The Pennsylvania State University

The Graduate School

College of Engineering

PORTABLE BIOMECHANICAL ASSESSMENT SUITE (PBAS)

A Thesis in

Mechanical Engineering

by

Michael L. Boyd

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2010

The thesis of Michael L. Boyd was reviewed and approved* by the following:

H. Joseph Sommer III
Professor of Mechanical Engineering
Thesis Advisor

Stephen J. Piazza
Associate Professor of Kinesiology, Mechanical Engineering, and Orthopaedics and Rehabilitation
Thesis Reader

Karen A. Thole
Professor of Mechanical Engineering
Department Head of Mechanical and Nuclear Engineering

*Signatures on file in the Graduate School

Abstract

The Portable Biomechanical Assessment Suite (PBAS) is a suite of sensors to measure the kinematics and kinetics of the human body during lifting tasks. The suite is made up of ten inertial measurement units (IMUs) and the Novel Pedar pressure insoles. The IMUs are made up of an Xbee radio, a triple-axis accelerometer, a dual-axis gyroscope, and a single-axis gyroscope. The data are sent via the 802.15.4 wireless network protocol to a collection device connected to a computer and are stored in a binary data file that is then read and processed by a Matlab program. The output of the program is four data matrices: x-acceleration, y-acceleration, z-angular velocity, and the calculated angular position of each IMU. The pressure insoles are synchronized with the IMUs to measure the ground reaction forces.

An initial test was run to determine how battery life, the zero voltage level, and dropout rate change with data transmission time. The tests showed that the IMUs could easily run for five hours with no effect on the dropout rate. The accelerometer was found to have very little zero-drift while the gyroscopes do exhibit some zero drift. This will need to be taken into account for long tests.

Three pendulum tests were run to test the IMUs, all yielding good results. The first test used three preliminary IMUs made on breadboards to validate the circuit design and general concept. The next two tests used five final IMUs to measure the kinematics of the pendulum. One test had all IMUs measuring in the sagittal plane while the second test used three sagittal plane IMUs and two IMUs reprogrammed to measure the corresponding signals in the coronal plane. Both tests showed that the IMU signals matched what was expected and validated the network of IMUs for future use in biomechanical testing.

Table of Contents

List of Figures

List of Tables

1. Introduction

The ultimate goal of this project is to help decrease the frequency of musculoskeletal disorders related to occupational lifting tasks. When forces within the body are repetitive or excessive, the risk of injury is increased. As a result, laboratory methods in biomechanics for modeling these musculoskeletal forces have been developed. Due to the instrumentation required to measure the input data (body motion and external forces acting on the body from the floor and load), current biomechanical safety models are based on conditions and activities approximated in a laboratory setting.

More specifically, the goal of this project is to develop a group of sensors that can be worn in the workplace to measure the loading on the lower back during occupational lifting tasks throughout the work day. This data can then be used to develop intervention strategies and keep workers safe.

A prototype Portable Biomechanical Assessment Suite (PBAS) has been developed to measure body kinematics and external loading in unstructured field conditions. The PBAS is able to be worn unobtrusively by a worker for the duration of testing.

The PBAS has immediate application by measuring the exposure of workers to repetitive motions and quantifying loads used as the bases for existing safety models. This system can help improve the current musculoskeletal loading models by providing data on body kinematics and external loading in actual field conditions, which was previously unavailable. The resulting information could lead to exposure-response relationships for musculoskeletal diseases in the work place and could ultimately lead to a more effective way to alleviate these adverse health effects.

The PBAS was designed to measure the stress on the musculoskeletal system in terms of body motion (kinematics) and forces causing or resulting from the motion (kinetics) during a full work day. These data can be used directly in biomechanical models that estimate forces on important structural areas of the body (e.g., spinal discs). It can also be used in current lifting equations to determine safe combinations of lifting loads and the frequency of lifts as well as in assessing repetitive motion of body segments.

This prototype focuses on the input to two widely accepted biomechanical safety models associated with lifting tasks (Freivalds et al., 1984 and Waters et al., 1993). Kinematic data needed for these models includes the motion of the head, thorax, pelvis, upper extremities, and lower extremities. Only motion in the sagittal plane will be measured. The kinetic data needed for these models are the external forces applied to the body at the hands and feet.

Kinematic data are measured by Inertial Measurement Units (IMUs) consisting of a three-axis accelerometer, a two-axis gyroscope, and a single-axis gyroscope. The IMUs measure two acceleration signals and one gyroscope signal in the sagittal plane or two acceleration signals and one gyroscope signal in the coronal plane and send the data via the 802.15.4 wireless network protocol. Data are collected by a computer where it is processed and organized by a MATLAB program for use by biomechanical software.

There has been significant work done in the area of instrumented biomechanical testing, but nothing for lifting tasks in the workplace. The PBAS will provide data that has never been collected before and will eventually be able to aid in preventative safety measures for workers performing lifting tasks.

2. Literature Review

2.1. Lifting

2.1.1. NIOSH Lifting Equation

Considerable work has been done to study lifting tasks and assess potential injury to workers if lifting is performed incorrectly. NIOSH (National Institute for Occupational Safety and Health) developed a revision of its lifting equation in 1991 that was used to recognize certain tasks that would lead to lower back pain (Waters et al. 1993). The revision came about because it could only be applied to sagittal plane lifting tasks and did not take into account other lifting factors. Both lifting equations were developed using scientific literature and the judgments of experts in the fields of biomechanics, psychophysics, and work physiology.

Revisions of the 1991 equation include allowing assessment of asymmetrical lifting tasks, lifting of objects with different sizes and shapes of handles and containers, and longer work periods and lifting frequencies. There were three criteria used to determine the components of the lifting equation: biomechanical, physiological, and psychophysical. The biomechanical criterion controls the effect of stress on the lumbar-sacrum region. The physiological criterion looks at the stress and fatigue from repetitive lifting tasks. The psychophysical criterion deals with the lifters' perception of their lifting ability.

The bases for the NIOSH lifting equation are three numbers: the standard lifting location (STL), the load constant, and the multipliers. The STL is the point used to assess a "worker's lifting posture." For the old equation, the STL was defined as a point at a height of 75 cm from the floor and a horizontal distance of 15 cm from the center point between the worker's ankles. The 1991 equation uses the 75 cm height, but changed the horizontal distance from 15 to 25 cm. This change was made as findings showed the larger distance was used during lifting so the worker's body did not interfere with lifting the load. Equation 1 shows the calculation for the recommended weight limit (RWL) for lifting tasks;

Equation 1: 1991 NIOSH lifting equation

$$RWL = LC \times HM \times VM \times DM \times AM \times FM \times CM$$

where, LC = load constant and HM, VM, DM, AM, FM, and CM are the multipliers described below.

The load constant is the maximum load that can be lifted at the STL under optimal lifting conditions. For ideal conditions (all multipliers in lifting equation equal to 1), the load constant had to be acceptable to 75% of women and 90% of men and the compressive force in the lumbosacral region had to be less than 3.4 kN. For the 1991 equation, the load constant was decreased from 40 to 23 kg.

There are six coefficients that require mathematical expressions for the lifting equation. These coefficients are used to decrease the load constant to account for less than ideal lifting conditions. The horizontal multiplier is used to account for the horizontal distance between the load and spine which can cause a moment on the spine and increase the force on the lumbosacral region. The horizontal multiplier is given in Equation 2. H is defined as the horizontal distance from hands to the midpoint between the ankles.

Equation 2: Horizontal multiplier

$$HM = \frac{25}{H_{cm}}$$

$$HM = \frac{10}{H_{in}}$$

The vertical multiplier accounts for increased forces on the spine when the load starts on the floor or at shoulder height. The vertical multiplier is shown in Equation 3. V is defined as the vertical height of the hands from the floor while lifting in centimeters or inches.

Equation 3: Vertical multiplier

$$VM = (1 - 0.003|V_{cm} - 75|)$$

$$VM = (1 - 0.0075|V_{in} - 30|)$$

The distance multiplier is used when the distance the load is moved is near the maximum possible (i.e. floor to shoulder height). Studies used by the NIOSH committee showed that a larger lifting distance decreased a person's acceptable lifting weight. The distance multiplier is shown in Equation 4. D is the total distance the load is moved in either centimeters or inches.

Equation 4: Distance multiplier

$$DM = 0.82 + \frac{4.5}{D_{cm}}$$

$$DM = 0.82 + \frac{1.8}{D_{in}}$$

The asymmetric multiplier is used for lifting loads away from the sagittal plane and is given in Equation 5. A is the angle between the sagittal plane and the vertical plane that intersects the center point between the ankles and the center point between the hands at the asymmetric location.

Equation 5: Asymmetric multiplier

$$AM = 1 - 0.0032 \times A$$

The coupling multiplier describes the condition of handles on the load. Appropriate handles allow for easier lifting and reduce the possibility of dropping the load. Table 1 shows the coupling multiplier.

Table 1: Coupling multiplier

	$V < 75$ cm (30 in)	$V \geq 75$ cm (30 in)
Couplings	Coupling multipliers	
Good	1·00	1·00
Fair	0·95	1·00
Poor	0·90	0·90

The frequency multiplier is obtained from Table 2.

Table 2: Frequency multiplier

Frequency lifts/min	Work duration ≤ 1 h $V < 75$	≤ 1 h $V \geq 75$	≤ 2 h $V < 75$	≤ 2 h $V \geq 75$	≤ 8 h $V < 75$	≤ 8 h $V \geq 75$
0·2	1·00	1·00	0·95	0·95	0·85	0·85
0·5	0·97	0·97	0·92	0·92	0·81	0·81
1	0·94	0·94	0·88	0·88	0·75	0·75
2	0·91	0·91	0·84	0·84	0·65	0·65
3	0·88	0·88	0·79	0·79	0·55	0·55
4	0·84	0·84	0·72	0·72	0·45	0·45
5	0·80	0·80	0·60	0·60	0·35	0·35
6	0·75	0·75	0·50	0·50	0·27	0·27
7	0·70	0·70	0·42	0·42	0·22	0·22
8	0·60	0·60	0·35	0·35	0·18	0·18
9	0·52	0·52	0·30	0·30	0·00	0·15
10	0·45	0·45	0·26	0·26	0·00	0·13
11	0·41	0·41	0·00	0·23	0·00	0·00
12	0·37	0·37	0·00	0·21	0·00	0·00
13	0·00	0·34	0·00	0·00	0·00	0·00
14	0·00	0·31	0·00	0·00	0·00	0·00
15	0·00	0·28	0·00	0·00	0·00	0·00
>15	0·00	0·00	0·00	0·00	0·00	0·00

Note:
‡ values of V are in cm; 75 cm = 30 in.

There are some limitations of the lifting equation. It is designed for specific biomechanical, physiology, and psychophysical data and assumptions and therefore cannot be applied to all lifting tasks. The limitations of the equation are:

1. It assumes manual handling tasks other than lifting (walking, carrying, pulling, pushing, etc.) do not require a significant amount of energy.
2. Unexpected conditions, such as slips or falls, are not included. Also, harsh environmental conditions would require further testing to determine changes in workers' behavior.
3. The 1991 equation does not include factors for one-handed lifting, lifting while seated or kneeling, lifting of people, lifting of unsafe objects, lifting in small spaces, lifting of wheel-barrels, shoveling, or high-speed lifting.
4. A 0.4 or 0.5 coefficient of static friction between the floor and the workers' shoes was assumed.
5. The 1991 equation applies the same level of risk to lifting and lowering.

2.1.2. Lifting maximum acceptable loads (Freivalds model)

The basic assumption of this model is that the body is made up of rigid links joined by simple joints. This is reasonable for the arms and legs, but not as good for the trunk which is made up of vertebrae, intervertebral discs, and cartilage that allow for more complex motion. There are seven links used in this model: foot, lower leg, upper leg, pelvis, head-thorax-abdomen, hand-forearm, and upper arm. The trunk is divided into two links, pelvis and head-thorax-abdomen, at the L_5/S_1 (lumbar-sacrum) vertebrae. The ankle is assumed to remain fixed to provide a reference point. The wrist is not treated as a joint as there is very little movement in planar motions (Freivalds et al. 1984).

There are four steps in determining forces and moments on the body from the motion input data: (1) resolution of the position of the body from the angles at each joint; (2) determination of the angular velocities and angular accelerations at each joint which gives the linear accelerations of each link; (3) calculation of inertial forces and resistance moments due to acceleration; (4) calculation of reactive moments and forces at each joint exerted by the muscles to overcome the resultant forces due to external loads and body weight.

There are three inputs to the model: (1) mass and length of each body segment; (2) description of the motion of each segment for the lifting task; (3) load being moved. The mass and length of the body segments comes from Dempster's body segment parameters and initial calibration. Dempster's body segment parameters are shown in Appendix A.

The output of the model was the following for each time step: (1) elapsed time; (2) angle of each segment; (3) angular velocity of each segment; (4) acceleration of the center of gravity for each segment; (5) reactive forces at each joint; (6) reactive torques at each joint; (7) compressive and shear forces at the lumbar-sacrum.

Freivalds ran an experiment to validate his model that had six male subjects lift four different boxes with various weights. He used a force plate to measure the ground reaction forces and a camera with stroboscopic light to capture the body segment motion. This was done by tracking the positions of reflective markers placed on the joints of each subject.

The results of the experiment validated the model they developed based on predicted and measured ground reaction forces matching well. Along with this, predicted lumbar-sacral forces correlated with the vertical ground reaction forces, which helped to validate the model and provide a way to estimate these forces.

2.2. Walking Models

2.2.1. Inverse dynamics based only on measured kinematics

Ren et al. (2008) describe a three-dimensional model for inverse dynamics analysis over a complete gait cycle. Predicted ground forces and moments were compared with force plate data to validate the model.

The body was represented by 13 segments: head, torso, pelvis, upper arms, lower arms, thighs, shanks, and feet. A sum of forces and moments was done on each segment. The only significant external forces and moments exerted on the body during walking are the ground reactions. In single support phase, where only one leg is in contract with the ground, the ground reaction force can be obtained directly from Newton's second law. When the subject is in double support phase, the problem becomes indeterminate.

In order to determine the forces during the double support phase, the authors developed the "Smooth Transition Assumption" (STA) which says:

1. "In double support phase, the ground forces and moments on the trailing foot change smoothly toward zero."

2. “The ratios of ground reactions to their values at contralateral heel strike (i.e. the non-dimensional ground reactions) can be expressed as functions of double support duration (known as transition functions).” These transition functions were found by a trial and error method.

The authors used the following algorithm to do their inverse dynamic analysis:

1) In single support, the ground reactions are found from a force equation on the subject.
2) In double support, the ground forces on the trailing foot are found using the STA. The ground forces on the leading foot are then found using the force equation.
3) The joint forces for each joint are calculated sequentially starting from the ones in contact with the ground and working away.
4) In single support, the ground moments are found from a moment equation.
5) In double support, the ground moments on the trailing foot come from the STA and then the ground moments of the leading foot are calculated using the moment equation.
6) The moments ate each joint are calculated in a similar way as the joint forces.

The authors tested three subjects to validate their model. Data was measured using a system of six cameras and two force plates. Good results were obtained for sagittal plane ground forces and moments as well as in the comparison of the predicted and measured joint reactions during the swing phase of the gait cycle. This data validated the model.

2.2.2. Using pressure insoles to obtain ground reaction forces

Fong et al. (2008) discussed a method to calculate the complete ground reaction forces using only data from pressure insoles. This was significant as the method did not require any motion capture systems and therefore could be used outside the laboratory. Several different devices have been developed to measure ground reaction forces, but pressure insoles were chosen for their cost and wide range of uses.

Testing was done on five male subjects who walked at their natural pace on a walking path in a laboratory. The subjects stepped on a force plate in the middle of the path with their right foot. Pressure insoles were worn in their shoes and measured the ground reaction forces at the same time. Each insole had 99 sensors which covered the entire foot.

The force plate and insole data were edited to look at one stride, from take off before the foot hit the force plate to the next take off from the force plate. A linear regression analysis was done to recreate the ground reaction force in three directions: anterior-posterior, medial-lateral, and vertical.

The complete ground reaction forces in the three directions were calculated using a regression model. Figure 1 shows which sensors were used for each of the three regression models. Four, six, and four sensors of the possible 99 were used for anterior-posterior, medial-lateral, and vertical respectively.

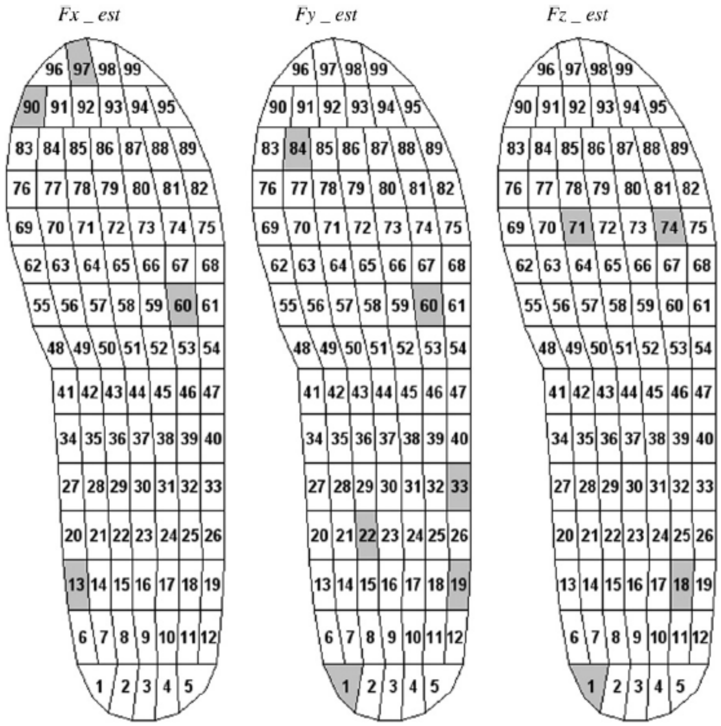

Figure 1: Location of sensors in right foot used in the three regression models (F_x: anterior-posterior, F_y: medial-lateral, F_z: vertical)

The model was very good in estimating the ground reaction forces in the anterior-posterior and vertical directions, with a fair approximation in the medial-lateral direction. This validation shows that this method could be used to find the complete ground reaction forces without the use of force plates or motion capture equipment.

2.2.3. Multi-segment foot model

Buczek et al. (2006) included the forces and moments between foot segments that are typically ignored in modeling. They used a free-body diagram and inverse dynamics to provide a force system that is affected by the inclusion of these internal forces and moments acting on the foot. Their work discusses specifically the foot, but has given a general format for modeling in biomechanics. Their theory shows that force systems do not need to be at the endpoints of the geometrical models of the body segments in order to use inverse dynamics. This condition is used by commercial software with the exception of placing the force system of the foot at the center of pressure rather than at the end of the body segment model.

2.3. Wireless Data Capture

2.3.1. Timed up and go test

The Timed Up and Go test is a widely used test to determine the balance and mobility of elderly patients, in particular those with Parkinson's disease (Zampieri et al. 2009). The problem with the test is that it is not sensitive enough to detect the disease in early stages. The test involves a series of sit-to-stand, stand-to-sit, walking and turning tasks with the test result being the time to complete the tasks. The authors developed a system called iTUG (Instrumented timed up and go) which has the subjects completing the Timed Up and Go test while wearing five portable inertial sensors. The purpose was to obtain more specific data about the certain tasks in the test and to look for a correlation with severity of the disease and the results of the test. The portable sensors involved two single-axis gyroscopes, two two-axis gyroscopes, and a sensor with a two-axis gyroscope and a three-axis accelerometer. They sampled data at 200 Hz and recorded it in a flash memory card.

They found that the iTUG can detect specific discrepancies in the gait and turning of early to mid stage Parkinson's patients and parts of the iTUG did correlate with the severity of the disease. Patients with untreated Parkinson's showed differences in their trunk rotation and arm swing as well as turning slower.

2.3.2. Wireless system for quantifying gait

There are many systems used for quantifying gait, but almost all need to be used in a laboratory setting. Initial wearable sensor systems have used wires to connect the sensors to a central processor (LeMoyne et al. 2009). The system used by the authors was the G-Link Wireless Accelerometer Nodes which consisted of a radio, three-axis accelerometer, and a rechargeable battery. This system provided a 70 meter line of sight wireless transmission range. G-Link allowed for an adjustable sampling rate up to 2048 Hz. The data was stored in a "datalogging mode" for wireless transmission.

In the experiment, one sensor attached just above the ankle was used to measure accelerations as the subject walked through a hallway. The data was downloaded from the device after the trial was completed. The authors were able to obtain important gait data and with more sensors, the system could be developed into a multiple body segment gait analysis tool.

2.3.3. Wireless accelerometer network for gait analysis

Stamatakis et al. (2008) developed a system of ten sensors to measure accelerations of different segments of the human body. The sensors contain a three-axis accelerometer and a microcontroller with a wireless radio and antenna. The data was collected by a wireless base station connected to a computer. The modules are powered by two AAA batteries and are 18.9 cm^2 in area without the batteries.

The wireless protocol used for this system is IEEE 802.15.4. This protocol enables 250 kbps of data transfer. The base station was set up as the network coordinator with the other modules being used as end devices. The base station sent a signal for the all of the devices to start sampling at a frequency of 100 Hz. The data from the three-axis accelerometer was stored in one of two rotating buffers, which took 170 ms to fill. The collector then collected the data from the buffers of each device sequentially. No acknowledge was sent back from the base device as their protocol did not allow multiple transmission attempts. The microprocessor onboard ran the radio while the sampling of the Analog-to-Digital converter was coordinated with a 10 ms timer and a memory controller.

When a data buffer was collected by the central collection unit, it was sent to the computer via a RS-232 connection. A Java application was run to show the accelerations and monitor for lost data. Sampling was ended with a signal sent from the base device.

Their devices could run for 55 hours using two AAA batteries with a transmission range of fifteen meters. Walking tests were run on two groups of elderly patients to validate the sensors.

2.3.4. Ambulatory assessment on a wearable sensor system

Liu et al. (2009) discuss a procedure for measuring the angles of the lower limb segments and other gait data using only accelerometers. It is called the double-sensor difference algorithm and is capable of finding the angle of rotation of a body segment about two local axes. The algorithm was developed to calculate the flexion/extension and abduction/adduction angles of the thigh in 3D space.

A three-axis accelerometer placed at a certain position on a body segment will have a measured acceleration as shown in Equation 6.

Equation 6: Measured acceleration of an accelerometer at position r

$$a = R(g + a_0) + \dot{\omega} \times r + \omega \times (\omega \times r)$$

In Equation 6, R is the attitude matrix of the body segment with respect to the global reference frame (ground), a_0 is the acceleration of the origin of the coordinate system with respect to ground, g is the gravitational field, and ω is the angular velocity.

It was assumed in this model that the body segments were rigid and the subjects walked with very little trunk swing, skin artifacts, and no significant rotation of the leg. When two accelerometers are attached at two different positions with each local axis in the same direction, the acceleration due to gravity, translational acceleration, skin artifact, and other noise from the two sensors should be the same based on the given assumptions. The only difference should be the angular acceleration.

The algorithm calls for placing two accelerometers on the same segment facing the same direction. The angular displacement, velocity, and acceleration can then be calculated from equations describing the relationships between the two accelerometers. This method was tested first on a mechanical arm then on eight subjects. The experiments validated the double-sensor algorithm.

2.4. Discussion

The work shown here represents a broad range of data collected in various areas of biomechanics. Substantial work has been done studying lifting using motion capture systems and force plates, but data has never been collected outside the laboratory. These experiments were the basis for the PBAS as the goal of this project is to collect lifting data outside of a laboratory.

There has also been work done on measuring gait characteristics, both using motion capture equipment and force plates and using wireless sensors. These studies validated the wireless systems and provide a good start for the PBAS.

However, walking is much different than lifting, so a different system with more sensors was needed. The work done by Fong et al. (2008) is valuable as the PBAS also uses pressure insoles to obtain ground reactions, but his regression models cannot be used as they have only been validated for walking experiments.

3. Analytical model

3.1. Body segment parameters

Body segment parameters are used to determine mass and length characteristics of limb segments. These parameters are ratios of mass and center of mass positions for all limb segments. The body segment parameters are used with measurements of the test subject to determine the mass and length values needed for body dynamics. A table of body segment parameters and equations for them are shown in Appendix A.

3.2 Kinetic Model

A general two-dimensional (2D) kinetic model of the human body was used to analyze occupational lifting tasks. The model comprises seven body segments and ground as shown in Figure 2. Only right side segments are labeled. Following Freivalds (1984), the lumbar region, torso, and head are lumped as a single body segment. Similarly hands and forearms are modeled as single body segments. Figure 3 shows the right side body segments with local reference frames connected to the mass center of each body segment.

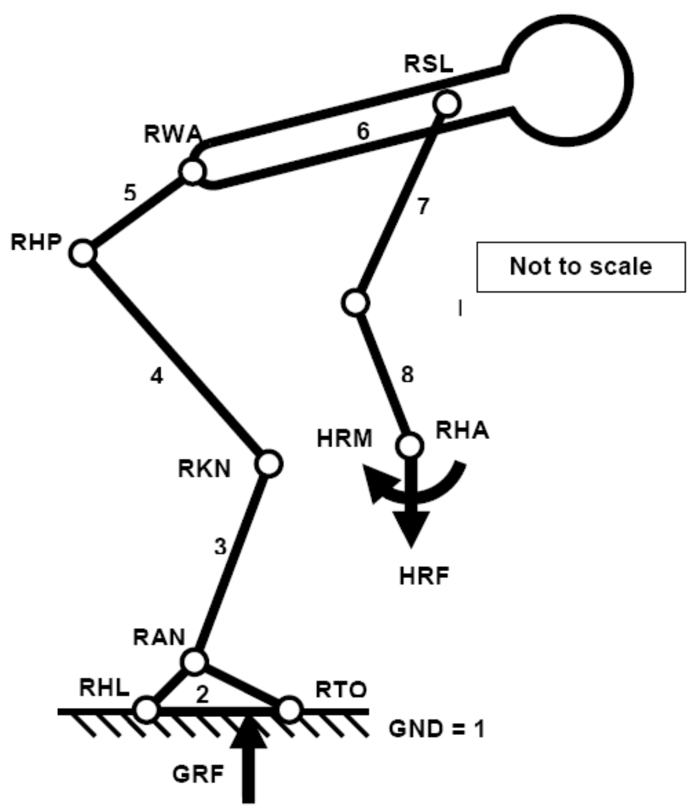

Figure 2: Two-dimensional biomechanical model of occupational lifting

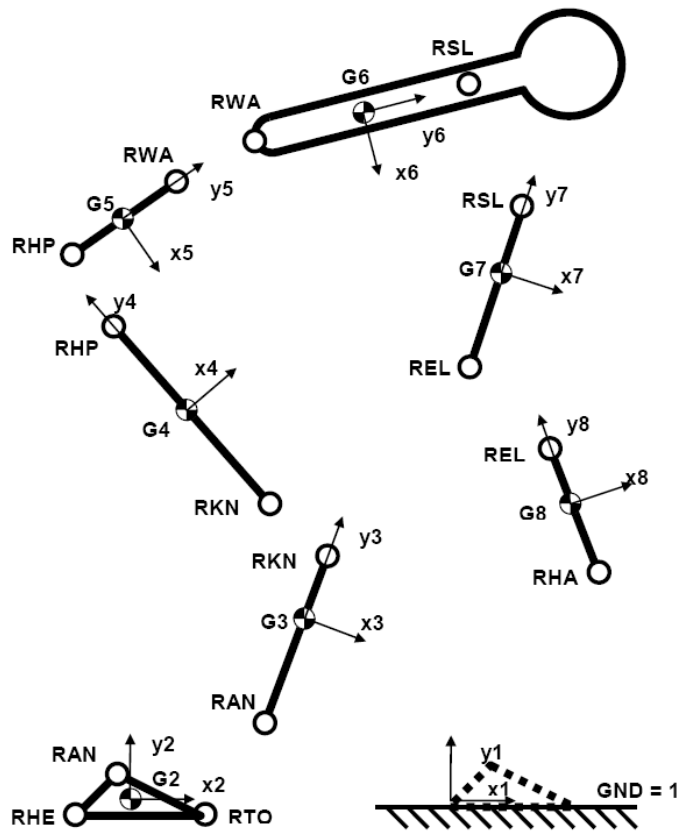

Figure 3: Local coordinate frames at centroids of body segments (x anterior, y proximal/superior)

The following assumptions are used in this model: the motions of contralateral limbs are symmetric and the payload is supported equally by both hands. Both feet are flat on the ground, are parallel and do not move. The ground reaction force (GRF) is exerted by the ground on the feet at the midpoint between the feet. The hand reaction force (HRF) and hand reaction moment (HRM) are exerted by the payload onto the hands at the midpoint between the hands. Table 1Table 3 lists all of the body segments, joints, and external objects and forces used in the biomechanical model.

Table 3: List of body segments, joints, and external objects, forces, and moments

Body Segments			Points		External Objects, Forces, Moments	
No.	Name	Description	Name	Description	Name	Descrition
2	RFT	foot (fixed to ground)	RHL	heel	1 (GND)	ground
3	RSK	shank	RTO	toe	GRF	ground reaction force
4	RTH	thigh	RAN	ankle	HRF	hand reaction force
5	RPV	pelvis	RKN	knee	HRM	hand reaction moment
6	RTA	head-thorax-abdomen	RHP	hip		
7	RAR	arm	RWA	waist		
8	RFA	forearm	RSL	shoulder		
			REL	elbow		
			RHA	hand		

3.3. Kinematic model

This section details the algorithm used in the kinematic model for the PBAS. This algorithm was developed following the methods of Haug (1989). The vector and matrix notation is explained in Appendix B.

1) Standing Calibration

The body segment parameters (BSPs) for each test subject were determined from standing calibration and the equations as described in section 3.1. The constant segment lengths and constant centroid location for each body segment relative to the endpoints based on joint locations were also measured during the standing calibration.

2) Determine the constant joint locations $\left\{{}^{Gi}s_{JNTi/Gi}\right\}$ relative to centroidal coordinate frames

$\left\{{}^{G2}s_{RHL2/G2}\right\}$ $\left\{{}^{G2}s_{RTO2/G2}\right\}$ $\left\{{}^{G2}s_{RAN2/G2}\right\}$

$\left\{{}^{G3}s_{RAN3/G3}\right\}$ $\left\{{}^{G3}s_{RKN3/G3}\right\}$

$\left\{{}^{G4}s_{RKN4/G4}\right\}$ $\left\{{}^{G4}s_{RHP4/G4}\right\}$

$\left\{{}^{G5}s_{RHP5/G5}\right\}$ $\left\{{}^{G5}s_{RWA5/G5}\right\}$

$\left\{{}^{G6}s_{RWA6/G6}\right\}$ $\left\{{}^{G6}s_{RSL6/G6}\right\}$

$\left\{{}^{G7}s_{RSL7/G7}\right\}$ $\left\{{}^{G7}s_{REL7/G7}\right\}$

$\left\{{}^{G8}s_{REL8/G08}\right\}$ $\left\{{}^{G8}s_{RHA8/G8}\right\}$

The distance between each joint and the mass centers of each segment will be determined using standing calibration by measuring the distances on a photograph of the subject.

3) Determine constant IMU locations $\left\{{}^{Gi}s_{IMUi/Gi}\right\}$ and attitude angle offsets ${}^{Gi}\theta_{IMUi}$ relative to centroidal coordinate frames based on IMU locations and sensor readings measured during initial standing calibration

$\left\{{}^{G2}s_{IMU2/G2}\right\}$ ${}^{G2}\theta_{IMU2}$

$\left\{{}^{G3}s_{IMU3/G3}\right\}$ ${}^{G3}\theta_{IMU3}$

$\left\{{}^{G4}s_{IMU4/G4}\right\}$ ${}^{G4}\theta_{IMU4}$

$\left\{{}^{G5}s_{IMU5/G5}\right\}$ ${}^{G5}\theta_{IMU5}$

$\{^{G6}s_{IMU6/G6}\}$ $\quad$ $^{G6}\theta_{IMU6}$

$\{^{G7}s_{IMU7/G7}\}$ $\quad$ $^{G7}\theta_{IMU7}$

$\{^{G8}s_{IMU8/G8}\}$ $\quad$ $^{G8}\theta_{IMU8}$

The distances from the IMUs to their corresponding mass centers will be measured on a photo of the subject. The angle offsets will be measured by using the accelerometers on each IMU and are in local directions. This is shown in Equation 7.

Equation 7: Angle offset of IMU i

$$^{Gi}\theta_{IMUi} = \tan\left(\frac{a_{xi}}{a_{yi}}\right)$$

4) Determine constant mass m_i for each segment based on body weight and segment lengths measured during initial standing calibration

$m_2 \quad m_3 \quad m_4 \quad m_5 \quad m_6 \quad m_7 \quad m_8$

5) Determine constant centroidal mass moment of inertia J_{Gi} for each segment based on body weight and segment lengths measured during initial standing calibration

$J_{G2} \quad J_{G3} \quad J_{G4} \quad J_{G5} \quad J_{G6} \quad J_{G7} \quad J_{G8}$

6) Use the following generalized position coordinates

$$\underset{14x1}{\{q_r\}} = \begin{Bmatrix} \{r_{G2}\} \\ \{r_{G3}\} \\ \{r_{G4}\} \\ \{r_{G5}\} \\ \{r_{G6}\} \\ \{r_{G7}\} \\ \{r_{G8}\} \end{Bmatrix} \qquad \underset{7x1}{\{q_\theta\}} = \begin{Bmatrix} \theta_{G2} \\ \theta_{G3} \\ \theta_{G4} \\ \theta_{G5} \\ \theta_{G6} \\ \theta_{G7} \\ \theta_{G8} \end{Bmatrix}$$

The generalized position vector is a vector of the global x- and y-locations of the mass centers of each body segment. The generalized attitude vector is a vector of the global attitudes of each body segment.

7) Attitude solution

7.1) Experimentally measure attitude θ_{IMUi} of IMUs with respect to gravity

7.2) Compute attitude angles θ_{Gi} for segments and complete $\{q_\theta\}$

$$\theta_{Gi} = \theta_{IMUi} - {}^{Gi}\theta_{IMUi}$$

The global attitude angles of each body segment are calculated by subtracting the initial offset angle of the IMU from the measured IMU angle. The initial offset angle comes from the standing calibration and is the angle at which the IMU is placed on the subject. The measured angle is calculated by obtaining the measured accelerations from the IMU and using Equation 7.

7.3) Compute attitude matrices for body segments and IMUs

$$[A_{Gi}] = \begin{bmatrix} \cos\theta_{Gi} & -\sin\theta_{Gi} \\ \sin\theta_{Gi} & \cos\theta_{Gi} \end{bmatrix}$$
$$[A_{IMUi}] = \begin{bmatrix} \cos\theta_{IMUi} & -\sin\theta_{IMUi} \\ \sin\theta_{IMUi} & \cos\theta_{IMUi} \end{bmatrix}$$

The attitude matrices will be used to transform distances measured in the local segment reference frames into global measurements.

7.4) Compute orthogonal attitude matrices for segments

$$[B_{Gi}] = [R][A_{Gi}] \qquad \text{for} \qquad [R] = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

The B matrices will be used in calculating Jacobians and the velocity and acceleration solutions as it is related to the derivative of the attitude matrices.

8) Position solution

8.1) Compute global locations of segments $\{q_r\}$ using the global origin at the heels

$$\underset{14\text{x}1}{\{q_r\}} = \begin{Bmatrix} \{r_{G2}\} \\ \{r_{G3}\} \\ \{r_{G4}\} \\ \{r_{G5}\} \\ \{r_{G6}\} \\ \{r_{G7}\} \\ \{r_{G8}\} \end{Bmatrix} = \begin{Bmatrix} -[A_{G2}]\{{}^{G2}s_{RHL2/G2}\} \\ \{r_{G2}\} + [A_{G2}]\{{}^{G2}s_{RAN2/G2}\} - [A_{G3}]\{{}^{G3}s_{RAN3/G3}\} \\ \{r_{G3}\} + [A_{G3}]\{{}^{G3}s_{RKN3/G3}\} - [A_{G4}]\{{}^{G4}s_{RKN4/G4}\} \\ \{r_{G4}\} + [A_{G4}]\{{}^{G4}s_{RHP4/G4}\} - [A_{G5}]\{{}^{G5}s_{RHP5/G5}\} \\ \{r_{G5}\} + [A_{G5}]\{{}^{G5}s_{RWA5/G5}\} - [A_{G6}]\{{}^{G6}s_{RWA6/G6}\} \\ \{r_{G6}\} + [A_{G6}]\{{}^{G6}s_{RSL6/G6}\} - [A_{G7}]\{{}^{G7}s_{RSL7/G7}\} \\ \{r_{G7}\} + [A_{G7}]\{{}^{G7}s_{REL7/G7}\} - [A_{G8}]\{{}^{G3}s_{REL8/G8}\} \end{Bmatrix}$$

The global locations of each mass center are found by adding the distance from the previous mass center to the next joint, then adding the distance from the joint to the next mass center. This can be done because the body as described in this model is a continuous open chain mechanism.

8.2) Compute global locations of joints on each segment

$$\{r_{JNTi}\} = \{r_{Gi}\} + [A_{Gi}]\{^{Gi}s_{JNTi/Gi}\}$$

These are computed in a similar way as the global mass centers. The global joint locations are found by converting the distance from the mass center to the joint in local directions into global directions and adding the position of the mass center in global directions.

8.3) Compute global locations of IMUs

$$\{r_{IMUi}\} = \{r_{Gi}\} + [A_{Gi}]\{^{Gi}s_{IMUi/Gi}\}$$

These are computed in a similar way as the global mass centers and joint locations.

9) Check position solution

9.1) Check revolute constraints at heel and joints that should be zero

$$\underset{14x1}{\{\Phi\}_{REV}} = \begin{Bmatrix} \{\Phi\}_{HEL} \\ \{\Phi\}_{ANK} \\ \{\Phi\}_{KNE} \\ \{\Phi\}_{HIP} \\ \{\Phi\}_{WAI} \\ \{\Phi\}_{SHL} \\ \{\Phi\}_{ELB} \end{Bmatrix} = \begin{Bmatrix} \{r_{RHL2}\} - \{r_{RHL1}\} \\ \{r_{RAN3}\} - \{r_{RAN2}\} \\ \{r_{RKN4}\} - \{r_{RKN3}\} \\ \{r_{RHP5}\} - \{r_{RHP4}\} \\ \{r_{RWA6}\} - \{r_{RWA5}\} \\ \{r_{RSL7}\} - \{r_{RSL6}\} \\ \{r_{REL8}\} - \{r_{REL7}\} \end{Bmatrix} = ?\ \{0_{14x1}\}$$

These conditions are used to check the position solution since the global location of corresponding joints on different body segments should be the same assuming the joints can be modeled as revolute constraints. This is a good assumption for healthy joints.

9.2) Check attitude angle constraints that should be zero

$$\underset{7\text{x}1}{\{\Phi\}_{\text{ATT}}} = \begin{Bmatrix} \theta_{G2} - \theta_{IMU2} + {}^{G2}\theta_{IMU2} \\ \theta_{G3} - \theta_{IMU3} + {}^{G3}\theta_{IMU3} \\ \theta_{G4} - \theta_{IMU4} + {}^{G4}\theta_{IMU4} \\ \theta_{G5} - \theta_{IMU5} + {}^{G5}\theta_{IMU5} \\ \theta_{G6} - \theta_{IMU6} + {}^{G6}\theta_{IMU6} \\ \theta_{G7} - \theta_{IMU7} + {}^{G6}\theta_{IMU7} \\ \theta_{G8} - \theta_{IMU8} + {}^{G6}\theta_{IMU8} \end{Bmatrix} = ?\ \{0_{7\text{x}1}\}$$

The attitude angle constraints check to see that the angles are measured correctly. These equations check that the angle of the body segment matches the measured angle during the Newton-Raphson algorithm used to obtain the position solution.

10) Jacobians

The Jacobians calculated here will be used later in the velocity and acceleration solutions.

10.1) Compute Jacobian of revolute constraints with respect to segment locations

$$\underset{14\text{x}14}{\left[\partial\Phi_{\text{REV}}/\partial q_r\right]} = \begin{bmatrix} [I_2] & [0_{2\text{x}2}] & [0_{2\text{x}2}] & & \cdots & & [0_{2\text{x}2}] \\ -[I_2] & [I_2] & [0_{2\text{x}2}] & & & & \\ [0_{2\text{x}2}] & -[I_2] & [I_2] & & & & \vdots \\ & & -[I_2] & [I_2] & & & \\ \vdots & & & -[I_2] & [I_2] & [0_{2\text{x}2}] & [0_{2\text{x}2}] \\ & & & & -[I_2] & [I_2] & [0_{2\text{x}2}] \\ [0_{2\text{x}2}] & & \cdots & & [0_{2\text{x}2}] & -[I_2] & [I_2] \end{bmatrix}$$

This Jacobian is the derivative of the revolute constraint vector with respect to the generalized position vector.

10.2) Compute Jacobian of attitude constraints with respect to segment locations

$$\underset{7\text{x}14}{\left[\partial\Phi_{\text{ATT}}/\partial q_r\right]} = \left[0_{7\text{x}14}\right]$$

This Jacobian is the derivative of the attitude constraint vector with respect to the generalized position vector. It is all zeros because the attitude constraints have only angles in them and no positions.

10.3) Compute Jacobian of revolute constraints with respect to segment attitudes

$$
\left[\partial\Phi_{\underset{14x7}{REV}}/\partial q_{\theta}\right]=\begin{bmatrix}
\left[B_{G2}\right]\left\{{}^{G2}s_{RHL2/G2}\right\} & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \\
-\left[B_{G2}\right]\left\{{}^{G2}s_{RAN2/G2}\right\} & \left[B_{G3}\right]\left\{{}^{G3}s_{RAN3/G3}\right\} & \left[0_{2x1}\right] & \\
\left[0_{2x1}\right] & -\left[B_{G3}\right]\left\{{}^{G3}s_{RKN3/G3}\right\} & \left[B_{G4}\right]\left\{{}^{G4}s_{RKN4/G4}\right\} & \\
\left[0_{2x1}\right] & \left[0_{2x1}\right] & -\left[B_{G4}\right]\left\{{}^{G4}s_{RHP4/G4}\right\} & \cdots \\
\left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \\
\left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \\
\left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] &
\end{bmatrix}
$$

$$
\begin{bmatrix}
 & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] \\
 & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] \\
 & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] \\
\cdots & \left[B_{G5}\right]\left\{{}^{G4}s_{RHP5/G5}\right\} & \left[0_{2x1}\right] & \left[0_{2x1}\right] & \left[0_{2x1}\right] \\
 & -\left[B_{G5}\right]\left\{{}^{G5}s_{RWA5/G5}\right\} & \left[B_{G6}\right]\left\{{}^{G6}s_{RWA6/G6}\right\} & \left[0_{2x1}\right] & \left[0_{2x1}\right] \\
 & \left[0_{2x1}\right] & -\left[B_{G6}\right]\left\{{}^{G6}s_{RSL6/G6}\right\} & \left[B_{G7}\right]\left\{{}^{G7}s_{RSL7/G7}\right\} & \left[0_{2x1}\right] \\
 & \left[0_{2x1}\right] & \left[0_{2x1}\right] & -\left[B_{G7}\right]\left\{{}^{G7}s_{REL7/G7}\right\} & \left[B_{G8}\right]\left\{{}^{G8}s_{REL8/G8}\right\}
\end{bmatrix}
$$

This Jacobian is the derivative of the revolute constraint vector with respect to the generalized attitude vector.

10.4) Compute Jacobian of attitude constraints with respect to segment attitudes

$$
\left[\partial\Phi_{\underset{7x7}{ATT}}/\partial q_{\theta}\right]=\left[I_7\right]
$$

This Jacobian is the derivative of the attitude constraint vector with respect to the generalized attitude vector. It is the identity matrix because each attitude constraint is a function of the corresponding body segment attitude.

11) Use the following generalized velocity coordinates

$$
\underset{14x1}{\left\{\dot{q}_r\right\}}=\begin{Bmatrix}\left\{\dot{r}_{G2}\right\}\\ \left\{\dot{r}_{G3}\right\}\\ \left\{\dot{r}_{G4}\right\}\\ \left\{\dot{r}_{G5}\right\}\\ \left\{\dot{r}_{G6}\right\}\\ \left\{\dot{r}_{G2}\right\}\\ \left\{\dot{r}_{G2}\right\}\end{Bmatrix}
\qquad
\underset{7x1}{\left\{\dot{q}_{\theta}\right\}}=\begin{Bmatrix}\dot{\theta}_{G2}\\ \dot{\theta}_{G3}\\ \dot{\theta}_{G4}\\ \dot{\theta}_{G5}\\ \dot{\theta}_{G6}\\ \dot{\theta}_{G7}\\ \dot{\theta}_{G8}\end{Bmatrix}
$$

12) Angular velocity solution

12.1) Experimentally measure angular velocities $\dot{\theta}_{IMUi}$ of IMUs

12.2) Compute angular velocities $\dot{\theta}_{Gi}$ for segments and complete $\{\dot{q}_{\theta}\}$

$$\dot{\theta}_{Gi} = \dot{\theta}_{IMUi}$$

The angular velocity of each body segment will be the same as the angular velocity of its corresponding IMU since they do not rotate relative to each other.

12.3) Compute time derivatives of attitude matrices for segments

$$[\dot{A}_{Gi}] = \dot{\theta}_{Gi}[B_{Gi}]$$

13) Translational velocity solution

13.1) Compute global velocities of segments $\{\dot{q}_r\}$ assuming heel does not move

$$\underset{14x1}{\{\dot{q}_r\}} = \begin{Bmatrix} \{\dot{r}_{G2}\} \\ \{\dot{r}_{G3}\} \\ \{\dot{r}_{G4}\} \\ \{\dot{r}_{G5}\} \\ \{\dot{r}_{G6}\} \\ \{\dot{r}_{G7}\} \\ \{\dot{r}_{G8}\} \end{Bmatrix} = \begin{Bmatrix} -[\dot{A}_{G2}]\{^{G2}s_{RHL2/G2}\} \\ \{\dot{r}_{G2}\} + [\dot{A}_{G2}]\{^{G2}s_{RAN2/G2}\} - [\dot{A}_{G3}]\{^{G3}s_{RAN3/G3}\} \\ \{\dot{r}_{G3}\} + [\dot{A}_{G3}]\{^{G3}s_{RKN3/G3}\} - [\dot{A}_{G4}]\{^{G4}s_{RKN4/G4}\} \\ \{\dot{r}_{G4}\} + [\dot{A}_{G4}]\{^{G4}s_{RHP4/G4}\} - [\dot{A}_{G5}]\{^{G5}s_{RHP5/G5}\} \\ \{\dot{r}_{G5}\} + [\dot{A}_{G5}]\{^{G5}s_{RWA5/G5}\} - [\dot{A}_{G6}]\{^{G6}s_{RWA6/G6}\} \\ \{\dot{r}_{G6}\} + [\dot{A}_{G6}]\{^{G6}s_{RSL6/G6}\} - [\dot{A}_{G7}]\{^{G7}s_{RSL7/G7}\} \\ \{\dot{r}_{G7}\} + [\dot{A}_{G7}]\{^{G7}s_{REL7/G7}\} - [\dot{A}_{G8}]\{^{G3}s_{REL8/G8}\} \end{Bmatrix}$$

This vector is found by taking the time derivative of the generalized position vector.

13.2) Compute global velocities of IMUs

$$\{\dot{r}_{IMUi}\} = \{\dot{r}_{Gi}\} + [\dot{A}_{Gi}]\{^{Gi}s_{IMUi/Gi}\}$$

The global IMU velocities are found by taking the time derivative of the global IMU positions.

14) Check velocity solution

14.1) General velocity solution

$$\underset{21x21}{\begin{bmatrix} [\partial\Phi_{REV}/\partial q_r] & [\partial\Phi_{REV}/\partial q_{\theta}] \\ [\partial\Phi_{ATT}/\partial q_r] & [\partial\Phi_{ATT}/\partial q_{\theta}] \end{bmatrix}} \underset{21x1}{\begin{Bmatrix} \{\dot{q}_r\} \\ \{\dot{q}_{\theta}\} \end{Bmatrix}} = \begin{Bmatrix} \{\nu\}_{REV} \\ \{\nu\}_{ATT} \end{Bmatrix} = \begin{Bmatrix} \{0_{14x1}\} \\ \{\dot{\theta}_{IMUi}\} \end{Bmatrix}$$

14.2) Use revolute equations to check matrix solution for global velocities of segments against direct solution in Equation 13.1

$$\{\dot{q}_r\} = ? - [\partial\Phi_{REV}/\partial q_r]^{-1}([\partial\Phi_{REV}/\partial q_\theta]\{\dot{q}_\theta\})$$

15) Use the following generalized acceleration coordinates

$$\underset{14x1}{\{\ddot{q}_r\}} = \begin{Bmatrix} \{\ddot{r}_{G2}\} \\ \{\ddot{r}_{G3}\} \\ \{\ddot{r}_{G4}\} \\ \{\ddot{r}_{G5}\} \\ \{\ddot{r}_{G6}\} \\ \{\ddot{r}_{G2}\} \\ \{\ddot{r}_{G2}\} \end{Bmatrix} \qquad \underset{7x1}{\{\ddot{q}_\theta\}} = \begin{Bmatrix} \ddot{\theta}_{G2} \\ \ddot{\theta}_{G3} \\ \ddot{\theta}_{G4} \\ \ddot{\theta}_{G5} \\ \ddot{\theta}_{G6} \\ \ddot{\theta}_{G7} \\ \ddot{\theta}_{G8} \end{Bmatrix}$$

16) Rotational acceleration solution

16.1) Experimentally measure biased translational accelerations $\{^{IMUi}\ddot{r}_{IMUi}\}_{BIASED}$ of IMUs relative to local IMU coordinate frames

The biased translational accelerations include the constant acceleration of gravity.

16.2) Transform biased translational accelerations of IMUs into global directions and remove gravity bias

$$\{\ddot{r}_{IMUi}\} = [A_{IMUi}]\{^{IMUi}\ddot{r}_{IMUi}\}_{BIASED} - \begin{Bmatrix} 0 \\ 1\,G \end{Bmatrix}$$

16.3) Acceleration of each IMU using acceleration of each segment

$$\{\ddot{r}_{IMUi}\} = \{\ddot{r}_{Gi}\} + [\ddot{A}_{Gi}]\{^{Gi}s_{IMUi/Gi}\} = \{\ddot{r}_{Gi}\} + \ddot{\theta}_{Gi}[B_{Gi}]\{^{Gi}s_{IMUi/Gi}\} - \dot{\theta}_{Gi}^2[A_{Gi}]\{^{Gi}s_{IMUi/Gi}\}$$

The acceleration of each IMU consists of the translational acceleration of the mass center (first term) plus the rotational acceleration of the IMU about the mass center. The rotational acceleration consists of a tangential acceleration (second term) and a normal acceleration (third term).

16.4) Acceleration of each segment using acceleration of each IMU found by rearranging Equation 16.3

$$\{\ddot{r}_{Gi}\} = -\ddot{\theta}_{Gi}[B_{Gi}]\{^{Gi}s_{IMUi/Gi}\} + \{\ddot{r}_{IMUi}\} + \dot{\theta}_{Gi}^2[A_{Gi}]\{^{Gi}s_{IMUi/Gi}\}$$

16.5) Generalized translation accelerations in terms of generalized rotational accelerations

$$\underset{14x1}{\{\ddot{q}_r\}} = -\underset{14x7}{[C]}\underset{7x1}{\{\ddot{q}_\theta\}} + \underset{14x1}{\{D\}}$$

$$
\underset{14x7}{[C]} = \begin{bmatrix} [B_{G2}]\{{}^{G2}s_{IMU2/G2}\} & [0_{2x1}] & [0_{2x1}] & \\ [0_{2x1}] & [B_{G3}]\{{}^{G3}s_{IMU3/G3}\} & [0_{2x1}] & \\ [0_{2x1}] & [0_{2x1}] & [B_{G4}]\{{}^{G4}s_{IMU4/G4}\} & \\ [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & \cdots \\ [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & \\ [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & \\ [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & \end{bmatrix}
$$

$$
\begin{bmatrix} & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] \\ & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] \\ & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] \\ \cdots & [B_{G5}]\{{}^{G5}s_{IMU5/G5}\} & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] \\ & [0_{2x1}] & [B_{G6}]\{{}^{G6}s_{IMU6/G6}\} & [0_{2x1}] & [0_{2x1}] \\ & [0_{2x1}] & [0_{2x1}] & [B_{G7}]\{{}^{G7}s_{IMU7/G7}\} & [0_{2x1}] \\ & [0_{2x1}] & [0_{2x1}] & [0_{2x1}] & [B_{G8}]\{{}^{G8}s_{IMU8/G8}\} \end{bmatrix}
$$

$$
\underset{14x1}{\{D\}} = \begin{Bmatrix} \{\ddot{r}_{IMU2}\} \\ \{\ddot{r}_{IMU3}\} \\ \{\ddot{r}_{IMU4}\} \\ \{\ddot{r}_{IMU5}\} \\ \{\ddot{r}_{IMU6}\} \\ \{\ddot{r}_{IMU7}\} \\ \{\ddot{r}_{IMU8}\} \end{Bmatrix} + \begin{Bmatrix} \dot{\theta}_{G2}^2 [A_{G2}]\{{}^{G2}s_{IMU2/G2}\} \\ \dot{\theta}_{G3}^2 [A_{G3}]\{{}^{G3}s_{IMU3/G3}\} \\ \dot{\theta}_{G4}^2 [A_{G4}]\{{}^{G4}s_{IMU4/G4}\} \\ \dot{\theta}_{G5}^2 [A_{G5}]\{{}^{G5}s_{IMU5/G5}\} \\ \dot{\theta}_{G6}^2 [A_{G6}]\{{}^{G6}s_{IMU6/G6}\} \\ \dot{\theta}_{G7}^2 [A_{G7}]\{{}^{G7}s_{IMU7/G7}\} \\ \dot{\theta}_{G8}^2 [A_{G8}]\{{}^{G8}s_{IMU8/G8}\} \end{Bmatrix}
$$

16.6) General acceleration solution

$$
\underset{21x21}{\begin{bmatrix} [\partial\Phi_{REV}/\partial q_r] & [\partial\Phi_{REV}/\partial q_\theta] \\ [\partial\Phi_{ATT}/\partial q_r] & [\partial\Phi_{ATT}/\partial q_\theta] \end{bmatrix}} \underset{21x1}{\begin{Bmatrix} \{\ddot{q}_r\} \\ \{\ddot{q}_\theta\} \end{Bmatrix}} = \underset{21x1}{\begin{Bmatrix} \{\gamma\}_{REV} \\ \{\gamma\}_{ATT} \end{Bmatrix}}
$$

$$
\{\gamma\}_{REV\_j\_i} = \dot{\theta}_{Gj}^2 [A_{Gj}]\{{}^{Gj}s_{JNTj/Gj}\} - \dot{\theta}_{Gi}^2 [A_{Gi}]\{{}^{Gi}s_{JNTi/Gi}\} \qquad \text{for} \qquad \{\Phi\}_{REV\_j\_i} = \{r_{JNTj}\} - \{r_{JNTi}\}
$$

16.7) Revolute equations using generalized translational and rotational accelerations

$$
\underset{14X14}{[\partial\Phi_{REV}/\partial q_r]} \underset{14X1}{\{\ddot{q}_r\}} + \underset{14X7}{[\partial\Phi_{REV}/\partial q_\theta]} \underset{7X1}{\{\ddot{q}_\theta\}} = \underset{14X1}{\{\gamma\}_{REV}}
$$

16.8) Revolute equations using only generalized rotational accelerations

$$\underset{14X14}{\left[\partial\Phi_{REV}/\partial q_r\right]}\left(-\underset{14x7}{\left[C\right]}\underset{7x1}{\left\{\ddot{q}_\theta\right\}}+\underset{14x1}{\left\{D\right\}}\right)+\underset{14X7}{\left[\partial\Phi_{REV}/\partial q_\theta\right]}\underset{7X1}{\left\{\ddot{q}_\theta\right\}}=\underset{14X1}{\left\{\gamma\right\}_{REV}}$$

16.9) Over-determined set of equations for generalized rotational accelerations

$$\underset{14X7}{\left[E\right]}\underset{7X1}{\left\{\ddot{q}_\theta\right\}}=\underset{14X1}{\left\{\gamma\right\}_{REV}}-\underset{14X14}{\left[\partial\Phi_{REV}/\partial q_r\right]}\underset{14x1}{\left\{D\right\}}$$

$$\underset{14X7}{\left[E\right]}=\underset{14X7}{\left[\partial\Phi_{REV}/\partial q_\theta\right]}-\underset{14X14}{\left[\partial\Phi_{REV}/\partial q_r\right]}\underset{14x7}{\left[C\right]}$$

16.10) Least-squares solution for generalized rotational accelerations

$$\underset{7X1}{\left\{\ddot{q}_\theta\right\}}=\left(\underset{7X14}{\left[E\right]^T}\underset{14X7}{\left[E\right]}\right)^{-1}\left(\underset{7X14}{\left[E\right]^T}\left(\underset{14X1}{\left\{\gamma\right\}_{REV}}-\underset{14X14}{\left[\partial\Phi_{REV}/\partial q_r\right]}\underset{14x1}{\left\{D\right\}}\right)\right)$$

16.11) Compute second time derivatives of attitude matrices for segments

$$\left[\ddot{A}_{Gi}\right]=\ddot{\theta}_{Gi}\left[B_{Gi}\right]-\dot{\theta}_{Gi}^2\left[A_{Gi}\right]$$

17) Translational acceleration solution

17.1) Compute global accelerations of segments $\left\{\ddot{q}_r\right\}$

$$\underset{14x1}{\left\{\ddot{q}_r\right\}}=\begin{Bmatrix}\{\ddot{r}_{G2}\}\\\{\ddot{r}_{G3}\}\\\{\ddot{r}_{G4}\}\\\{\ddot{r}_{G5}\}\\\{\ddot{r}_{G6}\}\\\{\ddot{r}_{G7}\}\\\{\ddot{r}_{G8}\}\end{Bmatrix}=\begin{Bmatrix}-\left[\ddot{A}_{G2}\right]\left\{{}^{G2}s_{RHL2/G2}\right\}\\\{\ddot{r}_{G2}\}+\left[\ddot{A}_{G2}\right]\left\{{}^{G2}s_{RAN2/G2}\right\}-\left[\ddot{A}_{G3}\right]\left\{{}^{G3}s_{RAN3/G3}\right\}\\\{\ddot{r}_{G3}\}+\left[\ddot{A}_{G3}\right]\left\{{}^{G3}s_{RKN3/G3}\right\}-\left[\ddot{A}_{G4}\right]\left\{{}^{G4}s_{RKN4/G4}\right\}\\\{\ddot{r}_{G4}\}+\left[\ddot{A}_{G4}\right]\left\{{}^{G4}s_{RHP4/G4}\right\}-\left[\ddot{A}_{G5}\right]\left\{{}^{G5}s_{RHP5/G5}\right\}\\\{\ddot{r}_{G5}\}+\left[\ddot{A}_{G5}\right]\left\{{}^{G5}s_{RWA5/G5}\right\}-\left[\ddot{A}_{G6}\right]\left\{{}^{G6}s_{RWA6/G6}\right\}\\\{\ddot{r}_{G6}\}+\left[\ddot{A}_{G6}\right]\left\{{}^{G6}s_{RSL6/G6}\right\}-\left[\ddot{A}_{G7}\right]\left\{{}^{G7}s_{RSL7/G7}\right\}\\\{\ddot{r}_{G7}\}+\left[\ddot{A}_{G7}\right]\left\{{}^{G7}s_{REL7/G7}\right\}-\left[\ddot{A}_{G8}\right]\left\{{}^{G3}s_{REL8/G8}\right\}\end{Bmatrix}$$

The global accelerations are found by taking the time derivative of the global velocities.

17.2) Compute global accelerations of IMUs and check against unbiased measurements

$$\left\{\ddot{r}_{IMUi}\right\}=?\ \left\{\ddot{r}_{Gi}\right\}+\left[\ddot{A}_{Gi}\right]\left\{{}^{Gi}s_{IMUi/Gi}\right\}$$

18) Calculate joint reaction forces and torques

$$\underbrace{\begin{bmatrix} [M] & [0_{14x7}] \\ [0_{7x14}] & [J_G] \end{bmatrix}}_{21x21} \underbrace{\begin{Bmatrix} \{\ddot{q}_r\} \\ \{\ddot{q}_\theta\} \end{Bmatrix}}_{21x1} + \underbrace{\begin{bmatrix} [\partial\Phi_{REV}/\partial q_r]^T & [\partial\Phi_{ATT}/\partial q_r]^T \\ [\partial\Phi_{REV}/\partial q_\theta]^T & [\partial\Phi_{ATT}/\partial q_\theta]^T \end{bmatrix}}_{21x21} \underbrace{\begin{Bmatrix} \{\lambda\}_{REV} \\ \{\lambda\}_{ATT} \end{Bmatrix}}_{21x1} = \underbrace{\begin{Bmatrix} \sum\{F\}_{EXT} \\ \sum\{T\}_{EXT} \end{Bmatrix}}_{21x1}$$

$\{\lambda\}_{REV}$ will be joint reaction forces measured in global directions
$\{\lambda\}_{ATT}$ will not be joint torques
$\{\lambda\}_{ATT}$ will be some measure of absolute torque because $\{\Phi\}_{ATT}$ is absolute attitude
$\{F\}_{EXT}$ and $\{T\}_{EXT}$ will only act on feet and hands and are known

4. Hardware and Data Collection

4.1. Introduction

Ten IMUs were constructed to take measurements on humans. The IMUs consist of an Xbee Radio module and a six degree of freedom Razor made by Sparkfun. The Xbee radio and Razor are described below. An interface board was designed to connect the Xbee to the Razor. Figure 4 shows the front, inside and back views of the finished IMUs.

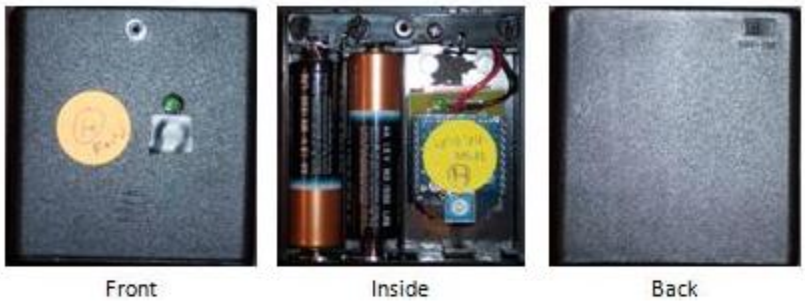

Figure 4: Final IMU

The IMUs begin sampling when they receive a command from a Pacer, which consists of an Xbee radio module and a PIC16F688 microcontroller. The Pacer is shown in Figure 5. The PIC is coded to send the start commands out of its USART (Universal Asynchronous Receiver Transmitter) to the Xbee when either the start button is pushed or the TTL trigger sees a high-to-low transition. These commands are addressed to each specific Xbee in the network.

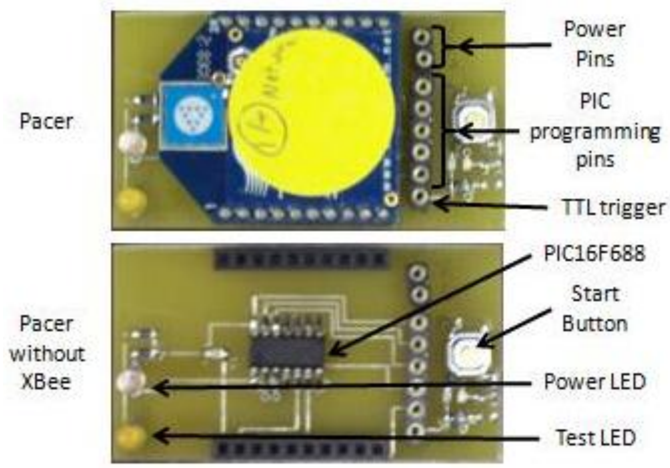

Figure 5: IMU pacer device

To collect the data, a PKG-U is connected to the USB port of a laptop. The PKG-U is the device provided by Digi to connect an Xbee to a computer. This device is used to program and communicate with an Xbee and is shown in Figure 6.

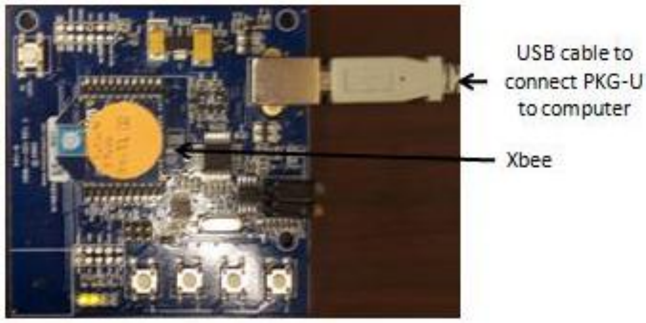

Figure 6: PKG-U

4.2. Electronics

4.2.1. Razor

The 6DOF Razor, shown in Figure 7, is made up of a three-axis accelerometer, a two-axis gyroscope to measure pitch and roll, and a single-axis gyroscope to measure yaw. The analog outputs from the three chips are broken out to two headers where they can be easily connected to the Xbee analog-to-digital converter (ADC) pins through the interface board. The Razor includes all of the filtering resistors and capacitors and requires a three volt power supply.

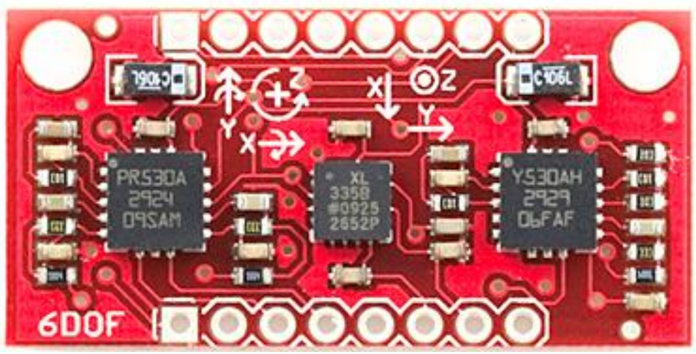

Figure 7: 6DOF Razor (Sparkfun Electronics 2010)

The Razor uses the ADXL335 accelerometer which measures accelerations within ±3 g. The chip has a user selected bandwidth by selecting the value of three capacitors. The bandwidth of the chip here is 50 Hz. For the PBAS, it is used to measure the angular position of the body segments by measuring the static acceleration of gravity in each direction.

The two-axis gyroscope used on the Razor is the LPR530AL manufactured by ST Microelectronics. It can measure ±300 °/s with positive being the counterclockwise direction, and is capable of sensing angular rates up to a bandwidth of 140 Hz. This chip has two outputs for each axis, a non-amplified and four times amplified signal. For the IMUs in the PBAS, the non-amplified signal is used. This chip comes in a land grid array (LGA) package with sixteen pins.

The single-axis gyroscope used on the Razor is the LY530ALH also made by ST Microelectronics. It can measure up to ±300 °/s with a bandwidth of 140 Hz. This chip also has two separate outputs with the non-amplified signal being used. The single-axis gyroscope comes in the same package as the two-axis gyroscope.

4.2.2. Xbee radio module

The PBAS uses the Xbee OEM RF Modules made by Digi for all radio communication. They use the 802.15.4 wireless radio protocol with a data rate of 250 kbps. The line-of-sight range for the modules is 300 ft with an operating frequency of 2.4 GHz. The modules in this application are programmed with a baud rate of 38400 Bd.

4.2.3. PIC16F688

The PIC16F688 is 14-pin, 8-bit microcontroller used on the pacer device of the PBAS. The PIC waits till it sees a high to low signal on one input pin and then sends the start sampling commands for all of the devices out of its USART to the Xbee radio for broadcasting. It is programmed to run at 38400 baud with an 8 MHz internal clock. This PIC allows for in circuit programming by connecting five pins to the computer through a set of headers on the pacer device.

4.3. Communication protocol

4.3.1. Networks

There are ten IMUs divided into two networks, with each network having its own Pacer and Collector. The networks are setup by programming the channel of each Xbee. IMUs A, B, C, D, and I form one network while IMUs E, F, G, H, and J form the other. Two networks are needed because of the amount of data coming into the collector from ten devices causes extra dropouts.

The networks are setup as peer-to-peer networks on two separate channels that have been programmed into the Xbees.

4.3.2. Addressing

Each Xbee is programmed with its own 64-bit and 16-bit source address. The 64-bit address is burned into the Xbee during manufacturing while the 16-bit is programmable. The ten 16-bit addresses are programmed with the hex number for the ASCII capital letters A through J.

When the IMU Xbees receive the start commands from pacer Xbee they are designed to send an acknowledgment and all data back to the pacer Xbee. In order for the collector to see the data transmissions, all Xbees are configured in broadcast mode.

4.3.3. Pacer device

The pacer device is made up of an Xbee radio module connected to a PIC16F688 microcontroller. This device is used to start the sampling of the IMUs by sending out remote AT commands. These commands are sent out of the PIC through its USART to the Xbee. This Xbee is programmed to be in API mode, which puts all data leaving or entering the Xbee into frames that define operations. The command for the IMUs to start sampling is sent to each IMU in the network. Each command has a destination address that specifies which Xbee is to receive the command. Figure 8 shows an example start command sent from the pacer device to IMU A.

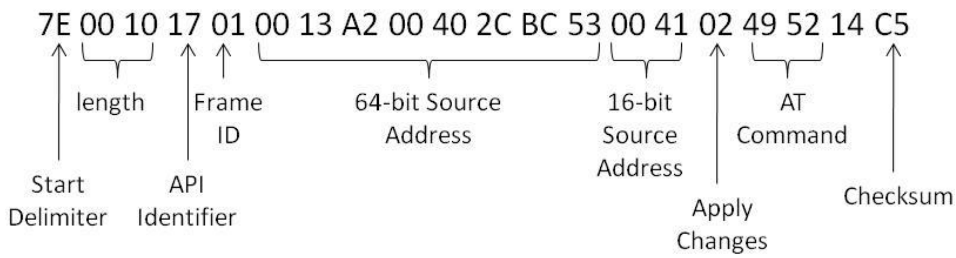

Figure 8: Remote AT command to start sampling of Xbee ADC lines

Once the start command is received by the Xbee on the IMUs, an acknowledgment is sent back to the pacer followed by the data. By design of the network, the acknowledgment and data are supposed to be sent back to the device that sent the command. To allow the collector to see everything, all Xbees in the networks are set to broadcast mode. Figure 9 shows the acknowledgment frame sent from IMU A

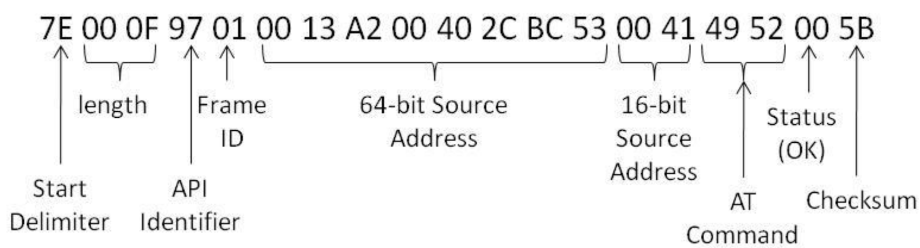

Figure 9: API acknowledgment frame

4.3.4. IMU

Each IMU consists of an Xbee radio connected to a 6DOF Razor. The six channels of the Razor are connected to the six ADC channels of the Xbee that convert the signals to send them to the collector device. Only three of the channels are turned on for each IMU, depending on if the IMU will be used in the sagittal plane or the coronal plane. The IMUs for the head and back will be in the coronal plane while all other IMUs will be in the sagittal plane. These two IMUs are placed parallel to the coronal plane as it will be more comfortable for the subjects to wear. Figure 10 shows the anatomical planes of the body.

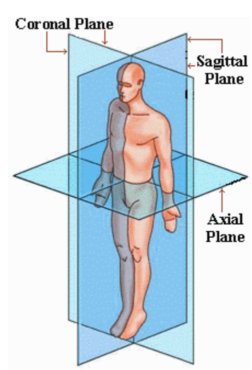

Figure 10: Body Planes (Bridwell 2010)

For the sagittal plane IMUs, the accelerations in x- and y-directions and the angular velocity about the z-axis are measured. To obtain the same directions of measurement, the coronal plane IMUs will measure the accelerations in the y- and z-directions and the angular velocity about the y-axis. The signals used for each IMU and there positive directions are shown in Figure 11.

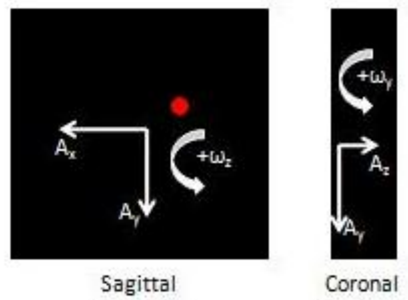

Figure 11: IMU signals in sagittal and coronal planes

The digital voltages measured by the Xbees are sent back to the collector in the corresponding API frame. The number of samples per packet is programmable and in this application is set to three. This was found to be the optimum number of samples per packet to minimize dropouts. Figure 12 shows an example data packet from IMU A.

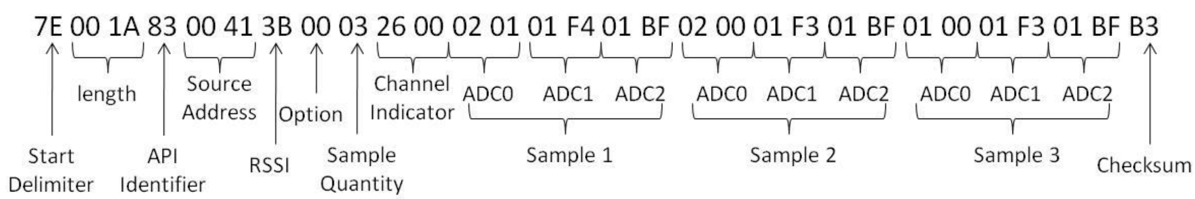

Figure 12: API data frame

4.4. Data collection

The data is received in packets as shown in Figure 12 by the PKG-U. This data is read into HyperTerminal and saved in a binary file. There are two sets of data (one for each network) with five IMUs in each data set. After data collection is complete, a Matlab program organizes the data into matrices for use.

4.5. Data organization

A Matlab program was created to combine the two data sets into four matrices, one for each signal and one for the calculated orientation angle of the IMUs. The code is shown in Appendices C-1, C-2 and C-3. The program calls a subroutine twice, once for each data set, and combines the data. The subroutine first reads in the data from the binary file. It then cuts off the acknowledgment packets at the beginning by looking for the acknowledgment from the last IMU and keeping all data after it. This provides a start of good data where all IMUs are running. After that, the program goes through every packet to make sure it is the correct size and places all good packets into a large packet matrix where each row corresponds to a packet and each column corresponds to a byte in the packet. Table 4 shows a sample of the packet matrix. For bad packets (too long or too short), it reads the IMU that it came from, puts its name in the correct columns of the matrix, and inserts NaN (not a real number) into the rest of the columns.

Table 4: Sample packet matrix

126	0	26	131	0	66	64	0	3	38	0	2	4	1	248	1	187	2	4	1	247	1	187	2	4	1	247	1	187	162
126	0	26	131	0	67	69	0	3	38	0	2	1	1	247	1	185	2	1	1	247	1	185	2	1	1	247	1	185	172
126	0	26	131	0	68	66	0	3	38	0	1	244	1	249	1	186	1	244	1	249	1	187	1	244	1	249	1	186	206
126	0	26	131	0	73	66	0	3	38	0	2	3	1	245	1	154	2	3	1	245	1	154	2	2	1	245	1	154	7

IMU Name

D0 D1 D2 D0 D1 D2 D0 D1 D2

Sample 1 Sample 2 Sample 3

After checking all packets for the proper structure, the program checks for dropped packets. The Xbees do not provide a counter or timer for data sent, so this is done by looking at five (or the number of IMUs in the network if less than 5) IMU names at a time and making sure all five IMUs are represented. If all five IMUs are in the group, there is no dropout. If one is missing, a dropout for that IMU is counted and NaN is inserted in the correct spot in the corresponding data vector. The dropped packets will be sent to a subroutine where they will be replaced by interpolated values based on the previous data.

A data vector is created for each signal along with a name vector that keeps track of which IMU the data belongs to. This is done by ripping the data out of the packet matrix and placing it in its corresponding data vector. The same is done for the IMU names. These vectors are shown in Table 5.

Table 5: Sample IMU name and channel vectors

IMU Name	Channel 0	Channel 1	Channel 2	
65 (A)	513	501	448	Sample 1
65	513	501	449	Sample 2
65	513	501	449	Sample 3
66 (B)	516	504	443	
66	516	503	443	
66	516	503	443	
67 (C)	513	503	441	
67	513	503	441	
67	513	503	441	
68 (D)	500	505	442	
68	500	505	443	
68	500	505	442	
73 (I)	515	501	410	
73	515	501	410	
73	514	501	410	

Finally, the program takes the data from the vectors and places it into the final data matrices. There are three matrices, one for each signal, with each column corresponding to one IMU. The data is inserted chronologically where it is sent back to the main program to be combined with the data from the second set. Table 6 shows a sample of the final data matrices. The data is now ready for analysis. The fourth data matrix is for the calculated orientation angle of the IMUs and is created by taking the arctangent of each entry in the acceleration matrices

Table 6: Sample final data matrices

Channel 0					Channel 1					Channel 2				
513	516	513	500	515	501	504	503	505	501	448	443	441	442	410
513	516	513	500	515	501	503	503	505	501	449	443	441	443	410
513	516	513	500	514	501	503	503	505	501	449	443	441	442	410
513	516	513	500	515	501	503	503	506	501	449	443	441	442	410
513	516	513	500	515	501	503	503	505	501	449	443	441	442	410
513	516	513	500	515	501	503	503	505	501	449	443	441	443	410
513	516	512	500	515	500	503	503	505	501	449	443	441	442	410
513	516	513	500	514	500	503	503	505	501	449	443	441	442	410
513	516	513	500	515	501	503	503	505	501	449	443	441	442	410
513	516	513	NaN	515	501	503	503	NaN	501	449	443	441	NaN	410
513	516	513	NaN	515	501	503	503	NaN	501	449	443	441	NaN	410
513	516	513	NaN	515	501	503	503	NaN	501	449	443	441	NaN	410
A	B	C	D	I										

Dropped Packet

4.6. Calibration

Prior to testing, the IMU signals must be calibrated to find the zero offset voltage for each sensor. This is done by first laying the IMUs flat with the z-axis pointing up as shown in Figure 13a. The IMUs are then flipped over so the z-axis is pointing down as shown in Figure 13b. In both orientations shown, the x- and y-acceleration and all angular velocity signals will be at their zero voltage. The z-acceleration will read -1g in the first position and +1g in the second. Data is collected and organized for both positions in a similar fashion as described in section 4.5. The voltages for the two calibration tests are then averaged together to give the zero voltage for all signals. The measured zero voltage is subtracted from the nominal zero voltage to give calibration voltages for all signals, which are then used in data processing for actual testing. The code for calibrating the IMUs is shown in Appendices C-4 and C-5.

Figure 13: Two calibration positions

5. Results

5.1. Battery test

An initial test was run with four IMUs to determine their behavior over time. The goal was to determine if the sensor measurements drift with time and if the temperature of the IMUs and amount of dropped packets change over time. First, the battery voltage and temperature of the IMUs were measured. The temperature was measured with a thermometer in the middle of the IMUs. Figure 14 shows the experimental setup of the battery test with the IMUs placed flat on a table. The temperature measured during this experiment is the outside temperature of the IMUs due to them being powered for an extended period. This was done to determine how it would affect the test subject during lift testing.

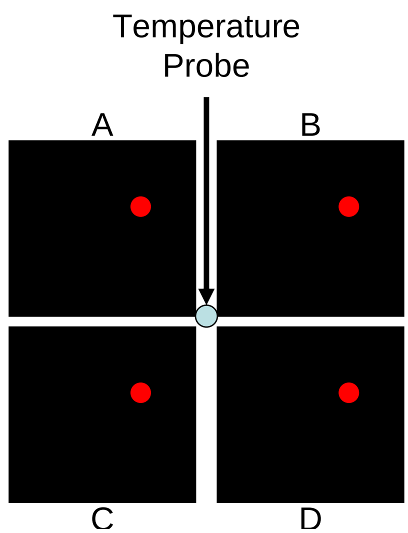

Figure 14: Experimental Setup

After the temperature and battery voltage was recorded, the four IMUs were started and sampled for approximately 30 seconds. Each of the three signals were averaged and recorded along with the percentage of dropped packets. The IMUs continued running for 15 minutes and were then reset and the procedure was repeated. The test lasted for a total of five hours. Figure 15 through 17 show the x- and y-acceleration signals and the signal for angular velocity in the z-direction with time. Figure 18 shows the measured temperature and dropped packet percentage with time.

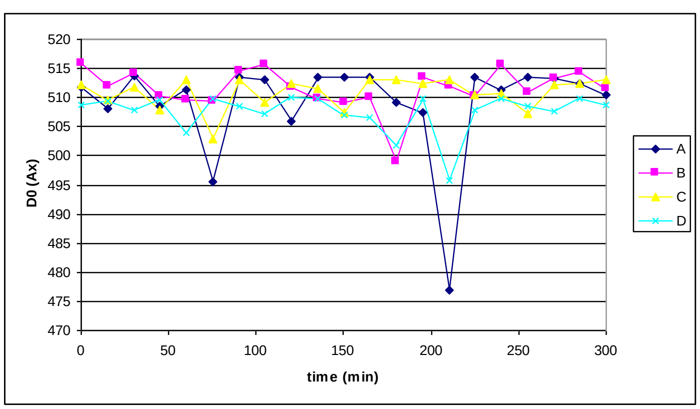

Figure 15: D0 Signal (Ax) vs. time

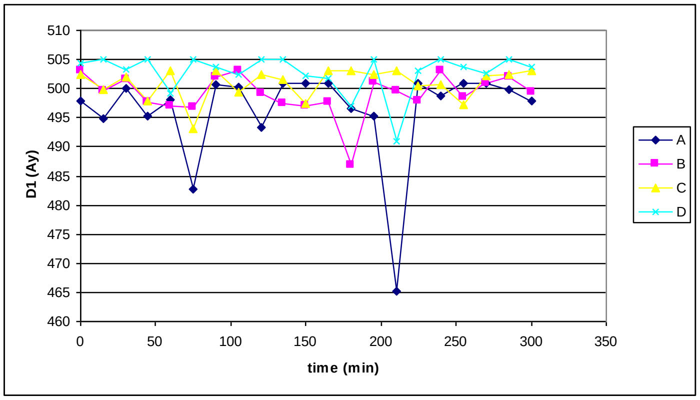

Figure 16: D1 Signal (Ay) vs. time

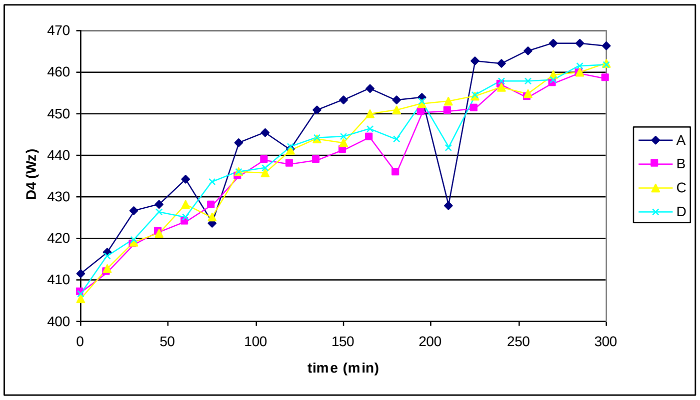

Figure 17: D4 Signal (Wz) vs. time

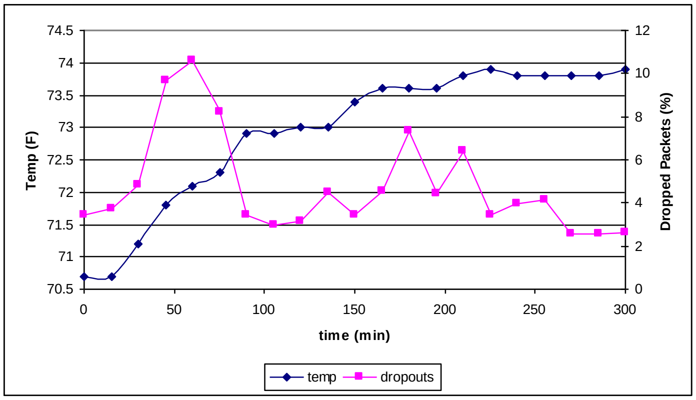

Figure 18: Temperature and dropped packets vs. time

The accelerometer shows very little zero drift over the course of the test while the gyroscope that measures the angular velocity in the z-direction shows significant zero drift that will need to be compensated during long lifting tests.

The outside temperature of the IMUs rises with time due to powering the device, but not significantly enough to cause problems with wearing the devices for extended periods of time. Dropped packet percentage also has no relationship with running time of the IMUs.

5.2. Pendulum tests

5.2.1. NIOSH Pendulum Test

A test was run at the NIOSH facility in Morgantown, WV to measure the x- and y-accelerations and z-angular velocity of three IMUs on a pendulum. The purpose of this test was to show the functionality of the system. A pendulum was used as pendulum motion is known analytically so the results could be validated easily. The setup for this test is shown in Figure 19. For the acceleration, the x-direction is perpendicular to the pendulum and the y-direction is along the pendulum. The IMUs for this test were built on small breadboards to validate the circuit design and an example of one is shown in Figure 20.

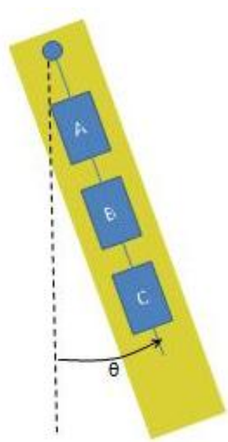

Figure 19: Pendulum test setup with breadboard IMUs

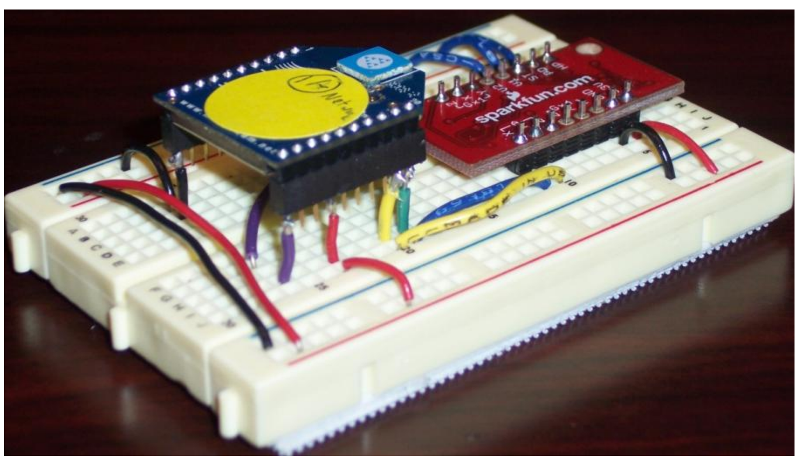

Figure 20: Breadboard IMU

The results of one run during testing are shown in Figure 21 through 23. Figure 21 shows the measured angular velocity of the three IMUs and came out as expected as the measured angular velocity should be the same for all IMUs. Figure 22 and Figure 23 show the measured acceleration in the x- and y-directions respectively. One might expect the acceleration signals to be the same, but this does not occur as explained below.

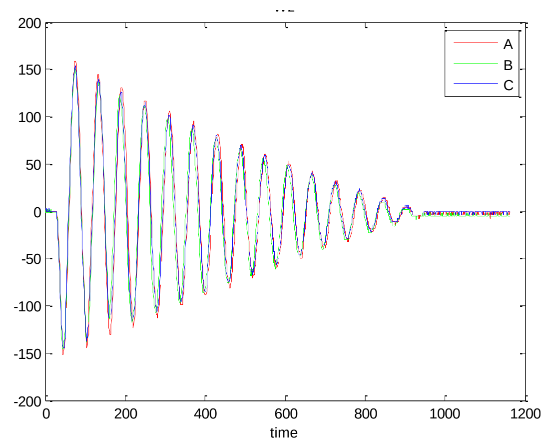

Figure 21: Angular velocity (degrees/second) vs. time for one NIOSH test

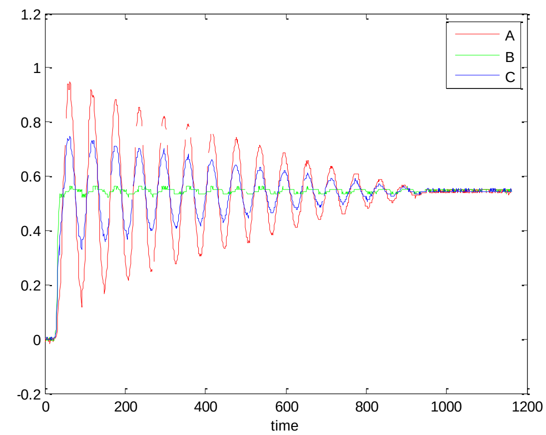

Figure 22: X-acceleration (units in g's) vs. time for one NIOSH test

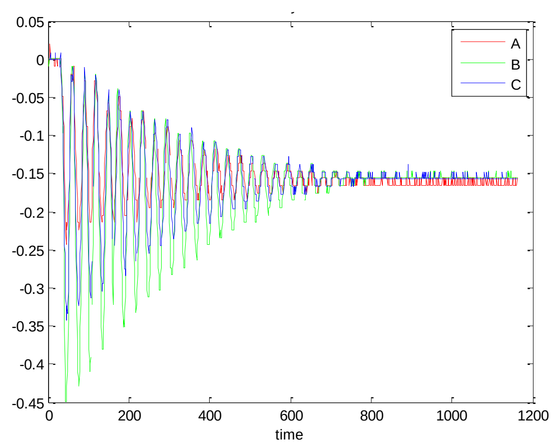

Figure 23: Y-acceleration (units in g's) vs. time for one NIOSH test

Separate tests were run to determine the cause of the irregularities with the measured accelerations. These tests were run by switching the location of IMUs then measuring their signals. Four tests were run with the results and IMU locations being shown in Figure 24 through 27.

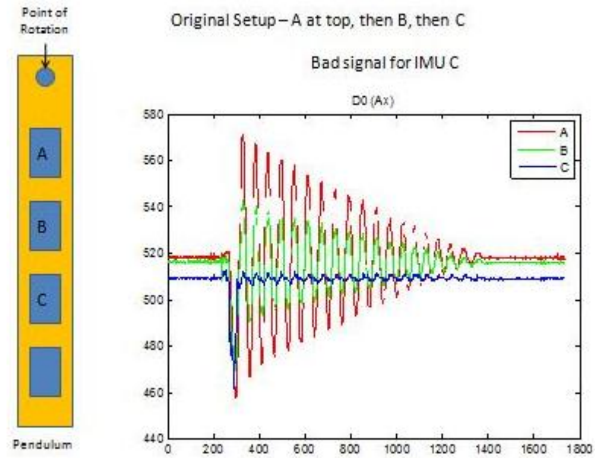

Figure 24: Test 1 setup and x-acceleration signal vs. time

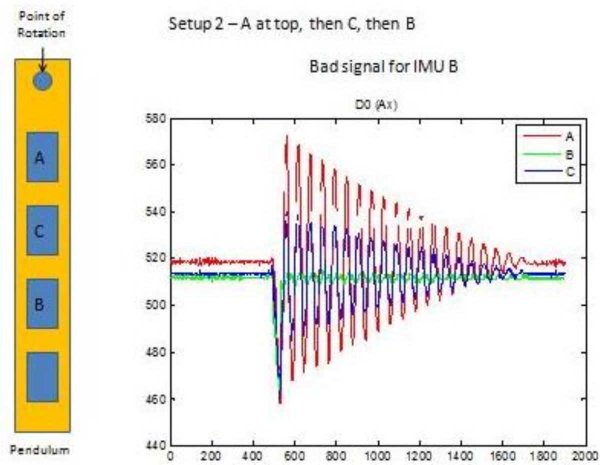

Figure 25: Test setup 2 and x-acceleration signal vs. time

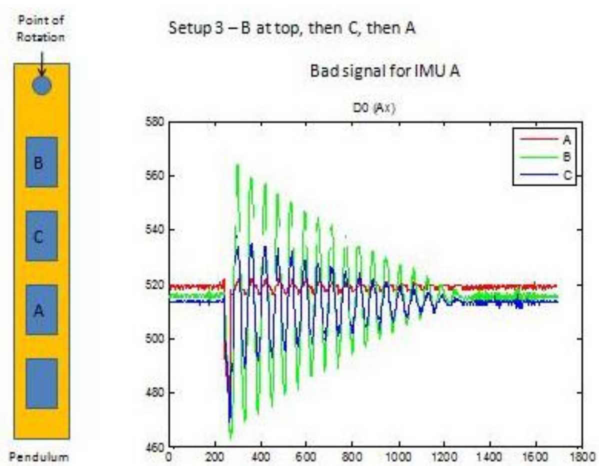

Figure 26: Test 3 setup and x-acceleration signals vs. time

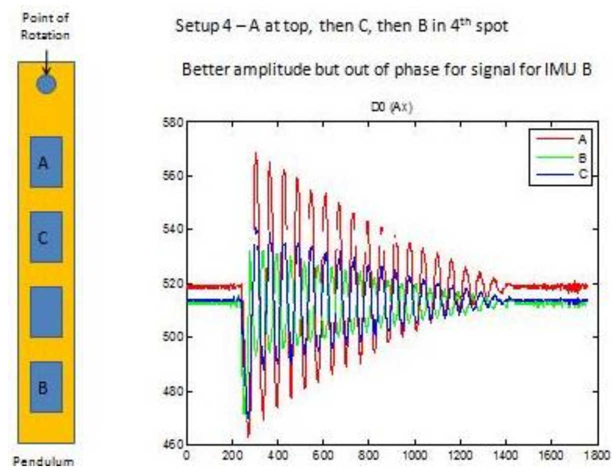

Figure 27: Test 4 setup and x-acceleration signal vs. time

From the results, the location of the IMU along the pendulum determines its measured acceleration. This phenomenon can be explained by investigating pendulum motion:

θ = pendulum angle (positive in direction shown in Figure 19)
θ_0 = initial angle of pendulum at release
f = frequency of oscillation dictated from the length of the pendulum, mass, and mass moment
t = time
$\ddot{\theta} = \text{angular acceleration of pendulum}$
$\dot{\theta} = \text{angular velocity of pendulum}$
$A_x = \text{Local horizontal acceleration of IMU on pendulum}$
$g = \text{static acceleration due gravity}$
$r = \text{distance from pivot of pendulum to IMU}$

$$\theta = \theta_0 \cos(2\pi ft)$$
$$\dot{\theta} = -\theta_0 (2\pi f) \sin(2\pi ft)$$
$$\ddot{\theta} = -\theta_0 (2\pi f)^2 \cos(2\pi ft)$$

$$A_{x,g} = g \sin\theta$$
$$A_{x,g} = g\theta \text{ for small angles}$$
$$A_{x,tan} = r\ddot{\theta}$$

$$A_{x,tot} = A_{x,g} + A_{x,tan} = g\theta + r\ddot{\theta}$$
$$A_{x,tot} = g\theta_0 \cos(2\pi ft) - r\theta_0 (2\pi f)^2 \cos(2\pi ft)$$
$$A_{x,tot} = (g - r(2\pi f)^2)\theta_0 \cos(2\pi ft)$$

$$\text{For } A_{x,tot} = 0 \rightarrow r = \frac{g}{(2\pi f)^2}$$

For f = 0.8224 Hz, calculated from the data, an IMU located 14.46 inches from the pivot will have $A_{x,tot}$ = 0. This is why IMU B is out of phase in test setup 4 shown in Figure 27. A Kalman filter will help alleviate this problem.

5.2.2. Pendulum test of sagittal IMUs

After production of the IMUs was complete, a second pendulum test was done with five IMUs. The IMUs were placed on the pendulum as shown in Figure 28 and tests were run in a similar fashion as in the NIOSH pendulum testing with the addition of initial calibration of the IMUs being done before this testing. The IMUs were calibrated to find the voltage offsets for the zero values of the sensors. After calibration, the testing was completed.

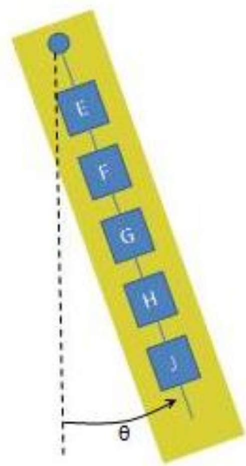

Figure 28: Pendulum test setup with final IMUs

The results of the testing are shown in Figure 29 and Figure 30. Figure 29 shows the calculated orientation angle of the IMUs and should be the same for all IMUs as they swing together on the pendulum. This does not occur because the angle is calculated using the accelerations, with the same phenomenon occurring as in the previous testing. Figure 30 shows the measured angular velocity of the IMUs. The signals are in phase and have the same amplitude as expected with discrepancies in the mean value being attributed to slight errors in the calibration. This test results shown had a dropout rate of 1.27% and a damaged packet rate of 0.96%. The results of this test were good and helped to validate the sensor system.

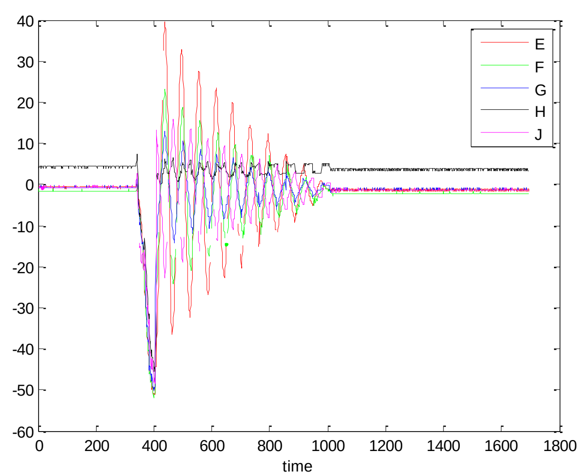

Figure 29: Calculated angle of pendulum in degrees vs. time

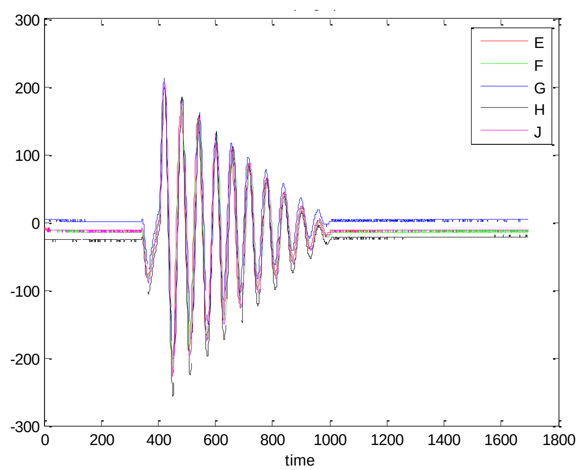

Figure 30: Measured angular velocity (deg/s) vs. time

5.2.3. Pendulum test of sagittal and coronal IMUs

After the data was validated for all sagittal plane IMUs, two IMUs were reprogrammed to be coronal plane IMUs. The coronal IMUs will be used on the pelvis and head. The setup for this test is shown in Figure 31.

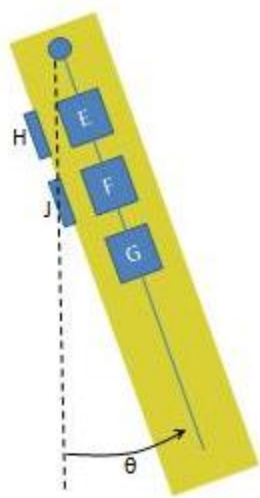

Figure 31: Pendulum test setup with sagittal and coronal IMUs

The coronal IMUs are programmed to send the z-acceleration, y-acceleration and y-angular velocity signals which should give the same measurements as the sagittal IMUs when aligned on the side of the pendulum as in Figure 31. The results of the test are shown in Figure 32 through 35. The coronal IMUs are H and J. The z-acceleration and y-angular velocity of the coronal IMUs should be the same as the x-acceleration and z-angular velocity of the sagittal IMUs, respectively.

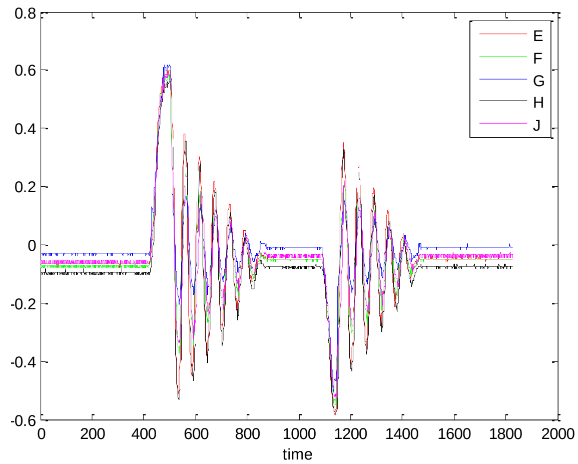

Figure 32: X-acceleration (g’s) vs. time of IMUs

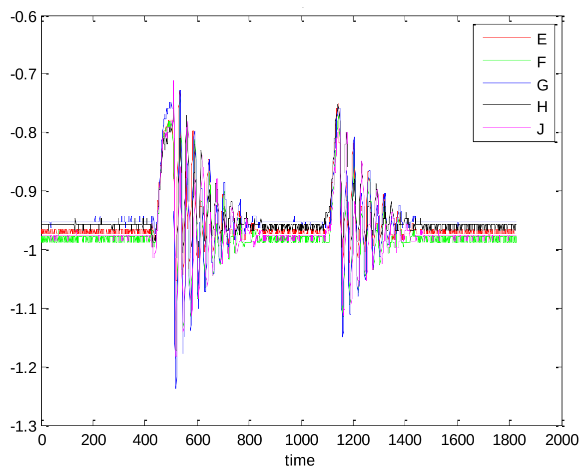

Figure 33: Y-acceleration (g's) vs. time of IMUs

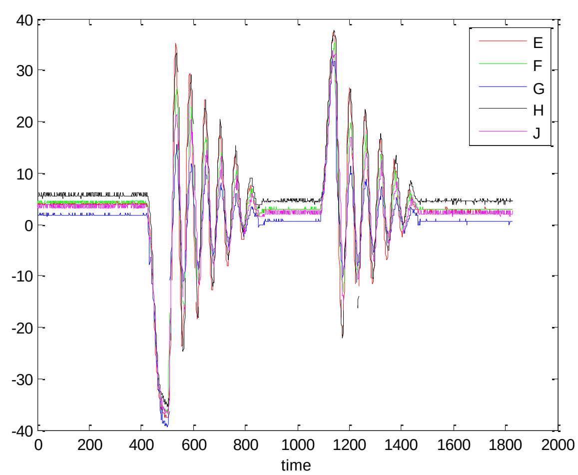

Figure 34: Measured angular position (degrees) of IMUs vs. time

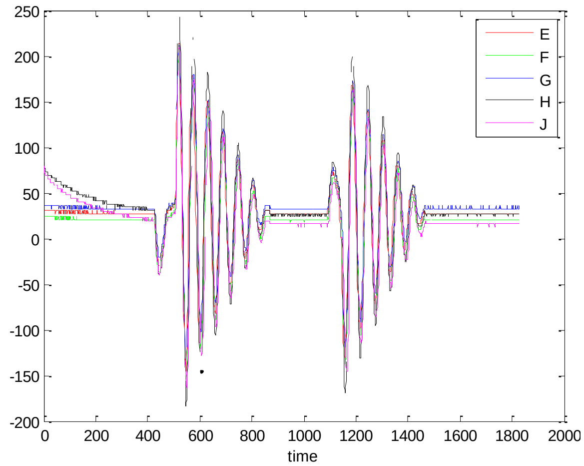

Figure 35: Angular velocity (deg/s) vs. time of IMUs

Figure 32 and Figure 33 show the x- and y-acceleration of each IMU respectively. The results of this test came out as expected as the acceleration signals matched up with each other much better than in the previous tests. This is a result of the IMU placement as the IMUs were grouped closer to the pivot of the pendulum and farther away from the point where the acceleration becomes zero. Figure 34 shows the calculated angle of each IMU by taking the inverse tangent of the ratio of x- to y-acceleration. Figure 35 shows the measured angular velocity of each IMU and came out as expected.

From the plots, the coronal IMUs match the sagittal IMUs and thus validate the use of them for the head and pelvis. The damaged packet rate for this test was 0.86% and the dropout rate was 0.61%, which are good results and show the data transmission of the network is unaffected by the change in signals of two of the IMUs.

6. Conclusion

The PBAS is a suite of sensors designed to measure the kinematics and kinetics of the human body during occupational lifting tasks. The suite consists of ten inertial measurement units (IMUs) synchronized with the Novel Pedar system. The IMUs were the focus of this thesis and measure the kinematics. They consist of an Xbee radio to send data and a 6DOF IMU Razor made by Sparkfun. The IMUs measure the accelerations and angular velocity of the subject in the sagittal plane.

The 1991 NIOSH lifting equation describes recommendations for occupational lifting. These equations are used to determine the maximum weight a worker should lift under certain lifting conditions. The 1991 equation is a revision of their previous equation to include changes in the frequency of lifting, asymmetrical lifting, and changes in the size and shape of handles and containers.

The basis for the model used in this thesis is the Freivalds model (1984). He modeled the human body as seven links (foot, lower leg, upper leg, pelvis, head-thorax-abdomen, hand-forearm, and upper arm). His model required the mass and length of each body segment, the motion of each segment, and the load being moved. The output of the Freivalds model included the position, angular velocity, acceleration, and joint reaction forces including the reaction forces on the lumbar-sacrum. However, his data was collected in the laboratory setting. In order to obtain data from a manufacturing setting, wireless data transmission is required. Wireless data has been collected for walking and Parkinson's disease testing, but not for lifting.

The PBAS uses an analytical model to combine acceleration and angular velocity data from the IMUs and calculate the joint reaction forces and kinematics of the limb segments. This model uses initial data from a standing calibration and the Dempster body segment parameters to determine the mass and length of each body segment as well as the positions of the center of mass and IMU. This data is then used to determine the angular positon, angular velocity, and translational acceleration for each body segment. Once the kinematics are obtained, the joint reaction forces can be calculated.

The IMUs are used to measure the kinematics needed for the model. They consist of an Xbee radio, a three-axis accelerometer, a two-axis gyroscope, and a single-axis gyroscope. They are commanded to sample and send their data by a Pacer device to a computer, where the data is collected and processed. The outputs of the processed data are matrices containing the angular position, angular velocity, and translational accelerations for each IMU.

Testing was initially done to determine the dependence of battery life, zero-drift, and dropout rate on transmission time. The tests showed that the acceleration signals' zero-level and the dropout rate did not change with time, but the gyroscope signals' zero-level do drift. This will need to be accounted for during long testing periods. The test ran for five hours with the battery voltage dropping from 3.12 V to 2.75 V. This validated the IMU design and choice of AA batteries as five hours is more than enough time for one test.

In order to validate the kinematic measurements of the IMUs, they were attached to a pendulum to measure its motion as pendulum motion is known analytically. An initial test was run at the NIOSH facility with three IMUs made on small breadboards in order to validate the circuit design and IMU communication. The measured angular velocity of each IMU matched as expected and the measured translational acceleration data was validated by a mathematical pendulum model.

Pendulum testing was also done on the finished IMUs with good results. One test was done with five IMUs configured for sagittal plane use and a second test was done with two IMUs configured for coronal plane use and three IMUs configured for sagittal use. Both tests provided the expected results and validated the IMU network.

As discussed in section 5.2.1, a Kalman filter will need to be implemented in the code in order to account for lateral acceleration effects in the measured data. The filter will check the variance of the signals in order to make sure they are being measured correctly. Once this is accomplished, kinematic and kinetic data can be measured for sagittal plane lifting. After the two-dimensional testing is complete, design of a three-dimensional PBAS will be considered.

References

Bridwell K. Anatomical planes of the body [Internet]. SpineUniverse.com; c1999-2010 [modified 2010 Feb 1; cited 2010 Feb 10].
Available from: http://www.spineuniverse.com/anatomy/anatomical-planes-body.

Buczek FL, Walker MR, Rainbow MJ, Cooney KM, Sanders JO. Impact of mediolateral segmentation on a multi-segment foot model. Gait & Posture. 2006;23:519-522.

Fong DT, Chan Y, Hong Y, Yung PS, Fung K, Chan K. Estimating the complete ground reaction forces with pressure insoles in walking. Journal of Biomechanics. 2008;41:2597-2601.

Freivalds A, Chaffin DB, Garg A, Lee KS. A dynamic evaluation of lifting maximum acceptable loads. Journal of Biomechanics. 1984;17(4):251-262.

Haug, E. Computer-aided kinematics and dynamics of mechanical systems. Needham Heights (MA): Allyn and Bacon; 1989.

LeMoyne R, Coroian C, Mastroianni T. Wireless accelerometer system for quantifying gait [Internet]. 2009 Feb 22 [cited 2010 Jan 12].
Available from http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=04906658

Liu K, Liu T, Shibata K, Inoue Y, Zheng R. Novel approach to ambulatory assessment of human segmental orientation on a wearable sensor system. Journal of Biomechanics. 2009;42(16):2747-2752.

Ren L, Jones RK, Howard D. Whole body inverse dynamics over a complete gait cycle based only on measured kinematics. Journal of Biomechanics. 2008;41:2750-2759.

Sparkfun Electronics. IMU 6DOF Razor – Ultra – Thin IMU [Internet]. Boulder (CO): Sparkfun Electronics; [cited 2010 Jan 12].
Available from: http://www.sparkfun.com/commerce/product_info.php?products_id=9431.

Stamatakis J, Gerard P, Drochmans P, Kezai T, Caby B, Macq B, Flandre D. Study and implementation of a wireless accelerometer network for gait analysis. 4th European Conference of the International Federation for Medical and Biological Engineering. Vol. 22. Springer Berlin Heidelberg; 2009. p. 2073-2076.

Waters TR, Putz-Anderson V, Garg A, Fine LJ. Revised NIOSH equation for the design and evaluation of manual lifting tasks. Ergonomics. 1993;36(7):749-776.

Winter DA. Biomechanics and motor control of human movement. 2nd edition. Toronto: John Wiley & Sons, Inc.; 1990.

Zampieri C, Salarian A, Carlson-Kuhta P, Aminian K, Nutt JG, Horak FB. The instrumented timed up and go test: potential outcome measure for disease modifying therapies in Parkinson's Disease [Internet]. 2009 Sep 2 [cited 2010 Jan 12]. JNNP Online.
Available from: http://jnnp.bmj.com/content/early/2009/09/02/jnnp.2009.173740.abstract

Appendix A: Dempster's body segment parameter data for 2-D studies (Winter 1990)

Segment name	Endpoints (proximal to distal)	Seg. mass /total mass (P)	Centre of mass /segment length ($R_{proximal}$)	(R_{distal})	Radius of gyration /segment length (K_{cg})	($K_{proximal}$)	(K_{distal})
Hand	wrist axis to knuckle II third finger	0.0060	0.506	0.494	0.297	0.587	0.577
Forearm	elbow axis to ulnar styloid	0.0160	0.430	0.570	0.303	0.526	0.647
Upper arm	glenohumeral joint to elbow axis	0.0280	0.436	0.564	0.322	0.542	0.645
Forearm & hand	elbow axis to ulnar styloid	0.0220	0.682	0.318	0.468	0.827	0.565
Upper extremity	glenohumeral joint to elbow axis	0.0500	0.530	0.470	0.368	0.645	0.596
Foot	lateral malleolus to head metatarsal II	0.0145	0.500	0.500	0.475	0.690	0.690
Leg	femoral condyles to medial malleolus	0.0465	0.433	0.567	0.302	0.528	0.643
Thigh	greater trochanter to femoral condyles	0.1000	0.433	0.567	0.323	0.540	0.653
Leg & foot	femoral condyles to medial malleolus	0.0610	0.606	0.394	0.416	0.735	0.572
Lower extremity	greater trochanter to medial malleolus	0.1610	0.447	0.553	0.326	0.560	0.650
Head	C7-T1 to ear canal	0.0810	1.000	0.000	0.495	1.116	0.495
Shoulder	sternoclavicular joint to glenohumeral joint	0.0158	0.712	0.288			
Thorax	C7-T1 to T12-L1	0.2160	0.820	0.180			
Abdomen	T12-L1 to L4-L5	0.1390	0.440	0.560			
Pelvis	L4-L5 to trochanter	0.1420	0.105	0.895			
Thorax & abdomen	C7-T1 to L4-L5	0.3550	0.630	0.370			
Abdomen & pelvis	T12-L1 to greater trochanter	0.2810	0.270	0.730			
Trunk	greater trochanter to glenohumeral joint	0.4970	0.495	0.505	0.406	0.640	0.648
Trunk & head	greater trochanter to glenohumeral joint	0.5780	0.660	0.340	0.503	0.830	0.607
Head, arms & trunk	greater trochanter to glenohumeral joint	0.6780	0.626	0.374	0.496	0.798	0.621
Head, arms & trunk	greater trochanter to midrib	0.6780	1.142	-0.142	0.903	1.456	0.914

Equations:

$$\sum_{i=1}^{n} P_i = 1.000$$

where n is the number of body segments and i is the segment number and P_i is the segment mass proportion

$$m_{total\ body} = \sum_{i=1}^{n} m_i$$

m_i is mass of a segment

$$R_{proximal} + R_{distal} = 1.000$$

R is distance to centre of gravity as proportion of segment length

$$r_{proximal} = R_{proximal} \times length$$

$r_{proximal}$ is distance from centre of gravity to proximal end

$$s_{cg} = s_{proximal} + R_{proximal}\,(s_{distal} - s_{proximal})$$

s represents position in x, y or z directions

$$s_{limb} = \frac{\sum_{i=1}^{L} P_i\, s_{cg_i}}{\sum_{i=1}^{L} P_i}$$

where L is the number of segments in the limb

$$s_{total\ body} = \sum_{i=1}^{n} P_i\, s_{cg_i}$$

$$k_{proximal} = K_{proximal} \times length$$

$k_{proximal}$ is radius of gyration for axes through the proximal end and $K_{proximal}$ is the radius of gyration as a proportion of the segment length

$$K_{cg} = \sqrt{K_{proximal}^2 - R_{proximal}^2}$$

$$K_{proximal} = \sqrt{K_{cg}^2 + R_{proximal}^2}$$

$$I_{cg} = m\,(K_{cg} \times length)^2$$

I_{cg} is moment of inertia about an axis through the centre of gravity

$$I_{proximal} = m\,k_{cg}^2 + m\,r_{proximal}^2$$

$$I_{proximal} = m\,(K_{cg} \times length)^2 + m\,(R_{proximal} \times length)^2$$

$$I_{total\ body} = \sum_{i=1}^{n} I_{cg_i} + \sum_{i=1}^{n} m_i r_i^2$$

where r_i is the distance between the total body centre of gravity and each segment's centre of gravity

Appendix B: Vector and Matrix Notation

$\left\{{}^{\mathrm{REF\,FR}}\mathrm{r}_{\mathrm{PT}}\right\}$ position of a point measured relative to a reference frame

2D example $\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{B4}}\right\}=\begin{Bmatrix}{}^{\mathrm{C2}}\mathrm{x}_{\mathrm{B4}}\\ {}^{\mathrm{C2}}\mathrm{y}_{\mathrm{B4}}\end{Bmatrix}$ position of point B attached to body 4 relative to coordinate frame C attached to body 2

3D example $\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{B4}}\right\}=\begin{Bmatrix}{}^{\mathrm{C2}}\mathrm{x}_{\mathrm{B4}}\\ {}^{\mathrm{C2}}\mathrm{y}_{\mathrm{B4}}\\ {}^{\mathrm{C2}}\mathrm{z}_{\mathrm{B4}}\end{Bmatrix}$ position of point B attached to body 4 relative to coordinate frame C attached to body 2

$\left\{{}^{\mathrm{REF\,FR}}\dot{\mathrm{r}}_{\mathrm{PT}}\right\}$ velocity of a point measured relative to a reference frame

$\left\{{}^{\mathrm{REF\,FR}}\ddot{\mathrm{r}}_{\mathrm{PT}}\right\}$ acceleration of a point measured relative to a reference frame

$\left\{{}^{\mathrm{REF\,FR}}\dddot{\mathrm{r}}_{\mathrm{PT}}\right\}$ jerk of a point measured relative to a reference frame

$\left\{{}^{\mathrm{REF\,FR}}\mathrm{s}_{\mathrm{PT/PT}}\right\}$ relative location of two points on the same body measured relative to directions defined by a reference frame

example $\left\{{}^{\mathrm{C2}}\mathrm{s}_{\mathrm{A3/B3}}\right\}=\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{A3}}\right\}-\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{B3}}\right\}$ relative location of point A attached to body 3 with respect to point B attached to body 3 measured relative to directions defined by coordinate frame C attached to body 2

$\left\{{}^{\mathrm{REF\,FR}}\mathrm{d}_{\mathrm{PT/PT}}\right\}$ relative location of two points on different bodies measured relative to directions defined by a reference frame

example $\left\{{}^{\mathrm{C2}}\mathrm{d}_{\mathrm{A4/B3}}\right\}=\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{A4}}\right\}-\left\{{}^{\mathrm{C2}}\mathrm{r}_{\mathrm{B3}}\right\}$ relative location of point A attached to body 4 with respect to point B attached to body 3 measured relative to directions defined by coordinate frame C attached to body 2

${}^{\mathrm{REF\_FR}}\theta_{\mathrm{FR}}$ 2D attitude angle of a given coordinate frame with respect to a reference frame

$\dot{\theta}_{\mathrm{BODY}}$ 2D angular velocity of one body

$\ddot{\theta}_{\mathrm{BODY}}$ 2D angular acceleration of one body

$\ddot{\theta}_{\mathrm{BODY}}$ 2D angular acceleration of one body

$\dddot{\theta}_{BODY}$ 2D angular jerk of one body

$\left\{{}^{REF\ FR}F_{PTonPT}\right\}$ force of one point exerted onto another point in directions described by a reference frame

example $\left\{{}^{C2}F_{B4onB3}\right\}$ force of point B attached to body 4 onto point B attached to body 3 measured relative to directions defined by coordinate frame C attached to body 2

$\left\{{}^{REF\ FR}T_{BODYonBODY}\right\}$ torque of one body exerted onto another body in directions described by a reference frame

example $\left\{{}^{C2}T_{4on3}\right\}$ torque of body 4 onto body 3 measured relative to directions defined by coordinate frame C attached to body 2

$\left[{}^{REF\ FR}A_{FR}\right]$ rotation matrix that describes relative attitude of a given coordinate frame with respect to a reference frame

example $\left[{}^{C2}A_{B3}\right]$ rotation matrix that describes attitude of coordinate frame B attached to body 3 with respect to coordinate frame C attached to body 2

$\left\{{}^{REF\ FR}\hat{f}_{FR}\right\}$ unit vector along local x axis measured with respect to a reference frame

$\left\{{}^{REF\ FR}\hat{g}_{FR}\right\}$ unit vector along local y axis measured with respect to a reference frame

$\left\{{}^{REF\ FR}\hat{h}_{FR}\right\}$ unit vector along local z axis measured with respect to a reference frame

3D example $\left\lfloor{}^{C2}A_{B3}\right\rfloor = \left[\left\{{}^{C2}\hat{f}_{B3}\right\} \quad \left\{{}^{C2}\hat{g}_{B3}\right\} \quad \left\{{}^{C2}\hat{h}_{B3}\right\}\right]$ unit directions of local axes for coordinate frame B attached to body 3 measured relative to coordinate frame C attached to body 2

$\left\{{}^{REF\ FR}v_{BODY}\right\}$ vector attached to one body measured with respect to a reference frame

$\left\{{}^{REF\ FR}p_{PT}\right\}$ homogeneous coordinates of a point measured relative to a reference frame used in 4x4 matrix methods

example $\left\{{}^{C2}p_{B4}\right\} = \left\{\begin{matrix} 1 \\ \left\{{}^{C2}r_{B4}\right\} \end{matrix}\right\}$ position of point B attached to body 4 relative to coordinate frame C attached to body 2

$\left[{}^{REF\ FR}T_{FR}\right]$ homogeneous coordinate transformation that describes relative location and attitude of a given coordinate frame with respect to a reference frame

example $\left[{}^{C2}T_{B3}\right]=\begin{bmatrix} 1 & \{0 \quad 0 \quad 0\} \\ \{{}^{C3}r_{B3}\} & \left[{}^{C2}A_{B3}\right] \end{bmatrix}$ coordinate transformation that describes location and attitude of coordinate frame B attached to body 3 with respect to coordinate frame C attached to body 2

Numbering and lettering

Bodies should be numbered consecutively beginning with 1. Body 1 is typically reserved for ground.
Points and coordinate frames should be lettered and should have a number for the specific body onto which they are attached.
Point O_i is typically reserved for the origin of a general coordinate frame attached to body i.
Point G_i is typically reserved for the mass centroid of body i.

Reference frames

If no reference frame is explicitly provided, information is measured relative to ground frame O_1.
If a body number but no point is provided for a reference frame, information is measured relative to the general origin O_i for that body

Subscripts and superscripts outside vector/matrix brackets

Post-superscript outside vector/matrix brackets is reserved for standard vector/matrix operations such as inverse, transpose and powers.
Post-subscript outside vector/matrix brackets is reserved for general descriptors or time values.
No pre-superscript or pre-subscript should appear outside brackets.

Subscripts and superscripts inside vector/matrix brackets

No post-superscript should appear inside vector/matrix brackets
Post-subscript inside vector/matrix brackets denotes a body, a point or a coordinate frame.
Pre-superscript inside vector/matrix brackets denotes reference coordinate frame. If no reference frame is specified, it indicates a reference frame attached to ground 1.
No pre-subscript should appear inside vector/matrix brackets

General vector/matrix operations

$\{\ \}^T, [\ ]^T$ vector/matrix transpose
$[\ ]^{-1}$ matrix inverse
$\det[\ ]$ determinant of matrix
tr[ ] trace of matrix (sum of diagonal elements)
$\{\mathrm{diag}[\ ]\}$ diagonal elements of matrix rearranged into column vector
$[\mathrm{diag}\{\ \}]$ elements of vector placed into a diagonal matrix
$[\ ]^n$ matrix to power n
norm{ } scalar norm of vector (magnitude)
$\{\hat{u}\}$ unit vector
$[\{\tilde{a}\}]$ 3x3 skew-symmetric matrix formed from vector $\{a\}$ (performs cross-product)
$[I_n]$ identity matrix of order n
$\{0\},[0]$ vector/matrix of zeros

Appendix C: Matlab Code

C-1. Main code to combine data from 2 sets of IMUs

```
% imu_combine_data.m: combine data from two receivers
% MLB, 1/11/10

clear all
clf

% get data from 1st set of IMU's (A,B,C,D,I)
fidData1 = fopen( '10_1_12_data/imu10_test5_a.bin' );
% number of imu's in 1st set of data
nimu1 = 5;
% number of samples per packet
it1 = 3;
% data set (1 or 2)
set1 = 1;
% 1 if H,J are set as frontal IMUs; 0 if H,J are set as sagittal IMUs
frontal1 = 0;
[Ax1,Ay1,Wz1,total1,bad_total1,drop_total1,ndrop1] =
get_data(set1,nimu1,it1,fidData1,frontal1);

% get data from 2nd set of IMU's (E,F,G,H,J)
fidData2 = fopen( '10_1_12_data/imu10_test5_b.bin' );
% number of imu's in 2nd set of data
nimu2 = 5;
% number of samples per packet
it2 = 3;
% data set (1 or 2)
set2 = 2;
% 1 if H,J are set as frontal IMUs; 0 if H,J are set as sagittal IMUs
frontal2 = 0;
[Ax2,Ay2,Wz2,total2,bad_total2,drop_total2,ndrop2] =
get_data(set2,nimu2,it2,fidData2,frontal2);

% total percent bad and dropped packets
bad_percent = 100*(bad_total1 + bad_total2)/(total1 + total2)
drop_percent = 100*(drop_total1 + drop_total2)/(total1 + total2)

% Calculate length of combined matrices
if length(Ax1)>length(Ax2)
    len = length(Ax1);
else
    len = length(Ax2);
end
nimu = nimu1+nimu2;
Ax = zeros(len,nimu);
Ay = zeros(len,nimu);
Wz = zeros(len,nimu);

% Insert first set into 1st 4 columns and second set into 2nd 4 columns
Ax(1:length(Ax1),1:nimu1) = Ax1;
Ax(1:length(Ax2),nimu1+1:nimu) = Ax2;
```

```
Ay(1:length(Ay1),1:nimu1) = Ay1;
Ay(1:length(Ay2),nimu1+1:nimu) = Ay2;

Wz(1:length(Wz1),1:nimu1) = Wz1;
Wz(1:length(Wz2),nimu1+1:nimu) = Wz2;

IMU = zeros(nimu,1);
% set IMU names used based on number of IMU's being used
for i=1:nimu
    IMU(i) = 65 + (i-1);
end

% Use the length of the shortest matrix for plotting (avoid the drop to
% zero)
diff = abs(length(Ax1)-length(Ax2));
DataPlot1(nimu,IMU,Ax,Ay,Wz,diff)
```

C-2. Function getdata: used to rip out data from binary files

```
function [Ax,Ay,Wz,total,bad_total,drop_total,ndrop] =
get_data(set,nimu,it,fidData,frontal)
% set = 1 for IMUs A,B,C,D,I, set = 2 for IMUs E,F,G,H,J
% nimu = number of imu's
% it = number of samples per packet
% frontal = 1 if H,J are set as frontal IMUs; 0 if H,J are set as sagittal
% IMUs
%% Initializations

if (it==2)
    num_bytes = 24;
elseif (it==3)
    num_bytes = 30;
end
% number of bytes in a packet (for IT = 2)
% num_bytes = 24;
% number of bytes in a packet (for IT = 3)
% num_bytes = 30;

% read data from file
s = fread( fidData, inf, 'uint8' );
% Set 1: IMU's A,B,C,D,I
if (set==1)
    % set acknowledgment for last imu of set
    switch (nimu)
        case {4}
            % acknowledgment of last IMU (D)
            str1 = '126;0;15;151;1;0;19;162;0;64;45;220;31;0;68;73;82;0;107';
        case {5}
            % acknowledgment of last IMU (I)
            str1 =
'126;0;15;151;1;0;19;162;0;64;45;219;183;0;73;73;82;0;207';
    end
    % set IMU names used based on number of IMU's being used
    IMU = zeros(nimu,1);
    if nimu<=4
        for i=1:nimu
            IMU(i) = 65 + (i-1);
        end
    else
        IMU = [65;66;67;68;73];
    end
% Set 2: IMU's E,F,G,H,J
else
    % set acknowledgment for last imu of set
    switch (nimu)
        case {3}
            % acknowledgment of last IMU (G)
            str1 = '126;0;15;151;1;0;19;162;0;64;45;220;20;0;71;73;82;0;115';
        case {4}
            % acknowledgment of last IMU (H)
            str1 =
'126;0;15;151;1;0;19;162;0;64;45;219;184;0;72;73;82;0;207';
        case {5}
```

```
            % acknowledgment of last IMU (J)
            str1 = '126;0;15;151;1;0;19;162;0;64;45;220;22;0;74;73;82;0;110';
    end
    % set IMU names used based on number of IMU's being used
    IMU = zeros(nimu,1);
    if nimu<=4
        for i=1:nimu
            IMU(i) = 69 + (i-1);
        end
    else
        IMU = [69;70;71;72;74];
    end
end

%% Read data from .bin file and put into matrix
% turn data vector into string
S = mat2str(s);
% find index of start of acknowledgment and length of
index = findstr(S, str1);

% check if the acknowledgment is found, if not then keep all of S
if (isempty(index))
    str_data = S;
else
    % create new string starting after acknowledgment (good data)
    S1 = S((index(1)+length(str1)+1):length(S));
    % add a [ to beginning of string for conversion to matrix
    str_data = strcat(S(1), S1);
end

% convert good data string to vector
good_data = eval(str_data);

% find indices of start delimiter
check_data = good_data - 126;
A = ~xor(check_data,0);
start_ind = find(A);

%%Check for bad packets, put good ones into new data matrix
junk_count_high = 0;
junk_count_low = 0;
good_count = 0;
row = 1;
i = 1;
while(i<=length(start_ind)-1)
    % packet too long
    if (start_ind(i+1) - start_ind(i))>num_bytes
        junk_count_high = junk_count_high+1;
        % fill bad data rows with NaN
        for n = 1:num_bytes
            data(row,n) = NaN;
        end
        % put IMU name into proper spot in data matrix
        % if IMU name is off in packet, get right one
        if good_data(start_ind(i)+4)==0
            data(row,5) = good_data(start_ind(i)+4);
```

```
            data(row,6) = good_data(start_ind(i)+5);
        else
            data(row,5) = good_data(start_ind(i)+5);
            data(row,6) = good_data(start_ind(i)+6);
        end

    % packet too short
    elseif (start_ind(i+1) - start_ind(i))<num_bytes
        % check if there are multiple 126's in a single packet
        y = 2;
        while ((start_ind(i+y)-start_ind(i))<=num_bytes)
            % Packet is right length now
            if (start_ind(i+y)-start_ind(i))==num_bytes
                good_count = good_count+1;
                data(row,1) = good_data(start_ind(i));
                for n = 2:num_bytes
                    data(row,n) = good_data(start_ind(i)+n-1);
                end
                i = i+y;
                if (i+y)>length(start_ind)
                    break
                end
            % Packet still too short
            elseif (start_ind(i+y)-start_ind(i))<num_bytes
                y = y+1;
            % Packet bad
            else
                junk_count_low = junk_count_low+1;
                for n = 1:num_bytes
                    data(row,n) = NaN;
                end
                % put IMU name into proper spot in data matrix
                % if IMU name is off in packet, get right one
                if good_data(start_ind(i)+4)==0
                    data(row,5) = good_data(start_ind(i)+4);
                    data(row,6) = good_data(start_ind(i)+5);
                else
                    data(row,5) = good_data(start_ind(i)+5);
                    data(row,6) = good_data(start_ind(i)+6);
                end
             end
        end

    % good packet
    else
        good_count = good_count+1;
        data(row,1) = good_data(start_ind(i));
        for n = 2:num_bytes
            data(row,n) = good_data(start_ind(i)+n-1);
        end
    end
    row = row+1;
    i = i+1;
end
% final row count
row = row-1;
% send back total to calculate percentage bad for combined data
```

```
total = junk_count_low + junk_count_high + good_count;
bad_total = junk_count_low + junk_count_high;
percent_bad = 100*(junk_count_low + junk_count_high)/(junk_count_low +
junk_count_high + good_count);
%% Check for rows of all zeros in data matrix and get rid of them
count=0;
for r=1:row
    if (data(r,1)==0)
        count = count+1;
    else
        data1(r-count,:) = data(r,:);
    end
end
row = length(data1);

%% Check for dropped packets, put NaN in as place holder for dropped packet
IMUname = zeros(it*length(data1),1);
v0 = zeros(it*length(data1),1);
v1 = zeros(it*length(data1),1);
v2 = zeros(it*length(data1),1);
% set number of dropped packets for each imu to zero (column 1=A, 2=B, etc.)
ndrop = zeros(nimu,1);

for r=1:nimu:row-(nimu-1)
    data_ind = r*it-(it-1);          % transform row number into data vector
index for v0, v1, and v2
    name_vec = zeros(nimu,1);        % initialize name vector (4x1)
    for x=0:nimu-1
        name_vec(x+1) = data1(r+x,5)*256 + data1(r+x,6);
    end
    % subtract IMU name from name_vec and use find to determine the number
    % of non-zero numbers in vector, if <=nimu-1, then the IMU name shows
    % up. if =nimu, then the IMU is missing (packet dropped)
    for x=1:nimu
        missing_imu(x) = length(find(name_vec-IMU(x)));
        % if missing_imu(1)=nimu, then there is no A in the name_vec (packet
was dropped)
        % if missing_imu(2)=nimu, then there is no B in the name_vec (packet
was dropped)
        % etc.
        if missing_imu(x)==nimu
            ndrop(x) = ndrop(x)+1;
            IMUname(data_ind) = IMU(x);
            v0(data_ind) = NaN;
            v1(data_ind) = NaN;
            v2(data_ind) = NaN;
            IMUname(data_ind+1) = IMUname(data_ind);
            v0(data_ind+1) = NaN;
            v1(data_ind+1) = NaN;
            v2(data_ind+1) = NaN;
            if (it==3)
                IMUname(data_ind+2) = IMUname(data_ind);
                v0(data_ind+2) = NaN;
                v1(data_ind+2) = NaN;
                v2(data_ind+2) = NaN;
            end
        end
```

```
    end
end
% send back total to calculate percentage dropped for combined data
drop_total = sum(ndrop);
percent_drop  = 100*(sum(ndrop))/(junk_count_low + junk_count_high +
good_count);
%% Put voltage values into row vectors for analysis
x = 1;
for r=1:row
    if x>length(v0)
        break
    end
    % only put in values for good packets (dropped packets have NaN already)
    if (~isnan(v0(x)))
        % put values from data matrix into row vectors
        IMUname(x) = data1(r,5)*256 + data1(r,6);
        v0(x) = data1(r,12)*256 + data1(r,13);      % 256 * high_byte +
low_byte
        v1(x) = data1(r,14)*256 + data1(r,15);
        v2(x) = data1(r,16)*256 + data1(r,17);
        IMUname(x+1) = IMUname(x);
        v0(x+1) = data1(r,18)*256 + data1(r,19);    % 256 * high_byte +
low_byte
        v1(x+1) = data1(r,20)*256 + data1(r,21);
        v2(x+1) = data1(r,22)*256 + data1(r,23);
        if (it==3)
            IMUname(x+2) = IMUname(x);
            v0(x+2) = data1(r,24)*256 + data1(r,25);    % 256 * high_byte +
low_byte
            v1(x+2) = data1(r,26)*256 + data1(r,27);
            v2(x+2) = data1(r,28)*256 + data1(r,29);
        end
    end
    x = x+it;
end

%% Split up voltage data for different IMU's
% Each IMU corresponds to a column in each matrix
% counter for each imu column
for x=1:nimu
    a(x) = 1;
end
% split up data into matrices
for x=1:length(IMUname)
    for i=1:nimu
        if IMUname(x)==IMU(i)
            D0(a(i),i) = v0(x);
            D1(a(i),i) = v1(x);
            D2(a(i),i) = v2(x);
            a(i) = a(i)+1;
        end
    end
end

%% plot voltages for different IMU's
time = 1:length(D0)-50;
time1 = length(D0)-50;
```

```
% For frontal IMUs (H, J), D0 in this code will be Ay and D1 will be Az
% Need to switch these since Az ==> Ax for frontal IMUs
if (nimu>=4)&&(set==2)&&(frontal==1)
    D1(:,4:nimu) = Az;
    D1(:,4:nimu) = D0(:,4:nimu);
    D0(:,4:nimu) = Az;
end

% Correct for initial voltages in sensors using zero voltage data from
% calibration files
load cal.mat
for i=1:nimu
    D0(:,i) = D0(:,i)-D0cal(i);
    D1(:,i) = D1(:,i)-D1cal(i);
    D2(:,i) = D2(:,i)-D2cal(i);
end

% Acceleration
% Ax=0 when D0=512, Ax=3g when D0=819 (2.4 V), Ax=-3g when D0=205 (0.6 V)
% 1g per 300mV from ADXL335 spec sheet, Vref = 3 V
% 2.4 V = +3g -> 819 in digital voltage
Ax = (D0-512)/(819-512)*3;
Ay = (D1-512)/(819-512)*3;

% Angular Velocity
% Wz=0 when D2=419, Wz=1200 (not amplified output) when D2=760 (2.23 V),
% Wz=-1200 when D2=80 (0.234 V) (0.83 mV/deg/s)
Wz = (D2-419)/(760-419)*1200;

% Calculate IMU orientation angle
theta = atan(Ax./Ay);
theta = theta*180/pi;

% Call function to plot data
% DataPlot(nimu,IMU,D0,D1,D2,Ax,Ay,Wz,theta);
```

C-3. Function DataPlot1: used to plot data vs. time

```
function DataPlot1(nimu,name_vec,Ax,Ay,Wz,diff)

% Number of samples in test, only use length of shortest matrix (1 or 2)
% extra zeros in those columns
time = 1:length(Ax)-diff;
time1 = length(Ax)-diff;
% Create vector of names for legend
names = char(name_vec);
leg = cellstr(names);

% set up color scheme
color = [1 0 0;      % red
         0 1 0;      % green
         0 0 1;      % blue
         0 0 0;      % black
         1 0 1;      % magenta
         0 1 1;      % cyan
         1 1 0;      % yellow
         1 0 0;      % red
         0 1 0;      % green
         0 0 1];     % blue

% Plot Ax
figure(1)
for i=1:nimu
    plot(time,Ax(1:time1,i),'Color',color(i,:))
    hold on
end
title('Ax')
xlabel('time')
legend(leg)

% Plot Ay
figure(2)
for i=1:nimu
    plot(time,Ay(1:time1,i),'Color',color(i,:))
    hold on
end
title('Ay')
xlabel('time')
legend(leg)

% Plot Wz
figure(3)
for i=1:nimu
    plot(time,Wz(1:time1,i),'Color',color(i,:))
    hold on
end
title('Wz')
xlabel('time')
legend(leg)
```

C-4. Matlab code to combine two calibration data sets

```
% imu_calibration.m: use two calibration data files to calibrate the zero
% voltage of the IMU signals
% MLB, 3/18/10

clear all
clc

% get data from 1st calibration file
fidData1 = fopen( '10_3_18_calibration1.bin' );
% number of imu's in 1st set of data
nimu = 5;
% number of samples per packet
it = 3;
% data set (1 or 2)
set = 2;
% Process calibration data
[D0avg1, D1avg1, D2avg1, drop1] = calibration(set,nimu,it,fidData1);

% get data from 2nd calibration file)
fidData2 = fopen( '10_3_18_calibration2.bin' );
% Process calibration data
[D0avg2, D1avg2, D2avg2, drop2] = calibration(set,nimu,it,fidData2);

% Subtarct the nominal zero value from the avergage of each IMU to get the
% calibration value for each IMU
% Ax, Ay, Az = 0 when D0, D1 = 512 and Wz = 0 when D2 = 419
D0avg = (D0avg1 + D0avg2)/2;
D1avg = (D1avg1 + D1avg2)/2;
D2avg = (D2avg1 + D2avg2)/2;

D0cal = D0avg-512;
D1cal = D1avg-512;
D2cal = D2avg-419;

% Save calibration numbers to .mat file for use by main program
savefile = 'cal.mat';
save(savefile, 'D0cal','D1cal','D2cal')
```

C-5. Function calibration: used to rip out data from binary files of calibration tests

```
% calibration.m: reads in calibration data from IMUs to get zero values for
% all signals
% The difference between this value and the nominal zero value is
% subtracted off of the measured voltages of the IMUs during regular
% testing
% MLB, 3/17/09

function [D0avg, D1avg, D2avg, percent_drop] =
calibration(set,nimu,it,fidData)
% set = 1 for IMUs A,B,C,D,I, set = 2 for IMUs E,F,G,H,J
% nimu = number of imu's
% it = number of samples per packet
% fidcal = calibration data file

%% Initializations

if (it==2)
    num_bytes = 24;
elseif (it==3)
    num_bytes = 30;
end

% read data from file
s = fread( fidData, inf, 'uint8' );

% IMU's A,B,C,D,I
if (set==1)
    switch (nimu)
        case {4}
            % acknowledgment of last IMU (D)
            str1 = '126;0;15;151;1;0;19;162;0;64;45;220;31;0;68;73;82;0;107';
        case {5}
            % acknowledgment of last IMU (I)
            str1 =
'126;0;15;151;1;0;19;162;0;64;45;219;183;0;73;73;82;0;207';
    end
    % set IMU names used based on number of IMU's being used
    IMU = zeros(nimu,1);
    if nimu<=4
        for i=1:nimu
            IMU(i) = 65 + (i-1);
        end
    else
        IMU = [65;66;67;68;73];
    end
% IMU's E,F,G,H,J
else
    switch (nimu)
        case {4}
            % acknowledgment of last IMU (H)
            str1 =
'126;0;15;151;1;0;19;162;0;64;45;219;184;0;72;73;82;0;207';
        case {5}
            % acknowledgment of last IMU (J)
```

```
            str1 = '126;0;15;151;1;0;19;162;0;64;45;220;22;0;74;73;82;0;110';
    end
    % set IMU names used based on number of IMU's being used
    IMU = zeros(nimu,1);
    if nimu<=4
        for i=1:nimu
            IMU(i) = 69 + (i-1);
        end
    else
        IMU = [69;70;71;72;74];
    end
end

%% Read data from .bin file and put into matrix
% turn data vector into string
S = mat2str(s);
% find index of start of acknowledgment and length of
index = findstr(S, str1);

% check if the acknowledgment is found, if not then keep all of S
if (isempty(index))
    str_data = S;
else
    % create new string starting after acknowledgment (good data)
    S1 = S((index(1)+length(str1)+1):length(S));
    % add a [ to beginning of string for conversion to matrix
    str_data = strcat(S(1), S1);
end

% convert good data string to vector
good_data = eval(str_data);

% find indices of start delimiter
check_data = good_data - 126;
A = ~xor(check_data,0);
start_ind = find(A);

%%Check for bad packets, put good ones into new data matrix
junk_count_high = 0;
junk_count_low = 0;
good_count = 0;
row = 1;
i = 1;
while(i<=length(start_ind)-1)
    % packet too long
    if (start_ind(i+1) - start_ind(i))>num_bytes
        junk_count_high = junk_count_high+1;
        % fill bad data rows with NaN
        for n = 1:num_bytes
            data(row,n) = NaN;
        end
        % put IMU name into proper spot in data matrix
        % if IMU name is off in packet, get right one
        if good_data(start_ind(i)+4)==0
            data(row,5) = good_data(start_ind(i)+4);
            data(row,6) = good_data(start_ind(i)+5);
```

```
        else
            data(row,5) = good_data(start_ind(i)+5);
            data(row,6) = good_data(start_ind(i)+6);
        end

    % packet too short
    elseif (start_ind(i+1) - start_ind(i))<num_bytes
        % check if there are multiple 126's in a single packet
        y = 2;
        while ((start_ind(i+y)-start_ind(i))<=num_bytes)
            % Packet is right length now
            if (start_ind(i+y)-start_ind(i))==num_bytes
                good_count = good_count+1;
                data(row,1) = good_data(start_ind(i));
                for n = 2:num_bytes
                    data(row,n) = good_data(start_ind(i)+n-1);
                end
                i = i+y;
                if (i+y)>length(start_ind)
                    break
                end
            % Packet still too short
            elseif (start_ind(i+y)-start_ind(i))<num_bytes
                y = y+1;
            % Packet bad
            else
                junk_count_low = junk_count_low+1;
                for n = 1:num_bytes
                    data(row,n) = NaN;
                end
                % put IMU name into proper spot in data matrix
                % if IMU name is off in packet, get right one
                if good_data(start_ind(i)+4)==0
                    data(row,5) = good_data(start_ind(i)+4);
                    data(row,6) = good_data(start_ind(i)+5);
                else
                    data(row,5) = good_data(start_ind(i)+5);
                    data(row,6) = good_data(start_ind(i)+6);
                end
             end
        end

    % good packet
    else
        good_count = good_count+1;
        data(row,1) = good_data(start_ind(i));
        for n = 2:num_bytes
            data(row,n) = good_data(start_ind(i)+n-1);
        end
    end
    row = row+1;
    i = i+1;
end
% final row count
row = row-1;
```

```
%% Check for rows of all zeros in data matrix and get rid of them
count=0;
for r=1:row
    if (data(r,1)==0)
        count = count+1;
    else
        data1(r-count,:) = data(r,:);
    end
end
row = length(data1);

%% Check for dropped packets, put NaN in as place holder for dropped packet
IMUname = zeros(it*length(data1),1);
v0 = zeros(it*length(data1),1);
v1 = zeros(it*length(data1),1);
v2 = zeros(it*length(data1),1);
% set number of dropped packets for each imu to zero (column 1=A, 2=B, etc.)
ndrop = zeros(nimu,1);

for r=1:nimu:row-(nimu-1)
    data_ind = r*it-(it-1);         % transform row number into data vector
index for v0, v1, and v2
    name_vec = zeros(nimu,1);       % initialize name vector (4x1)
    for x=0:nimu-1
        name_vec(x+1) = data1(r+x,5)*256 + data1(r+x,6);
    end
    % subtract IMU name from name_vec and use find to determine the number
    % of non-zero numbers in vector, if <=nimu-1, then the IMU name shows
    % up. if =nimu, then the IMU is missing (packet dropped)
    for x=1:nimu
        missing_imu(x) = length(find(name_vec-IMU(x)));
        % if missing_imu(1)=nimu, then there is no A in the name_vec (packet
was dropped)
        % if missing_imu(2)=nimu, then there is no B in the name_vec (packet
was dropped)
        % etc.
        if missing_imu(x)==nimu
            ndrop(x) = ndrop(x)+1;
            IMUname(data_ind) = IMU(x);
            v0(data_ind) = NaN;
            v1(data_ind) = NaN;
            v2(data_ind) = NaN;
            IMUname(data_ind+1) = IMUname(data_ind);
            v0(data_ind+1) = NaN;
            v1(data_ind+1) = NaN;
            v2(data_ind+1) = NaN;
            if (it==3)
                IMUname(data_ind+2) = IMUname(data_ind);
                v0(data_ind+2) = NaN;
                v1(data_ind+2) = NaN;
                v2(data_ind+2) = NaN;
            end
        end
    end
end
```

```
percent_drop  = 100*(sum(ndrop))/(junk_count_low + junk_count_high +
good_count);
%% Put voltage values into row vectors for analysis
x = 1;
for r=1:row
    if x>length(v0)
        break
    end
    % only put in values for good packets (dropped packets have NaN already)
    if (~isnan(v0(x)))
        % put values from data matrix into row vectors
        IMUname(x) = data1(r,5)*256 + data1(r,6);
        v0(x) = data1(r,12)*256 + data1(r,13);      % 256 * high_byte +
low_byte
        v1(x) = data1(r,14)*256 + data1(r,15);
        v2(x) = data1(r,16)*256 + data1(r,17);
        IMUname(x+1) = IMUname(x);
        v0(x+1) = data1(r,18)*256 + data1(r,19);    % 256 * high_byte +
low_byte
        v1(x+1) = data1(r,20)*256 + data1(r,21);
        v2(x+1) = data1(r,22)*256 + data1(r,23);
        if (it==3)
            IMUname(x+2) = IMUname(x);
            v0(x+2) = data1(r,24)*256 + data1(r,25);    % 256 * high_byte +
low_byte
            v1(x+2) = data1(r,26)*256 + data1(r,27);
            v2(x+2) = data1(r,28)*256 + data1(r,29);
        end
    end
    x = x+it;
end

%% Split up voltage data for different IMU's
% Each IMU corresponds to a column in each matrix
% counter for each imu column
for x=1:nimu
    a(x) = 1;
end
% split up data into matrices
for x=1:length(IMUname)
    for i=1:nimu
        if IMUname(x)==IMU(i)
            D0(a(i),i) = v0(x);
            D1(a(i),i) = v1(x);
            D2(a(i),i) = v2(x);
            a(i) = a(i)+1;
        end
    end
end

% For frontal IMUs (H, J), D0 in this code will be Ay and D1 will be Az
% Need to switch these since Az ==> Ax for frontal IMUs
if (nimu>=4)&&(set==2)
    Az = D1(:,4:nimu);
    D1(:,4:nimu) = D0(:,4:nimu);
    D0(:,4:nimu) = Az;
end
```

```
% gives mean value of each column of the matrices, excludes dropped or bad
packets
count0 = zeros(nimu,1);
sum0 = zeros(nimu,1);
count1 = zeros(nimu,1);
sum1 = zeros(nimu,1);
count2 = zeros(nimu,1);
sum2 = zeros(nimu,1);
for j=1:nimu
    for i=1:length(D0)
        if (~isnan(D0(i,j)))
            sum0(j) = sum0(j) + D0(i,j);
            count0(j) = count0(j) + 1;
        end
        if (~isnan(D1(i,j)))
            sum1(j) = sum1(j) + D1(i,j);
            count1(j) = count1(j) + 1;
        end
        if (~isnan(D2(i,j)))
            sum2(j) = sum2(j) + D2(i,j);
            count2(j) = count2(j) + 1;
        end
    end
end

D0avg = sum0./count0;
D1avg = sum1./count1;
D2avg = sum2./count2;
```