on the new x'' created by the two prior sequential rotations. When $\theta = \pm\pi/2$, both the z and x'' axes coincide, creating a mathematical singularity [80].

Bibliography

- [1] K. Clowers, T. Clark, L. Harvill, R. Morency, and S. Rajulu, "Effects Of Varying Surface Inclines And Suit Pressure; Implications On Space Suit Design", 2008
- [2] H. Hu, L. Ding, C. Yang, X. Yuan, "Investigation On Ergonomics Characteristics Of Protective Clothing Based On Capture Of Three-Dimensional Body Movements", First International Conference on Digital Human Modeling, 2007
- [3] S. Baggerman, D. Berdich, M. Whitmore, "Human Systems Integration (HSI) Case Studies From The NASA Constellation Program", Human Systems Integration Symposium, 2009
- [4] A. S. Iberall, The experimental design of a mobile pressure suit, *Journal of Basic Engineering*, pp. 251-264, 1970.
- [5] D. Akin, "Experimental Assembly of Structures in EVA: Overview of Selected Results", NASA Conference on Space Construction, 1986
- [6] D. Dinua, R. Bidiugan, F. Natta, N. Houel, "Preliminary Study Of Accuracy And Reliability Of High-Speed Human-Motion Tracking Using Miniature Inertial Sensors", 9th Conference of the International Sports Engineering Association, Elsevier, 2012
- [7] B. Sherwood, "Comparing Future Options For Human Space Flight", *Acta Astronautica* Vol. 69, Elsevier, 2011
- [8] G. Welch, G. Bishop, "An Introduction To The Kalman Filter", 2006
- [9] K. Sloan, M. DeCesar, B. Knowles, S. McGuire, N. Moshtagh, J. Trump, Dr. L. Long, "Integrated Astronaut Control System For EVA", RASC-AL Forum, The Pennsylvania State University Mars Society, 2003
- [10] J. C. Moreno, E. R. de Lima, A. F. Ruiz, F. J. Brunetti, J. L. Pons, "Design And Implementation Of An Inertial Measurement Unit For Control Of Artificial Limbs: Application On Leg Orthoses", The 19th European Conference on Solid-State Transducers, 2006
- [11] D. Giansanti, G. Maccioni, F. Benvenuti, V. Macellari, "Inertial Measurement Units Furnish Accurate Trunk Trajectory Reconstruction Of The Sit-To-Stand Manoeuvre In Healthy Subjects", International Federation for Medical and Biological Engineering, 2007
- [12] H. Wang, X. H. Gao, M. H. Jin, L. B. Du, J. D. Zhao, H. Y. Hu, H. G. Cai, T. Q. Li, H. Liu, "A Passive Robot System for Measuring Spacesuit Joint Damping Parameters", International Conference on Robotics and Automation, 2003

- [13] R. Mahony, S.-H. Cha, T. Hamel, "A Coupled Estimation And Control Analysis For Attitude Stabilisation Of Mini Aerial Vehicles.", Australasian Conference on Robotics and Automation, 2006
- [14] C. E. Carr, D. J. Newman, "When Is Running More Efficient Than Walking In A Space Suit?", 35th International Conference on Environmental Systems, 2005
- [15] E. R. Bachmann, I. Duman, U. Y. Usta, R. B. McGhee, X. P. Yun, M. J. Zyda, "Orientation Tracking For Humans And Robots Using Inertial Sensors", International Symposium on Computational Intelligence in Robotics and Automation, 1999
- [16] W. Premerlani, P. Bizard, "Direction Cosine Matrix IMU: Theory", 2009
- [17] M. H. Jin, J. D. Zhao, H. Y. Hu, L. B. Du, X. H. Gao, T. Q. Li, Y. J. Zhao, H. Liu, "A Six DOF Passive Robot System For Spacesuit Measurement", Advanced Intelligent Mechatronics, IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2003
- [18] S. E. Jacobs, E. A. Benson, D. L. Akin, "Kinematic Analysis Of A Robotically Augmented Pressure Suit For Planetary Exploration", SAE International, 2007
- [19] P. Schmidt, "An Investigation Of Space Suit Mobility With Applications To EVA Operations", 2001
- [20] D. Comotti, "Orientation Estimation Based On Gauss-Newton Method And Implementation Of A Quaternion Complementary Filter", 2007
- [21] L. T. Aitchison, "A Comparison Of Methods For Assessing Space Suit Joint Ranges Of Motion", International Conference on Environmental Systems, AIAA, 2012
- [22] D. J. Newman, P. B. Schmidt, D. Rahn, N. Badler, and D. Metaxas, "Modeling The Extravehicular Mobility Unit (EMU) Space Suit: Physiological Implications For Extravehicular Activity (EVA)", 30th International Conference on Environmental Systems, 2000
- [23] P. B. Schmidt, D. J. Newman, E. Hodgson, "Modeling Space Suit Mobility: Applications To Design And Operations", International Conference on Environmental Systems, 2001
- [24] J. Favre, R. Aissaoui, B. M. Jolles, J. A. de Guise, K. Aminian, "Functional Calibration Procedure For 3D Knee Joint Angle Description Using Inertial Sensors", Journal of Biomechanics, Elsevier, 2009
- [25] D. Comotti , M. Ermidoro, "Report Of The Course "Progetto Di Microelettronica", 2007

- [26] D. Valish, K. Eversley, "Space Suit Joint Torque Measurement Method Validation", International Conference on Environmental Systems, AIAA, 2012
- [27] J. Zimmerman, "Human Postural Variations Between Simulation And On-Orbit Environments", 1996
- [28] A. J. Callaway, J. E. Cobb, I. Jones, "A Comparison Of Video And Accelerometer Based Approaches Applied To Performance Monitoring In Swimming", International Journal of Sports Science and Coaching Vol. 4, 2009
- [29] D. L. Akin, S. Saripalli, K. Hodges, K. Young, M. Di Capua, K. Davis, N. D'Amore, "Results From Desert FLEAS III: Field Tests Of EVA/Robotic Collaboration For Planetary Exploration", 42nd International Conference on Environmental Systems, AIAA, 2012
- [30] Y. Umetani, Y. Yamada, T. Morizono, T. Yoshida, S. Aoki, "Skil Mate, Wearable Exoskeleton Robot", IEEE International Conference on Systems, Man and Cybernetics, 1999
- [31] M. Lapinski, E. Berkson, T. Gill, M. Reinold, J. A. Paradiso, "A Distributed Wearable, Wireless Sensor System For Evaluating Professional Baseball Pitchers And Batters", International Symposium on Wearable Computers, 2009
- [32] W. J. E. Teskey, M. Elhabiby, N. El-Sheimy, "Inertial Sensing To Determine Movement Disorder Motion Present Before And After Treatment", Sensors, 2012
- [33] E. A. Sorenson, R. M. Sanner, C. U. Ranniger, "Experimental Testing Of A Power-Assisted", IEEE, 2009
- [34] A. Y. Kalery, I. V. Sorokin, M. V. Tyurin, "Human Space Exploration Beyond The International Space Station: Role Of Relations Of Human, Machine And The Earth", Acta Astronautica Vol. 67, Elsevier, 2010
- [35] H. Martin Schepers, H. F. J. M. Koopman, P. H. Veltink, "Ambulatory Assessment Of Ankle And Foot Dynamics", IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING Vol. 54, 2007
- [36] R. Mahony, T. Hamel, J.-M. Pfimlin, "Nonlinear Complementary Filters On The Special Orthogonal Group", IEEE TRANSACTIONS ON AUTOMATIC CONTROL Vol. 53, 2008
- [37] K. S. Tee, M. Awad, A. Dehghani, D. Moser, S. Zahedi, "A Portable Gait Monitoring System For Lower Limb Prosthetic Alignment", World Congress on Engineering, 2011
- [38] P. Moller, R. Loewens, I. P. Abramov, E. A. Albats, "EVA SUIT 2000: A JOINT EUROPEAN/RUSSIAN SPACE SUIT DESIGN", Acta Astronautica Vol. 36, Elsevier, 1995

- [39] A. Reinhardt, "Results And Applications Of A Space Suit Range-Of-Motion Study", NASA Technical Memorandum, 1989
- [40] H. J. Luinge, P.H. Veltink, "Measuring Orientation Of Human Body Segments Using Miniature Gyroscopes And Accelerometers", *Medical and Biological Engineering and Computing* Vol. 43, 2005
- [41] C. Schiefer, T. Kraus, E. Ochsmann, I. Hermanns, R. Ellegast, "3D Human Motion Capturing Based Only On Acceleration And Angular Rate Measurement For Low Extremities", *Digital Human Modeling: Lecture Notes in Computer Science* Vol. 6777, Springer, 2011
- [42] H. J. Luinge, "Inertial Sensing Of Human Movement", 2002
- [43] J. H. M. Bergmann, R. E. Mayagoitia, I. C. H. Smith, "A Portable System For Collecting Anatomical Joint Angles During Stair Ascent: A Comparison With An Optical Tracking Device", *Dynamic Medicine* 8:3, 2009
- [44] C. S. Kothera, M. Jangid, J. Sirohi, N. M. Wereley, "Experimental Characterization And Static Modeling Of McKibben Actuators", *Journal of Mechanical Design* Vol. 131, 2009
- [45] G. Baldwin, R. Mahony, J. Trumpf, T. Hamel, T. Chevion, "Complementary Filter Design On The Special Euclidean Group SE(3)", *European Control Conference*, 2007
- [46] J. Zhao, Y. Liu, H. Cai, H. Liu, "An Improved Algorithm Of Measuring Extravehicular Mobility Unit (EMU) Spacesuit Joint Damping Parameters For The Old Passive Robot System", *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008
- [47] C. E. Carr, D. J. Newman, "Characterization Of A Lower-Body Exoskeleton For Simulation Of Space-Suited Locomotion", *Acta Astronautica* Vol. 62, 2007
- [48] Terry R. Hill and Brian J. Johnson, "EVA Space Suit Architecture: Low Earth Orbit Vs. Moon Vs. Mars", *2010 IEEE Aerospace Conference*, 2010
- [49] M. El-Gohary, L. Holmstrom, J. Huisinga, E. King, J. McNames, F. Horak, "Upper Limb Joint Angle Tracking With Inertial Sensors", *33rd Annual International Conference of the IEEE EMBS*, 2011
- [50] C. Blackledge, S. Margerum, M. Ferrer, R. Morency, S. Rajulu, "Modeling The Impact Of Space Suit Components And Anthropometry On The Center Of Mass Of A Seated Crewmember", *2010 3rd International Conference on Applied Human Factors and Ergonomics*, 2009

- [51] S. E. Jacobs, M. Di Capua, S.-A. A. Husain, A. Mirvis, D. L. Akin, "Incorporating Advanced Controls, Displays And Other Smart Elements Into Space Suit Design", SAE International 2009-01-2472, 2009
- [52] H. Dejnabadi, B. M. Jolles, E. Casanova, P. Fua, K. Aminian, "Estimation And Visualization Of Sagittal Kinematics Of Lower Limbs Orientation Using Body-Fixed Sensors", IEEE Transactions of Biomedical Engineering Vol. 53, 2006
- [53] Shane M. McFarland, "Injury Potential Testing Of Suited Occupants During Dynamic Spacecraft Flight Phases", AIAA, 2011
- [54] D. Comotti, M. Ermidoro, "Progetto Di Microelettronica Sviluppo Di Algoritmi Per La Stima Dell'orientamento Di Un Sensore Inerziale", 2011
- [55] M. Ling, A. Young, "IMUSim: Simulating Inertial And Magnetic Sensor Systems In Python", 10th SciPy Conference, 2011
- [56] David L. Akin, S. Saripalli, M. Di Capua, "Developing And Testing Technologies For Planetary Surface Exploration And Habitation", AIAA SPACE 2011 Conference and Exposition, 2011
- [57] A. D. Young, M. J. Ling, D. K. Arvind, "IMUSim: A Simulation Environment For Inertial Sensing Algorithm Design And Evaluation", 10th ACM/IEEE International Conference on Information Processing in Sensor Networks, 2011
- [58] H. J. Luinge, "Inertial Sensing Of Human Movement", 2002
- [59] J. Matty, "Results And Analysis From Space Suit Joint Torque Testing", 40th International Conference on Environmental Systems, AIAA, 2012
- [60] Shane E. Jacobs, Massimiliano Di Capua and David L. Akin, "Neck-Entry Suitports: A Novel Concept Utilizing Morphing Upper Torso Technology", SAE International 2009-01-2571, 2009
- [61] D. L. Akin, S. Saripalli, M. Di Capua, K. Davis, K. Hodges, K. Young, "Field Analogue Simulations Investigating EVA/Robotic Collaboration In Lunar Exploration", 41st International Conference on Environmental Systems, AIAA, 2011
- [62] R. L. Kobrick, C. E. Carr, F. Meyen, A. R. Domingues, D. J. Newman, S. E. Jacobs, "Using Inertial Measurement Units For Measuring Spacesuit Mobility And Work Envelope Capability For Intravehicular And Extravehicular Activities", 63rd International Astronautical Congress, 2012
- [63] M. Di Capua, D. L. Akin, K. Davis, "Development And Testing Of Advanced Pressure Suit Technologies And Analogues For Earth-Based Simulations", 41st International Conference on Environmental Systems, AIAA, 2011

- [64] M. Di Capua, D. L. Akin, "Body Pose Measurement System (BPMS): An Inertial Motion Capture System For Biomechanics Analysis And Robot Control From Within A Pressure Suit", 42nd International Conference on Environmental Systems, AIAA, 2012
- [65] A. F. J. Abercromby, S. S. Thaxton, E. A. Onady, S. L. Rajulu, "Reach Envelope And Field Of Vision Quantification In Mark III Space Suit Using Delaunay Triangulation", 2006
- [66] M. Di Capua, "Augmented Reality For Space Applications", 2008
- [67] A. A. Kocielniak, "Development And Testing Of A Metabolic Workload Measurement System For Space Suits", 2007
- [68] M. Di Capua, D. L. Akin, J. C. Brannan, "MX-3: Design And Technology Development Of An Advanced All-Soft High Mobility Planetary Space Suit", 40th International Conference on Environmental Systems, 2010
- [69] D. J. Newman, M. Canina, G. L. Trotti, "Revolutionary Design For Astronaut Exploration Beyond The Bio-Suit System", Space Technology and Applications International Forum, 2007
- [70] N. C. Jordan, "Multidisciplinary Spacesuit Modeling And Optimization: Requirement Changes And Recommendations For The Next-Generation Spacesuit Design", 2006
- [71] K. Bethke, "The Second Skin Approach: Skin Strain Field Analysis And Mechanical Counter Pressure Prototyping For Advanced Spacesuit Design", 2005
- [72] A. J. Nejman, "Kinematic Analysis and Joint Hysteresis Modeling For A Lower-Body, Exoskeleton-Style Space Suit Simulator", 2011
- [73] S. E. Jacobs, "Pressure-Constrained, Reduced-DOF, Interconnected Parallel Manipulators With Applications To Space Suit Design", 2009
- [74] R. W. Orloff, D. M. Harland, "Apollo: The Definitive Sourcebook", Springer Praxis, 2006
- [75] I. P. Abramov, A. I. Skoog, "Russian Spacesuits", Springer Praxis, 2003
- [76] G. L. Harris, "The Origins And Technology Of The Advanced Extravehicular Space Suit", Univelt Incorporated, 2001
- [77] K. S. Thomas, H. J. McMann, "US Spacesuits", Springer Praxis, 2006
- [78] L. D. Kozloski, "U.S. Space Gear: Outfitting The Astronaut", Smithsonian Institution, 1994

- [79] W. J. Larson, L. K. Pranke, "Human Spaceflight: Mission Analysis And Design", McGraw-Hill Higher Companies, 1999
- [80] B. Wie, "Space Vehicle Dynamics And Control", American Institute of Aeronautics and Astronautics Inc., 1998
- [81] A. Bichara, A. Del Fra, "Geometrica: Per La Nuova Laurea In Ingegneria", L.S.D., 2001
- [82] R. H. Battin, "An Introduction To The Mathematics And Methods Of Astrodynamics", American Institute of Aeronautics and Astronautics Inc., 1999
- [83] P. C. Hughes, "Spacecraft Attitude Dynamics", Dover Publications, Inc., 2004
- [84] Slabaugh, G.G. (1999): Computing Euler angles from a rotation matrix. Report from personal website. <http://www soi.city.ac.uk/~sbbh653/publications/euler.pdf>
- [85] D. Akin, J. Corde Lane, B. J. Roberts, S. Weisman, "Robotic Capabilities for Complex Space Operations", AIAA Space 2001 Conference and Exposition, Aug. 28-30 2001, AIAA-2001-4538
- [86] A. Young, "Spacesuits", Smithsonian Natrional Air And Space Museum, PowerHouse books, 2009
- [87] A. Minetti, "Biomechanics: Walking on other planets", Nature, ISSN: 0028-0836, EISSN: 1476-4687, 2001
- [88] M. Stolen, B. Dillow, S. Jacobs, D. Akin, "Interface for EVA Human-Machine Interaction, 38th International Conference on Environmental Systems, 2008
- [89] S. Jacobs, J. Braden, D. Akin, D. Graziosi, "Morphing Upper Torso: A Novel Concept in EVA Suit Design", 36th International Conference on Environmental Systems, 2006
- [90] S. Jacobs, D. Akin, "Dynamic Analysis of an Adjustable Torso Design for a Planetary Pressure Suit", 38th International Conference on Environmental Systems, 2008
- [91] J. Chardonens, J. Favre, K. Aminian, "An effortless procedure to align the local frame of an inertial measurement unit to the local frame of another motion capture system" *JournalofBiomechanics*, June 2012
- [92] D. Ren, L. Wu, M. Yan, M. Cui, Z. You, M. Hu "Design and Analyses of a MEMS Based Resonant Magnetometer", *Open Access Journal of Sensors*, ISSN 1424-8220, 2009
- [93] Q. Yuan, I-Ming Chen, "Human velocity and dynamic behavior tracking method for inertial capture system", *Journal of Sensors and Actuators*, June 2012

Datasheets: (Dec-10-2012)

- [94] Analog Devices, Digital 3 Axis Accelerometer Datasheet, <http://www.analog.com/en/mems-sensors/mems-inertial-sensors/adxl345/products/product.html>
- [95] Honeywell, Three-Axis Digital Compass IC HMC5883L Datasheet, http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense_Brochures-documents/HMC5883L_3-Axis_Digital_Compass_IC.pdf
- [96] InvenSense, ITG-3200 Integrated Triple-Axis Digital-Output Gyroscope Datasheet, <http://invensense.com/mems/gyro/itg3200.html>
- [97] Digi, X-Bee and ZigBee documentation reference page, <http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/zigbee-mesh-module/xbee-zb-module#docs>
- [98] Roving Networks, Bluetooth SMD Module - RN-41, <http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Wireless/Bluetooth/rn-41-ds-v3.3r.pdf>
- [99] Texas Instruments, TPS61200: Low Input Voltage Synchronous Boost Converter Datasheet, <http://www.sparkfun.com/datasheets/Prototyping/tps61200.pdf>
- [100] Battery Space, *LiFePO₄* 18650 Rechargeable Cell: 3.2V 1500 mAh, 4.5A Rate, 4.32Wh Datasheet, http://www.batteryspace.com/prod-specs/919_1.pdf
- [101] ATmega328P, 8-bit microcontroller with 32K Bytes In-System Programmable Flash Memory Datasheet, <http://www.atmel.com/Images/doc8161.pdf>
- [102] Piezoelectric Flex Sensor, [http://www.sparkfun.com/datasheets/Sensors/Flex/FLEXSENSOR\(REVA1\).pdf](http://www.sparkfun.com/datasheets/Sensors/Flex/FLEXSENSOR(REVA1).pdf)
- [103] LY530AL Single Axis Gyro, <http://www.sparkfun.com/datasheets/Sensors/IMU/LY530ALH.pdf>
- [104] LPR530AL Dual Axis Gyro, <http://www.sparkfun.com/datasheets/Sensors/IMU/lpr530al.pdf>
- [105] ADXL335 Triple Axis Accelerometer, <http://www.sparkfun.com/datasheets/Components/SMD/adx1335.pdf>
- [106] HMC5843 Triple Axis Magnetometer, <http://www.sparkfun.com/datasheets/Sensors/Magneto/HMC5843.pdf>
- [107] Sparkfun 9DOF Razor IMU, (<http://www.sparkfun.com/products/9623>)

Open Source Software references: (Dec-10-2012)

[108] Arduino project home page, <http://www.arduino.cc/>

[109] How-To Guide to using an Arduino as an AVR ISP, <http://arduino.cc/en/Tutorial/ArduinoISP>

[110] How-To Guide to flashing the Arduino Bootloader, <http://arduino.cc/en/Hacking/Bootloader>

[111] FreeIMU: an Open Hardware Framework for Orientation and Motion Sensing <http://www.varesano.net/projects/hardware/FreeIMU>