University of Technology, Sydney
Faculty of Engineering and Information Technology

Programming of an Inertial Measurement Unit

By

Jian Carlo Calaunan

Student Number: 11244252
Project Number: S14-142
Major: Electrical Engineering

Supervisor: Dr Steven Su

A 12 Credit Point Project submitted in partial fulfilment of the
requirement for the Degree of Bachelor of Engineering

18 June 2015

Statement of Originality

The work contained in this report, other than that specifically attributed to another source, is that of the author(s) and has not been previously submitted for assessment. I understand that, should this declaration be found to be false, disciplinary action could be taken and penalties imposed in accordance with University policy and rules.

Jian Carlo Calaunan

18/06/2015

Abstract

Inertial Measurement Unit (IMU)

Jian Carlo Calaunan

Autumn 2015

The Capstone project centers on a device comprised of both accelerometers and gyroscopes to reproduce the human vestibular system, which determines balance and spatial orientation. This Inertial Measurement Unit (IMU) is a fundamental component in this project, as a Graphical user Interface (GUI) will be programmed to utilize the device for its viability in Gait analysis using only its accelerometer readings.

The development of this interface is to be conducted on LabVIEW where a visual programming approach is required to build the Virtual Instrument (VI). This GUI will interface with the IMU in real-time to provide visual quantitative readings from the device, such that a Gait Analysis can be performed utilizing those readings. There will be no physical interface between the program and the device, as the transmission of data between the two will be wirelessly communicated via Bluetooth.

The instrument will be program as a typical pattern recognition system and evaluate gait patterns by using a K Means clustering algorithm.

The success of the project is beneficial to the community as it can be applied to the medical field where it can be utilized in providing feedback for artificial attachments known as a prosthesis and many other systems. This medical application is known as Gait Analysis, it typically uses the IMU's to assist people with abnormal movement and/or posture problems and fix these issues by providing remedial treatments.

Acknowledgement

I would like to acknowledge Steven Su for providing me with this great learning experience and opportunity for this Capstone project as I have a particular personal interest in these sort of medical applications. I appreciate his guidance and suggestions in helping complete this project.

My acknowledgements also extend to the UTS library as the textbooks and online educational access to Literature Reviews and Journals have provided me with a sufficient amount of knowledge to conduct this project.

1 TABLE OF CONTENTS

2 LIST OF FIGURES

3 LIST OF TABLES

4 LIST OF EQUATIONS

5 INTRODUCTION

The motivation behind undertaking this type of project is the development of a medical analysis instrument programmed on a software known as LabVIEW. The program will be used to conduct the viability of Gait analysis using the Inertial Measurement Unit's (IMU) accelerometer sensors and the software itself.

The literature review will investigate the methods and applications of Gait analysis, the theory of a pattern recognition system, its inner workings and the various type of models.

The research conducted will provide a deeper understanding into the projects scope of works here the primary goals of the project will be described.

The equipment essential to the project is listed, both hardware and software requirements as well as the associated costs of the project.

6 LITERATURE REVIEW

6.1 GAIT ANALYSIS

Gait analysis is the study of human movement, whereby the examination of specific bodily reactions for particular repetitive movements results in the capability of quantifying the human body's actions, which can be used for other applications.

The analysis of gait begins with deciding the type of movement to be performed by the body, then the analysis of the body is sectioned based upon which body parts perform repetitive movements for instance, the gait analysis for walking would consider two main areas separately, they are the movements of the arms and legs. The repetitive movement being performed is known as a gait pattern and is divided into two distinct phases, which are the Stance and Swing phase.

Gait analysis was primarily performed in laboratories that were specifically designed to analyse the body's movement through the use of video camera systems in combination with force platforms, these platforms determined the ground reaction forces that the body produced. However, with the improvement of technology, it has allowed portable sensors that can be securely attached to the human body to be used in gait analysis.

6.1.1 Technology used in Gait Analysis
With the improvement in technology in the recent years, the equipment to perform gait analysis has become cheaper, more compact, accurate and its versatility, such as how the data is collected and its ability to interact with other technology.

6.1.1.1 Sensors

6.1.1.1.1 Inertial Measurement Unit (IMU)
This device was primarily comprised of two fundamental sensors, an Accelerometer and a Gyroscope. The IMU can be integrated with other type of compact sensors that increases the performance such as it precision and stability, as well as extending the IMU's ability and applications. The sensors typically manufactured in combination with the sensors mentioned above, are the Magnetometer and the Pressure sensor.

6.1.1.1.1.1 Accelerometer

This sensor is attached on the leg to allow the measurement of the linear acceleration of the leg during gait.

Typically three accelerometers, known as multi-axis models are required, in order to measure the acceleration with respect to its reference axis and provide a 3D-model representation i.e. x, y and z-axis.

6.1.1.1.1.1.1 Types of Accelerometers

There are a number of various types of accelerometers that are have been developed for different purposes, only a few are applicable for gait analysis and have been narrowed down to piezoelectric, piezoresistive, and capacitive accelerometers (Tao et al. 2012). Of the three mentioned the two ideal sensor types are the piezoresistive and capacitive, as they provide greater stability and dual acceleration components i.e. magnitude and direction (Tao et al. 2012).

6.1.1.1.1.2 Gyroscope

A sensor that is fundamental in aeronautics as it allows the detection of pitch i.e. to rotate up/down, roll i.e. rotate left/right and yaw i.e. rotate counter-clockwise/clockwise. The application of this sensor in gait analysis allows the measurement of the angular velocity and the angle displacement of leg during gait.

Similar to the accelerometer sensor, this sensor provides the rotational data during gait and typically requires three sensors in measuring different axis' i.e. roll, pitch and yaw. This sensor is particularly useful in assessing the balance and posture of an individual.

6.1.1.1.1.2.1 Types of Gyroscopes

Gyroscopes based on other operating principles also exist, such as the electronic, microchip-packaged MEMS gyroscope devices typically found in consumer electronic devices, solid-state ring lasers, fibre-optic gyroscopes, and the extremely sensitive quantum gyroscope.

6.1.1.1.1.3 Magnetometer

The basis of this sensor operates on the principle of changing magnetic flux, in other words if a magnetic field is present then the tilt angle relative to the reference point

will change in the device. This is of particular use in gait analysis, as it detects when a change in orientation occurs.

6.1.1.1.1.3.1 Types of Magnetometer

There are two general types of magnetometers and they are either a scalar or vector type. The scalar type provides simply the magnitude of the magnetic field whilst the vector provides both magnitude and direction. The magnetoresistive sensor is ideal for gait analysis and falls under the vector type category.

6.1.1.1.1.4 Force Sensor

The strain gauge sensors otherwise known as load cells, are typically utilized for force sensors used in gait patterning as they are compact and relatively inexpensive to reproduce. It operates based upon electrical resistance due to twisting displacement from its original place. The sensor is typically used to detect the ground reaction forces exerted during gait and provides a 3D vector representation of the magnitude exerted.

6.1.1.2 Electromyography (EMG)

This technique measures the voltage potentials of impulses emitted during muscle contractions using either surface or wire electrodes. The electrical activity emitted by the skeletal muscles is recorded and evaluated to determine the nerve cell health of the muscles. Generally, surface electrodes are utilized for gait analysis, as they are less invasive and more widespread than its counterpart is.

6.1.2 Applications of Gait Analysis

With improvements occurring in the biomedical engineering field, gait analysis has become widespread particularly with the use of several sensor combinations to analyze gait patterns. The applications have become widespread influencing various aspects of life to order to assist and improve the body's mobility and functionality. The two main areas where gait analysis has been applied to are the sporting community and the medical field.

6.1.2.1 *Athletic Community*

Its application in the athletic community is to assist in improving athletic performance, this is achieved by recognizing the muscular faults in the gait patterns that are inefficient could be improved, it is also includes injury prevention by correcting improper form during the movement. Recent technology has provided the availability of portable, wireless and remote coaching of individuals whether they be competitive athletes, sporting enthusiasts or hobbyist to ordinary individuals. Devices such as 'Beast' and 'GymWatch' are comprised of sensors mentioned in section 6.1.1 that interface via Wi-Fi or typically through Bluetooth to store and display the data onto an application on a smart device such as a tablet or smartphone. The application allows for real-time analysis that can be viewed remotely by a professional for immediate feedback, the application may also provide suggestions on improvement based upon baseline measurements built into the program itself.

6.1.2.2 *Medical Field*

The medical field application of gait analysis allows for the improvement primarily in diagnostic checks of the body. The ambulatory method is the adaptation of gait for walking, this particular method is important to determine irregularities in their gait pattern and is frequently applied in the industry.

Due to the medical field having a wide variety of professions, the careers that benefit from accurate diagnosis in terms of human movement are Physicians. These individuals are generalized as specialist in restoring and maintaining human health. This profession allows for specialization into specific treatment areas such as surgeons, cardiologist, dermatologist, endocrinologist, gastroenterologist, etc. The specialist field of interest is known as orthopedics, as this field is the study the human musculoskeletal system, which are the essential parts of the body that allow movement i.e. bones, joints, ligaments, tendons, muscles and nerves. The Orthopedic physicians utilize their specialized knowledge to diagnose and provide treatment to irregularities occurring in the musculoskeletal system. There are numerous applications of gait analysis practiced by orthopedic physicians that have been tested and verified in their treatments. Joint replacement otherwise known as Arthroplasty, was used in combination with gait

analysis for monitoring the progress before and after treatment. This application was successful as it provided data that showed significant improvements overtime (Atallah et al. 2011, Aminian & Najafi 2004) in the patient's health. These types of treatments use the ambulatory method as the baseline in order to determine if the patient's progress is improving.

Gait analysis has also been utilized to detect diseases, specifically the ones that affect the motor functions of the body such as Parkinson's, which causes motor dysfunctions within the body. Recent studies on the use of gait analysis as an alternative measure of the severity of Parkinson's disease have been increasing. *Salarian et al.* (Salarian et al. 2004) performed gait measurement in patients with Parkinson's disease using a developed wearable sensor device and concluded that stride length is highly correlated with the severity of the disease. Similarly in section 6.1.2.1 to how sensors were tailored for athletic improvement, the manufacturing of sensors for patients diagnosed with this disease have been developed.

The sensors tailored for Parkinson's patients are also applicable to stroke patients, as they have impaired mobility and as a result are unable to move independently due to the incident. Ambulatory gait analysis is applied in the automatic identification of temporal gait parameters of post-stroke individuals to provide an assessment of the functional utilization of the affected lower extremity as part of the behavior enhancing feedback (Salarian et al. 2004, Saremi et al. 2006).

6.1.2.3 *Security*

The application of gait analysis extends to personal security based upon the personal characteristics of an individual, this is known as biometric security. This type of security provides a unique signature that cannot be reproduced by someone else. In terms of gait pattern, when a gait comparison between two individuals is closely observed and analyzed, the results will show there are slightly variations in the kinematic and kinetic patterns between the two individuals. This is due to several factors such as age, weight, height, etc. which affect the musculoskeletal system of the individual. To achieve a detailed gait analysis, a complex system comprised of multiple sensors would be required however due to the technological improvements as of late, a precision accelerometer sensor would be adequate enough to record a gait pattern (Mäntyjärvi et al. 2005, Gafurov, Helkala & Søndrol 2006).

The biometric gait security system uses a recognition algorithm to validate whether the attempted gait pattern is the same, otherwise if there are major discrepancies to the gait pattern stored in the database the authentication would obviously fail.

There are typically two implementations of gait pattern recognition, either vision or sensor-based. The vision-based recognition operates based on two classes, either model based or model free. Both models operate on statistical measurements, they only differ in what the measurements are referenced with respect to i.e. model based uses the models discernable dimensions, such as height and width whereas the latter uses the outline of the models body.

The operation of sensor-based recognition works primarily off sensory data from the multi-axis accelerometer (Mäntyjärvi et al. 2005, Gafurov, Snekkenes & Buvarp 2006). It can also be combined with a multi-axis gyroscope to provide a more distinct sample than with an accelerometer alone, allowing the balance and posture of the individual to be taken into account.

6.2 PATTERN RECOGNITION

The fundamental process of performing pattern recognition involves processing the input data and assigning a label to the data based upon its characteristics, which is determined by the pattern recognition system.

The processed label is just one of the many labels that help identify the input data, as it defines one of the many unique characteristics of the processed data. The label is utilized for comparisons of a separate data set to determine whether the new data set matches the processed data set.

The two common categories that the pattern recognition system defines a label are either classification or estimation, where the output label is dependent upon the vector input to the system.

6.2.1 Classification

This category is based upon a set of observation parameters that define a finite set of labels, with no sequential order, to the observed input data. The parameters are set prior to the analysis of a sample data set and typically have predefined initial classes otherwise known as a group of labels that have similar or identical characteristics.

6.2.2 Estimation

In contrast to the classification pattern recognition, this category is more quantitative as it has an infinite set of labels to assign. The estimation category is related to organizing numerical input data, into either exact values or if it falls within a range of values based upon the defined mapping.

6.2.3 Uncertainty

Classification errors typically occur when the input data being processed have similarities with two or more separate classes; the system typically assigns the label to the highest probable class. The classification category is dependent upon whether the label matches or not, whereas estimation is simplistic in its approach of assigning the labels to its appropriate or closest matching classes by comparing sequentially throughout all infinite classes until the best match or its closest approximate is found.

6.3 MODELS

The internals of a pattern recognition system can consist of one or a combination of models for classifying the input data for analytical comparisons, the standardized models are listed below.

6.3.1 Fixed Models

This model is simply comprised of closed-form equation/s that classify the outputs based upon the mathematical calculation. Prior knowledge is required in developing this particular model in order to formulate the correct mathematical equation.

This model is simplistic however does not cater to varying parameters as the calculation is based upon a known constant for that specific context, the next model addresses this issue.

6.3.2 Parametric Models

Data-driven models are similar to fixed models in the way they are structured; the key difference is that the equation takes several parameters to produce its output, thus a parametric model. It is far more flexible than the former model as it creates mappings dependent upon these parameters. These mappings once defined, are internally compared against each other in order to select the most viable mapping closest to the desired output.

6.3.3 Nonparametric / Data-driven Models

The model centers around handles large volumes of unknown data typically read from data acquisition systems. The model requires an analyzed data set to compare the new incoming data with the predefined data set.

6.3.4 Pre-processing Models

This particular model is used for complex situations where classifying the input data to the output labels is not possible when the quantity of data is of a larger magnitude than what a nonparametric model can process in an ideal time frame. Though the nonparametric model would be able to complete the task, it would take a sufficient amount of time and data to complete the pattern recognition.

Pre-processing takes into account several key factors to look out for; as a result, it decreases the period for each calculation to be completed.

Simply put this model takes a large magnitude of data inputs and screens it accordingly to the internal parameters, the outputs of this model is filtered data that is routed typically into a non-parametric model for labelling however it can also be applied to parametric models.

6.4 LEARNING ALGORITHM

The main algorithms of concern are the non-parametric models present in the Machine Learning Toolkit (MLT) (refer to Appendix D – Machine Learning Toolkit User Manual), as the context of this project will require the manipulation of continuous data being input into the system via the IMU.

6.4.1 Nonparametric

The following algorithms that fall under this category, will group data points based upon the training instructions and the algorithms methodology.

The nonparametric algorithms fall under two subcategories either unsupervised or supervised learning.

A supervised learning algorithm alters the parameters used to pattern the input data based upon the trained data. The unsupervised learning algorithm clusters the data with no specific order or pattern, the output data is essentially unlabeled.

6.4.1.1 *Multilayered Perceptron (MLP)/Backpropagation (BP)*

A form of neural network whose original purpose was to model neurological practices, the MLP achieves this objective by combining its architecture with the BP algorithm. This allows it to form nonlinear relationships between the incoming input data and the output labels formed by the algorithm.

In terms of just nonparametric modelling, MLP makes use of several tiers typically consisting of three layers as seen in Figure 1 but for patterns of greater complexity additional tiers, known as hidden layers are required. For an n number of inputs to the MLP's first tier, the input layer will consist of an n number of nodes that represent the weight of that input. The next tier otherwise known as the hidden layer, contains nodes where individually the values are the weighted sums of the input nodes due to a sigmoidal function (Kennedy 1997). This node summation process continues until the

output layer where the label is the weighted sums of the inputs.

Input layer

Hidden layer   Output layer

Input signal

Output signal

The MLP output is used by the BP algorithm to obtain the desired results; this is achieved over several iterations of tuning the weights of the MLP via a mean-squared error (MSE) function Equation 1 (Kennedy 1997).

$$E = \sum_{i=1}^{N_{train}} \left\{ \frac{1}{2} \sum_{j=1}^{J} (d_{ij} - y_{ij})^2 \right\}$$

Equation 1. MSE (Mean-squared Error) Function

where i and j are the indexes of N_{train} patterns of the training set and each output variable respectively. The trained output d_{ij} of j index output from i index pattern and y_{ij} is the calculated output from the model.

6.4.1.2 *Radial Basis Functions (RBF)*

Similar to MLP in mapping inputs and outputs based upon weighted sums, the function however uses radial Gaussians Equation 2 instead of the MSE function Equation 1 (Kennedy 1997).

$$y = \sum_{i=1}^{N} w_i e^{\left(-\frac{1}{2\sigma_i^2 h}\right)(X-Mean_i)^T(X-Mean_i)}$$

where *y* is the output of the algorithm, *N* are the number of radial Gaussian functions, *X* is the vector input, *i* is the i-th radial Gaussian function, $Mean_i$ is the location of the center, σ_i is the spread, *h* being the overlap factor, w_0 is the bias and w_i is the weighted sum of that specific radial Gaussian function (Kennedy 1997).

RBF is similarly trained as BP however, its processing time can be improved when combined with K Means Clustering, and the combination of the two minimizes the variation between the training patterns to the Gaussian centers. This optimization is achieved by tuning the weights of each Gaussian such that the MSE is minimized on each iteration.

6.4.1.3 *K Nearest Neighbors (KNN)*

KNN is simplistic in its operation in that it stores the input and output pairs of a training set in a database, the size of the stored database is highly dependent on the dimensionality of the training set. The database acts as a reference for new input patterns however despite the pre-processed training pattern the limiting factor of this algorithm is in the overall dimensionality of the training set, essentially the more patterns in the stored training set the longer the comparison process will take to examine through the training set.

KNN utilizes the Euclidean measure Equation 3 as a means of comparison between the patterns within the training to the input pattern; Equation 3 allows the input pattern to find its closest match in the training set, thus its nearest neighbor.

$$Dist(X,Y) = \sqrt{\sum_{i=1}^{D} (x_i - y_i)^2}$$

where the distance calculated is of *i* dimensions between the input pattern to the closest pattern within the training set.

6.4.1.4 K Means Clustering

K means is a widely utilized algorithm primarily due to its simplicity in assigning or organizing data into the nearest cluster. These clusters, otherwise known as classes are user dependent and thus influence the amount of data points per cluster.

These cluster centers are calculated based upon the mean distance of each individual data point assigned to it.

$$J(X,V) = \sum_{i=1}^{c} \sum_{x_k \in C_i} \|x_k - v_i\|^2$$

where C_i is the *i*th cluster, x_k is an element of set C_i, v_i is the current cluster center of C_i and $\|x_k - v_i\|$ is the distance between these two points (Vathy-Fogarassy & Abonyi 2013).

Once it is centered the cluster is split in half and the data points previously assigned to the prior cluster become members of the new clusters, where the new clusters are evaluated using the same process as the above i.e. the cluster finds its center by calculating the mean distance of the surrounding data points. After the original cluster is split, the process will iterate until the number of clusters specified by the user is met. The data points are re assigned based upon their distance to the nearest cluster.
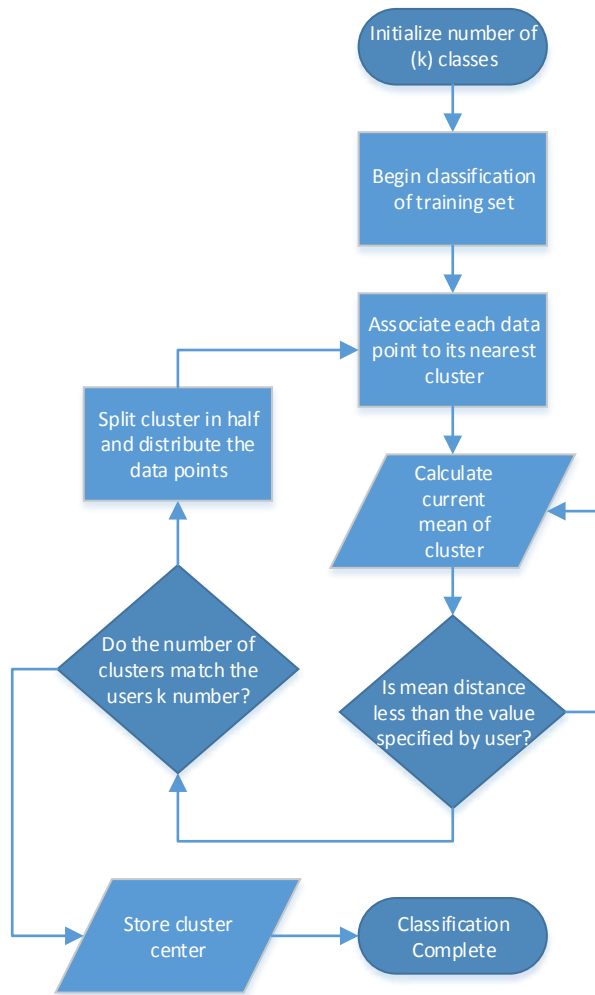
Figure 2. K Means Training Flow Chart (Kennedy 1997)

K Means Clustering typically creates a range of 2 to $\sqrt{K}$ clusters for data estimation purposes however for classification the number of clusters increases depending on the context (Kennedy 1997).

7 SCOPE OF WORKS

7.1 PROBLEM STATEMENT

The purpose of this Capstone project is to design and implement a working program that will allow the IMU to work in combination with the program for gait analysis. The IMU data outputs to be processed by the program are the accelerometer readings in the X, Y and Z-axis.

7.2 DELIVERABLE

The projects primary objective is the development of this gait analysis program. From the thorough research performed analyzing several literature reviews, it has been deduced that the program will operate as a pattern recognition system in order to achieve the objective.

This pattern recognition system will quantify the typical human movement patterns, such as walking and running. The system will store these quantified biomechanics as individual test data sets in the system, where it will use these data sets for future classification comparisons.

8 EQUIPMENT

8.1 HARDWARE

As this Capstone Projects mainly revolves around programming, there is very little hardware required to achieve the 7.1 Deliverable of this document. The main hardware requirements are the IMU itself and a computer loaded with the LabVIEW software.

8.1.1 Sensor board

In the context of this project, the IMU in its entirety allows the measurement and digital transmission of raw acceleration data to a measurement system. The IMU itself is comprised of two boards that each facilitate a specific operation, the sensor board provides the essential circuitry and regulators needed for the attached sensor to operate.

Figure 3. IMU 6DOF v4 Sensor Board

The sensor board has pins pre-soldered from the manufacturer, as seen in Figure 3 above; this is to allow connection with the controller board for added functionality.

8.1.1.1 Freescale MMMA7260Q three Axis Low-g Micro machined Accelerometer (1.5 – 6g)

The IMU is comprised of multiple modules; the most significant of them is the three axis accelerometers. For the simplicity and transparency of the project, only the accelerometers are to be utilized which is due to the acceleration changes occurring during the legs swing motion when movement occurs.

The reference directions of the X, Y and Z-axis are as depicted on Figure 4.
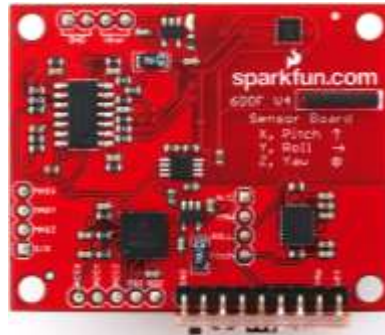
Figure 4. IMU 6DOF v4 Sensor Board Top View

8.1.2 Controller board

The controller board provides the user with the control of the sensor through the boards LPC2138 ARM7 Processor, the board also provides power for both platforms. Once connection is established, the memory in the processor can be manipulated with the WinARM software, provided by SparkFun, to alter the code for private or commercial use.

Figure 5. IMU 6DOF v4 Controller Board

On the underside of the board is where the built-in Bluetooth module is positioned, this module allows for wireless communication with any computer provided the appropriate adapter is present in pairing with the device.

8.1.2.1 Roving Networks Bluetooth Module

This module comes built-in to the IMU from the manufacturer; the additional module provides a wireless communication alternative method. The Bluetooth module is essential to this project as data samples from the IMU will be sent whilst on the move, due to the fact that it is to be attached to a moving ligament for gait analysis.

Figure 6. IMU 6DOF v4 Controller Board Back View showing Bluetooth Module

8.1.3 Bluetooth 2.0 mini USB Adapter

A minimum of a class 1 Bluetooth v2.0 adapter is required for Bluetooth communication to function correctly as recommended by SparkFun for this particular Bluetooth module as seen on their website (refer to http://www.sparkfun.com/products/9434).

The limitation of the adapter is reportedly a maximum of 100m with only one possible active connection.

Figure 7. Bluetooth 2.0 mini USB Adapter

8.2 SOFTWARE

It is crucial for the computer to have the appropriate hardware requirements to run and have compatibility with the following software.

8.2.1 Terminal Program

The IMU Bluetooth connection to the computer is made as a serial communications port is created, commonly seen in Device Manager as COM1, COM2, etc. The software required to communicate and access the data transmitted through this port is a terminal program.

Once a Bluetooth connection between the computer and the IMU is established in the computers Device Manager, a terminal program is initially required to communicate with the device to setup and ensure its operating correctly as specified by the IMU data sheet (refer to Appendix B – 6 Degrees Of Freedom v4 Datasheet).

8.2.1.1 PuTTy

Under the supervisors and another academics' use of the IMU, this particular software is picked amongst the other notable emulator terminal programs available on the Internet, primarily for its portability and cost effectiveness.
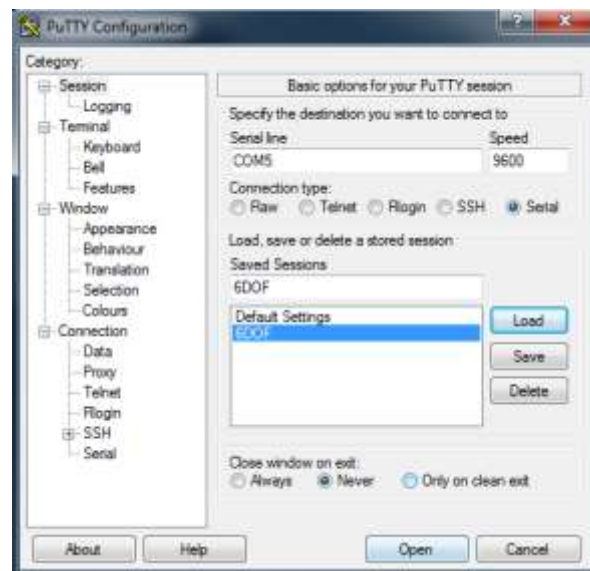

Figure 8. Putty Interface

8.2.2 NI LabVIEW (National Instruments Laboratory Virtual Instrument Engineering Workbench)

Software designed by National Instruments that have designed it specifically towards engineers and scientists that spend time on any measurement or control system.

The software allows for the programming of new instrument programs known as Virtual Instruments, it's also utilized for the improvement and/or adaptation of existing Virtual Instruments for private, commercial or educational purposes.

This is the primary software to be utilized throughout this project; the software will be programmed for data collection and manipulation to meet the objectives of this project, which is for gait analysis.
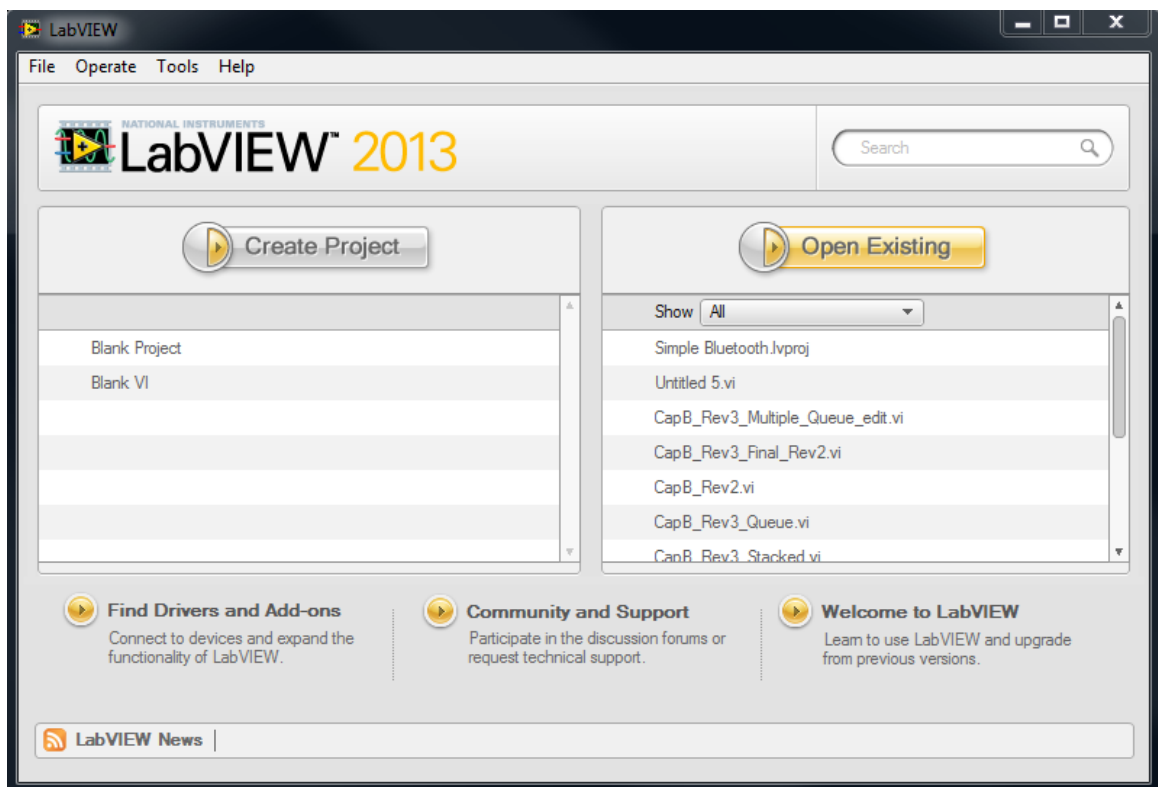

Figure 9. NI LabVIEW 2013

8.2.3 NI MAX (National Instruments Measurement & Automation Explorer)

The explorer allows the configuration of all physical devices connected to LabVIEW or utilized by its Virtual Instruments. The devices capable of integrating with LabVIEW are specific to those provided by National Instruments that are:

- CAN (Controller Area Network)
- DAQ (Data Acquisition)
- FieldPoint
- GPIB (General Purpose Interface Bus)
- IMAQ
- Motion

- IVI (Interchangeable Virtual Instruments)
- Modular Instruments
- NI Switch Executive
- VI (Virtual Instrument) Logger
- VISA (Virtual Instrument Software Architecture)

The software is quite sophisticated, it is not just limited to customizing the devices name and changing basic or advanced port settings, but also provides diagnostic checks, viewing of the devices attributes, write and reading privileges to connected devices.

This particular software comes included with the installation of LabVIEW and replaces the PuTTy software as it was utilized late in the project when troubleshooting of the virtual instrument was required due to errors occurring due to the device.

Figure 10. NI MAX (National Instruments Measurement & Automation Explorer) Program

9 COST

A minimal amount was invested on the projects overall costs, as the majority of the equipment was provided by the University of Technology, Sydney. The software and hardware requirements for the project are readily available in the Engineering Building computer laboratories.

The IMU was provided by the Capstone Supervisor, however should the unit be damaged, the cost of sourcing another identical unit would amount to the costs seen in Table 1 below.

Product id	Component Name	Cost ($AUD)
SEN – 08454	SparkFun IMU 6DOF v4 with Bluetooth Capability (RN-41)	$ 449.95
SEN – 08726	SparkFun Standalone Sensor Board	$ 349.95
SEN – 08727	SparkFun Standalone Controller Board, built-in Bluetooth (RN – 41)	$ 99.95

Table 1. Estimated Costs for Provisional Equipment

The main expenses of the project were spent on its ongoing costs; in particular, its power source to ensure the device was always in working order.

Component Name	Cost (AUD)
Energizer MAX AA Alkaline Battery (20 Pack)	$ 21.98

Table 2. Ongoing expenses

A single investment for the IMU platform and a Velcro hook strap was made for the continuation of the project. The platform was purchased prior to the start of this project and was adjusted for the IMUs' placement.

An elastic Velcro hook loop strap was essential to the project, as it acts to harness the IMU and its platform to the subject's leg, due to its elasticity it can cater to a wide range of test candidates with varying leg sizes.

Component Name	Cost (AUD)
Bluetooth mini-USB adapter	$ 10.95
Elastic Velcro hook loop strap	$ 5.00
Platform	$ 7.00

Table 3.Cost of Components Essential to Project

10 EXPERIMENTAL SETUP

10.1 EQUIPMENT SETUP

For the computer to interact with the device the Bluetooth Adapter Module must be plugged into the computer, afterwards the device is switched on. The device will be detected by the computer as seen in Figure 11 and is added via the Devices and Printers in Control Panel.

If communications with the device are successful, a typical Bluetooth pairing request will be visible on the screen, entering the pairing code "*1234*" will complete the connection and a serial com port number will be assigned to the device as seen in Figure 12.
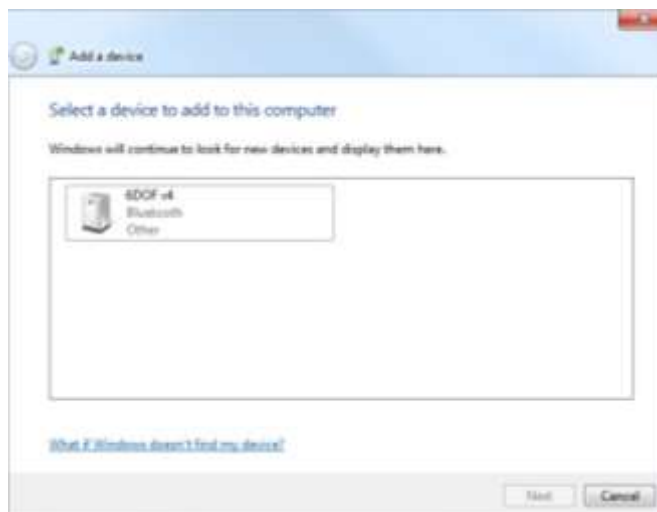
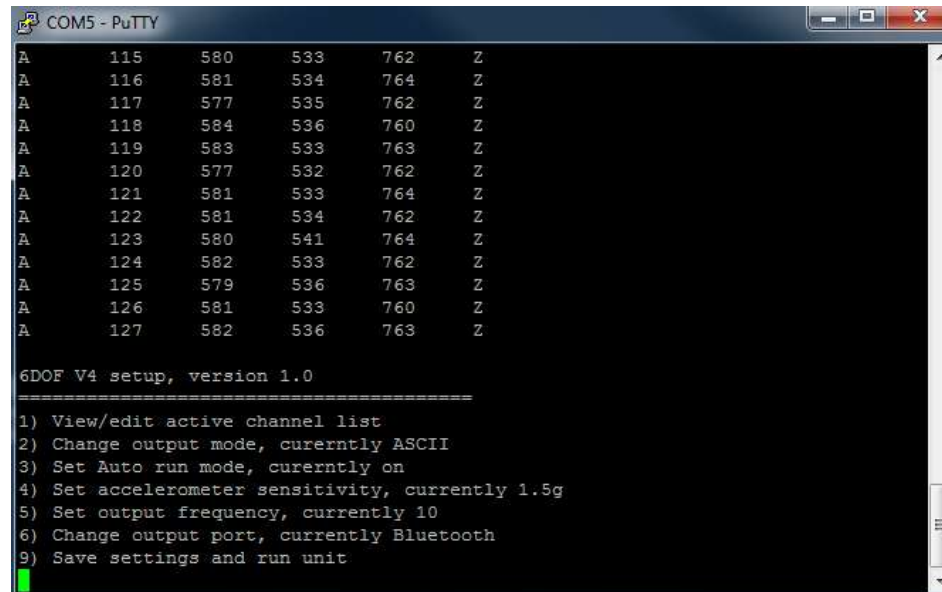*Figure 11. Add Device via Control Panel*

Figure 12. Device Manager Serial Port Settings

The next step involves changing the baud rate of the device to the specified 115,200 baud (Appendix B – 6 Degrees Of Freedom v4 Datasheet) then the initialization of the either IMU thru the terminal program PuTTy or NI MAX.

10.1.1 Terminal Program Initialize

Using the terminal program and the COM port number provided in Figure 12, entering the appropriate input parameters as seen in Figure 8, should initialize the IMU device such that Figure 13 is seen. The initialization is a success if the tri-color LED is illuminated with all three colors red, blue and green lighting simultaneously as seen in Figure 14.

Figure 13. PuTTy Interface - IMU Configuration Menu

Figure 14. IMU Idle State (White LED)

The device is configured such that the only active channel lists being output by the device are the X, Y and Z-axis accelerometer readings.

To enable the sampling or recording of accelerometer readings, Auto run mode must be configured to be On and once setup like in Figure 13, sampling can begin by entering '*9*'.

Exiting the terminal program is required in order for LabVIEW to gain access to the device.

10.1.2 NI MAX Initialize

Using the NI MAX program, the device can be found from the drop down selection in Devices and Interfaces as seen in Figure 15.

The IMU device can be renamed from its COM# name to a unique name for easier identification.

The initialization is much simpler than terminal and only requires opening the visa test panel as indicated in Figure 15.
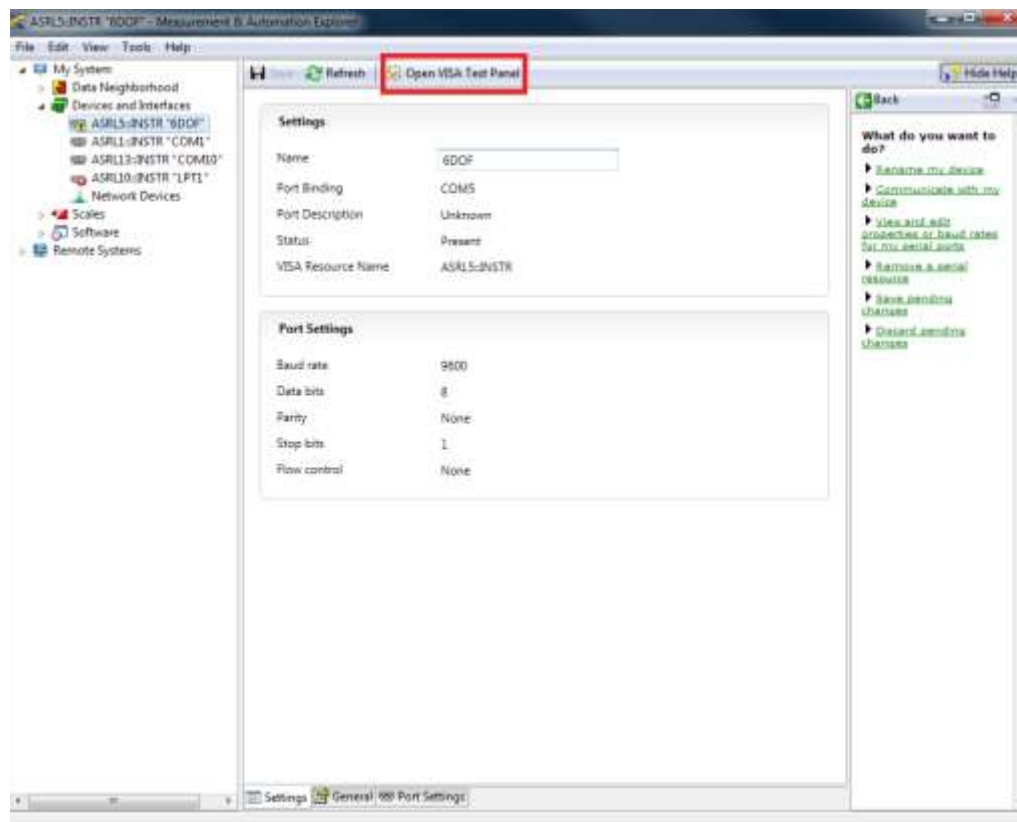
Figure 15. NI MAX Initialization

Upon clicking the Open VISA Test Panel and selecting the Input/Output tab as seen in Figure 16 will complete the initialization process via NI MAX. Closing the NI MAX program finishes the procedure.
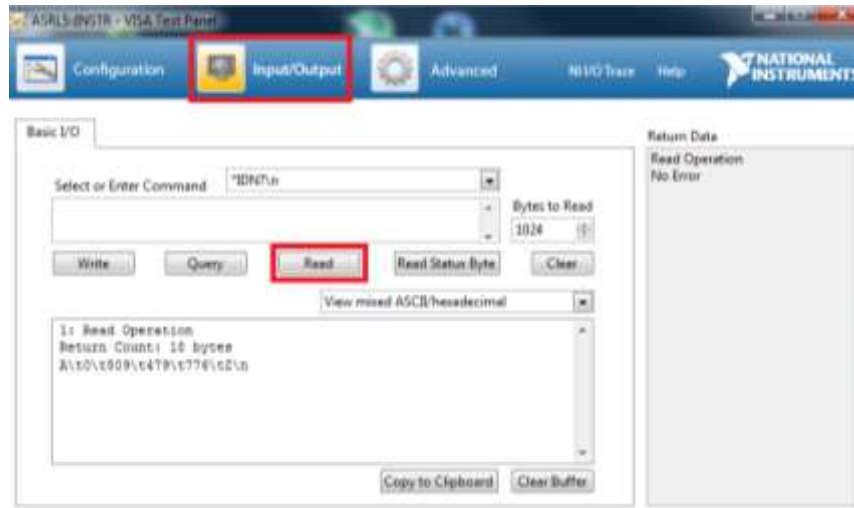
10.1.3 Auto run Mode

When Auto run mode is enabled and sampling begins the lights on the device will begin to alternate from red, blue to green as seen in Figure 17, Figure 18 and Figure 19 below.

Entering '9' in the terminal program or selecting 'Read' in the Open VISA Test Panel Input/Output tab as seen in Figure 16 will activate this mode on the device.
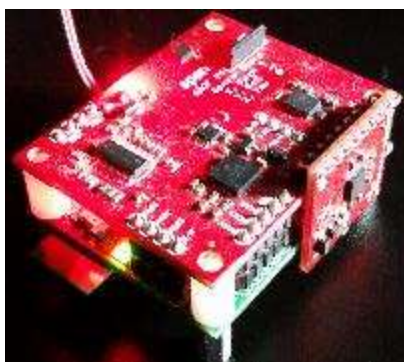
Figure 17. IMU Autorun Mode (Green LED)

Figure 18. IMU Autorun Mode (Red LED)
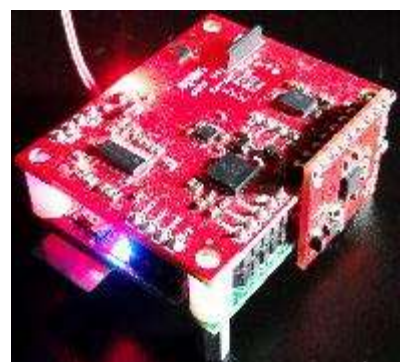
Figure 19. IMU Autorun Mode (Blue LED)

10.1.4 Summary

A summary of the startup procedure for the experiment is as follows:

1. Insert Bluetooth Adapter Module

2. Turn On IMU

3. Add Device via Control Panel

4. Enter Pairing Code 1234

    a. If successful, the tri-colour LED will be illuminating on the device (refer to figure), otherwise turn off and repeat Step 2.

5. Take note of COM port number

6. Initialize device thru:

    a. Terminal

    b. NI MAX

7. Close Terminal or NI MAX

8. Open LabVIEW VI

11 DESIGN

11.1 IMU SETUP DIAGRAM
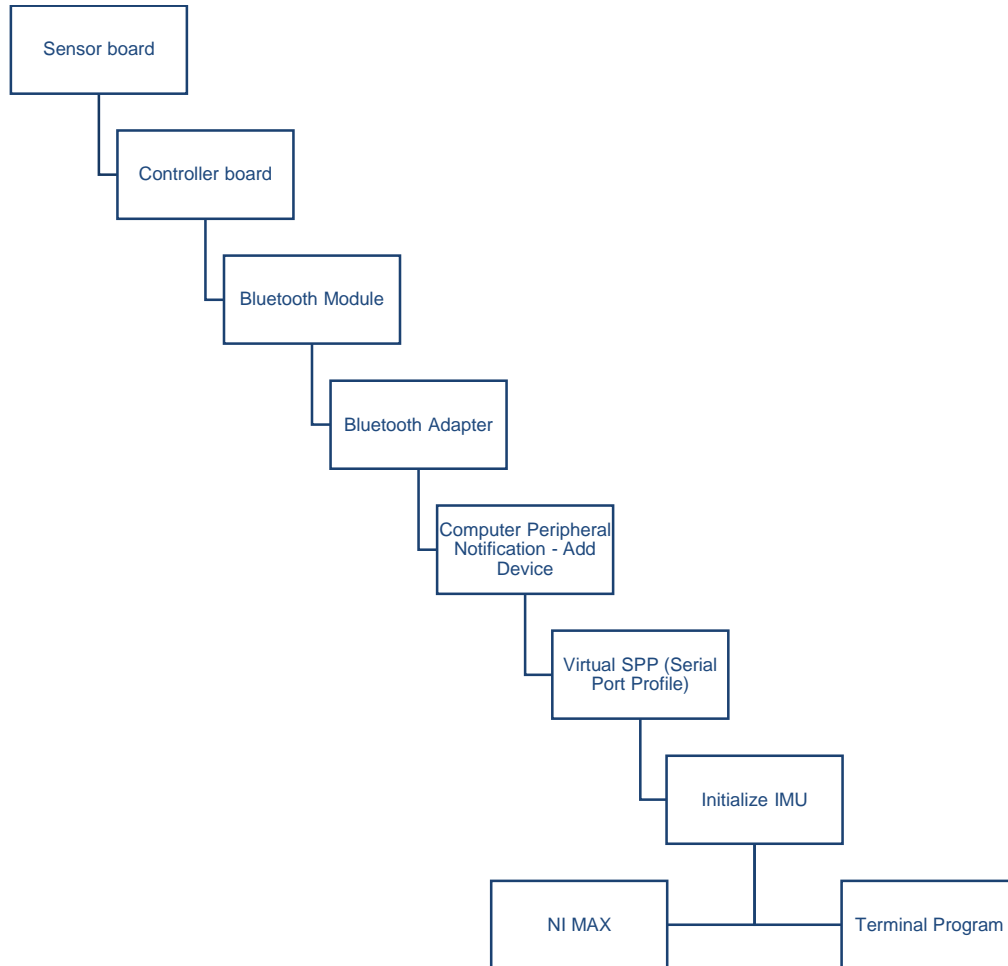
Figure 20. IMU Connection Diagram

The figure above shows the sequential data flow process required to read the raw outputs of the device. After the creation of the Virtual SPP, it becomes necessary to initialize the device to ensure communications with the device are functional and to wake up the device from sleep mode, should it become inactive after a length of time.

The initialization stage is important as the device can only have one active session open, therefore once the initialization is complete either NI MAX or the terminal program must be closed prior to running the LabVIEW VI.

11.1.1 Pattern Recognition System
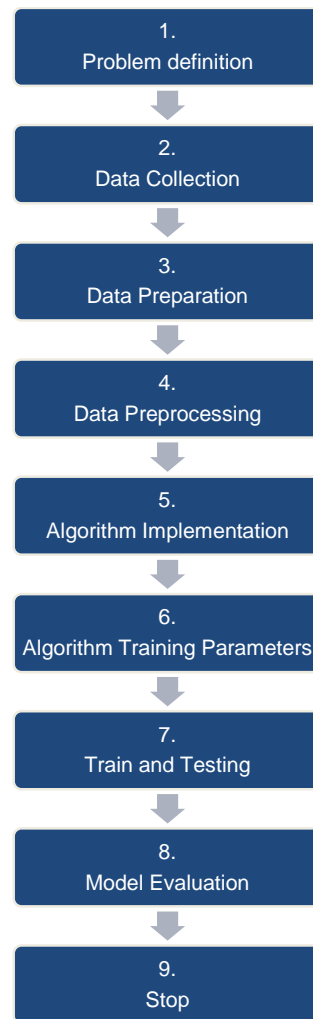
Figure 21. Pattern Recognition System Development Process (Kennedy 1997)

11.1.1.1 Problem Definition Stage

This stage involves evaluating the problem, in this case the pattern to be observed and data mined or modeled for recognition purposes. Once the evaluation is complete, the model/s that provide the best classifications are developed.

The data to be analyzed are the accelerometer readings output via Bluetooth to the computer and displayed on a terminal program as indicated on the IMU's datasheet (refer to Appendix B – 6 Degrees Of Freedom v4 Datasheet). Therefore, a data driven model will be suitable for this program, as it will be required to handle the continuous flow of data from the IMU.

The device requires initialization to ensure the connection is established and communication between the computer and the device is working, this state will be indicated on the device by a white LED when achieved. Accessing the device thru either the terminal program or VI MAX will accomplish this process. No sign of a white LED light indicates the device is still in an IDLE state.

11.1.1.2 Data Collection

The collection stage is straightforward and determines how the data is to be collected from the IMU by the program. As the IMU datasheet specifies the outputs can be either binary or in ASCII format, here the latter is chosen as LabVIEW interprets ASCII as strings.

The string format output is of the form "*A 0 123 123 123 Z*" where the spaces in between are tab delimiters, A and Z denote the start and end of this particular sample respectively. The first integer denotes the sample count, then X, Y and Z-axis accelerometer readings follow after in that order evident in Figure 16.

After the IMU has been setup as seen in Figure 20. IMU Connection Diagram the collection of data can proceed in LabVIEW. The sampling of data is indicated by the short flash of the tri-color LED toggling between red, green and blue (refer to section 10.1.3).

Using the built in LabVIEW, instrument input/output function known as VISA (Virtual Instrument Software Architecture), can communication access and control of the IMU be possible.

The main LabVIEW functions of concern within the VISA category are:

- Configure Serial Port: Required for the visual display and detection of the IMU COM port.

- Open: Once the above is configured then a singular session can be opened with the IMU.

- Close: Closes the session, thus terminating LabVIEW's access over the device.

- Write: Allows the user to send a string to the IMU, similar to the operation of the PuTTy except without a visual display.

- Read: VISA Read allows a string of x number of bytes to be read from the device.

The setup of the VI as seen in both Figure 26 and Figure 27 is comprised of three structures, the outer structure is the While loop used to keep the VI running until the user wishes to stop sampling. The nested structure inside the While loop is a Case Structure that runs a certain operation depending upon the conditions, in this case the Read On button. Inside this structure is another Case Structure, this structure makes use of both True and False Cases based upon the bytes read.

The Queue needs to be initialized, by using the Obtain Queue function and wiring in the expected data type to be queued i.e. Strings.

The Queues functions fall under the Queue Operations category, the main functions utilized within the context of this project are:

- Obtain Queue: Allows the creation of a Queue that handles a specific type of data otherwise known as an element.

- Enqueue Element: This adds the data of the element wired to the Queue.

- Dequeue Element: Typically wired to another section of the block diagram, this function removes the data from the queue.

- Release Queue: Similar operation to the Close Visa function, it ends the Queue.

Once the collection of one sample of data is read from the device, it is added to a Queue using the Enqueue Element function as seen in Figure 26, which allows the data is to be processed in a separate loop. The other case Figure 27 writes to the device to enable the Auto Run mode.

11.1.1.3 Data Preparation

The structure for this stage is the same as in the previous stage seen in Figure 28, this While loop structure will terminate as the user opts to stop sampling.

The Dequeue function extracts the raw data from the queue, this raw data is of a string format as mentioned previously; it is unusable and cannot be processed by numerical functions due to data incompatibility. The data format conversion to an integer or double is achieved using the Scan From String Function as seen in Figure 28 of

Appendix A – LabVIEW Virtual Instrument, once converted the data is bundled into a cluster using the Bundle function and is wired into a separate queue that specifically holds the new data type. This queue requires a separate Obtain Queue function to be initialized by Cluster constant consisting of a string, integer, integer, integer and a string in that exact order.

The Cluster functions can be found in the Cluster, Class, & Variant category and are useful for handling multiple types of different data types. The only functions utilized are:

- Bundle: Groups together various data types to create the cluster data type

- Unbundle: Converts the cluster data type back into its individual elements.

The output error of the Dequeue and Scan from String function are merged together, wired to the Clear Error function, and fed back into their respective inputs.

11.1.1.4 Data Preprocessing

The preprocessing of the data occurs in the same structure of the algorithm however this step is completed quickly. This step only involves using the prepared data and assigning the readings and the sample count into arrays as seen in Figure 29. The structure of this stage is a While loop with the same terminating parameters as the previous two.

The Dequeue function takes the data out of the queue and unbundled, the only data types of concern are the integer values of the accelerometer readings. Three individual 1D arrays are built using these readings, each reading is associated with the sample count at which it was recorded, and the While loop iteration count is used as the sample count. These array outputs are built into another array of two dimensions, the 2D array is the previous iterations output. The shift register allows new data to be continuously built on top of the previous iterations loop.

A user input on the VI Front Panel is created for the change in testing environment, using the average of the stationary Z reading from the trained pattern as a reference, the Z-axis reading prior to running the algorithm will need to be recorded to complete the preprocessing stage.

11.1.1.5 Algorithm Implementation

The MLT algorithm begins its calculation provided that 2D array is wired into its inputs. The algorithm outputs the classification labels and its centroids ready to be plotted onto the MLT 2D Visualizer. The data samples are rewired into the Visualizer as the labels coincide with the output labels of the algorithm.

11.1.1.6 Algorithm Training Parameters

Three numeric user inputs are created in the block diagram, these inputs are used to represent the number of classes/clusters.

11.1.1.7 Train and Testing of Model

At this stage in the process, the algorithm is using the inputs that have been preprocessed and are wired into the K Means function where it will create the mappings of the trained models for each individual axis.

The testing and training of the model was immediately performed once for three distinct gait i.e. stationary/standing, walking and running patterns were collected and saved to a spreadsheet file. These files are read back into separate VI (Figure 30), the data is prepared then preprocessed before the final phase can begin. The data being collected from the spreadsheets are read using a Read CSV Sub VI (Figure 35) and are wired into another Sub VI (Figure 36. Next Sample Sub VI), this VI filters data based upon the index input and only displays array indices before that index.

As indicated in Figure 30 the above Sub VI's prepare and process the data where it is wired into Stacked Sequence Structure, this structure reduces clutter and saves space, inside the structure a Fixed Array Sub VI permits the size of the array to only be of similar sizes. The test and trained data sets must be evaluated with the identical data counts otherwise the validity index tool outputs N/A. The outer loop structure is a For loop ensured to run once, with the most outer loop being a Case structure.

After the Sub VI has processed the array to be of correct size, the same wiring procedure used in 11.1.1.4 and 11.1.1.5 as seen in Figure 30.

11.1.1.8 Model Evaluation

The evaluation is conducted on a separate VI (Figure 30), the final phase involves using the K Means VI output class/cluster labels and wiring the test and trained data into the MLT Rand Index VI, an indicator is created to display the accuracy of the two data sets.

The indicators for the MLT 2D Plot VI's are created and organized into Tabs on the Front Panel according to the gait, this includes the Rand Index indicators (Figure 34).

The accuracy of all three axis are compared if they're above the minimum matching threshold, if all true for that specific case then the Stacked Structure of that loop writes to a string indicator.

11.2 Do a decision logic program flow
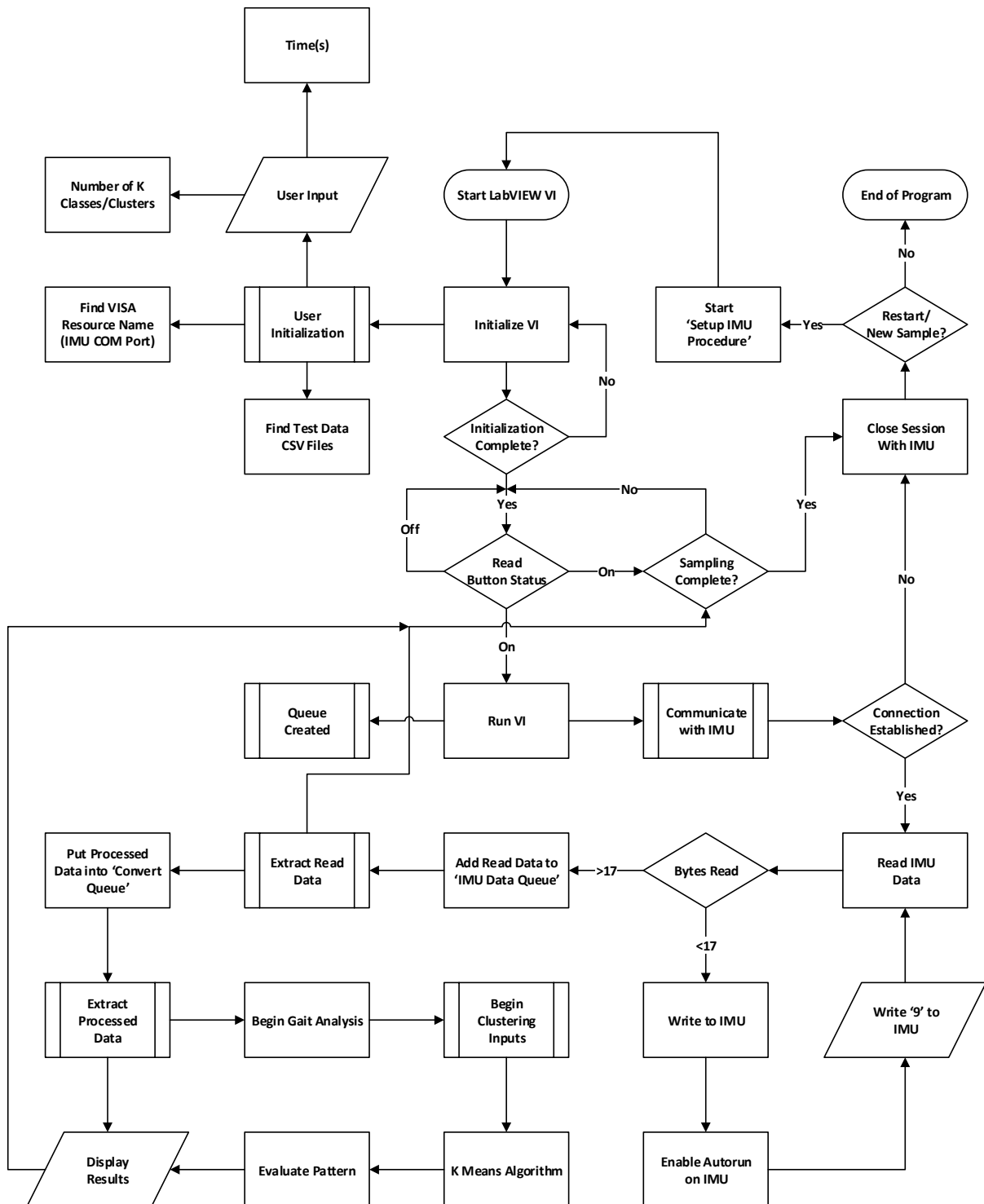
Figure 22. LabVIEW Logic Flow Diagram

12 FINDINGS

An unsupervised learning algorithm was chosen to classify the training set of IMU accelerometer axis readings into K number of clusters. The K Means clustering algorithm is implemented due to its wide applications in other research areas, its simplicity and in particular its memory usage and training time in comparison to KNN, the other unsupervised learning algorithm.

Similar to how the serial read and write example (Figure 23), a typical K means clustering example was discovered and analyzed to suit the needs of this project (refer to section 2.1.1 of

Appendix D – Machine Learning Toolkit User Manual).

The initial design requirements of the VI had to be able to communicate with the device, using prior knowledge and experience of this software led to the investigation of the NI Example Finder, it was here an example of a VI was found that met the initial requirements.

LabVIEW provides several base development VI's as well as examples for private and commercial development, Figure 23 is one of the many examples built in to the software. The modification of this VI example led to the creation of Figure 26 of

Appendix A – LabVIEW Virtual Instrument.
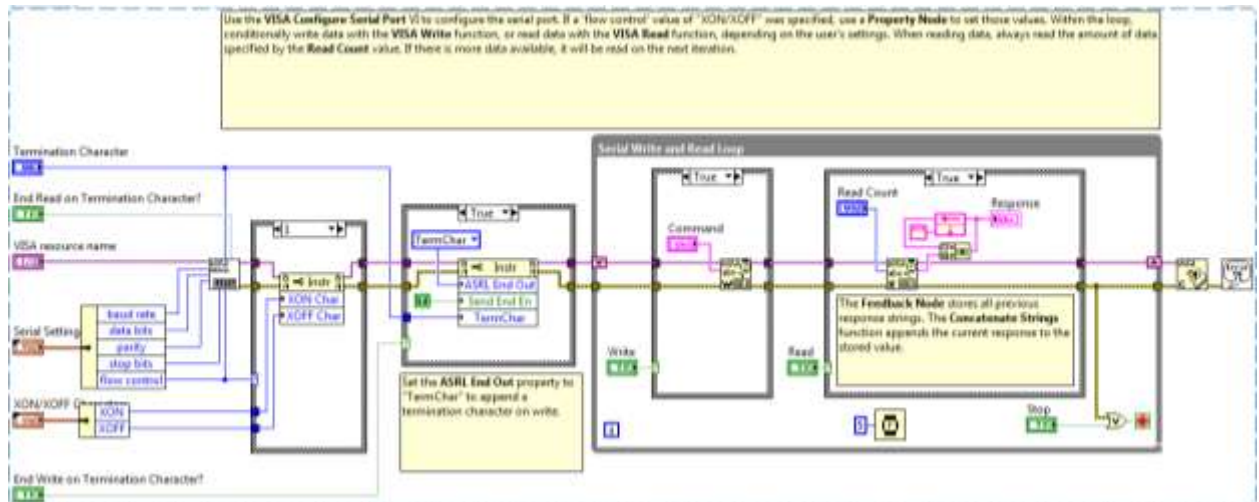
Figure 23. LabVIEW Continuous Serial Read and Write Example

In the first few prototypes the amount of bytes to be read were unclear thus a Property Node was used to detect the Serial Settings of that COM port, as seen in Figure 24 enclosed in the red box.

Figure 24. LabVIEW VI Early Prototype before Producer/Consumer Method

Another issue arose as to how to sample a gait pattern, after several observations of each specific the movement it was deduced that the amount of data to collect is dependent upon how quick the movement is performed i.e. the gait, as each individual has a unique body. It is predicted that the running gait pattern will have less samples, due to the action being

performed quicker than walking. The baseline will be assumed a stationary position that mimics both standing and sitting.

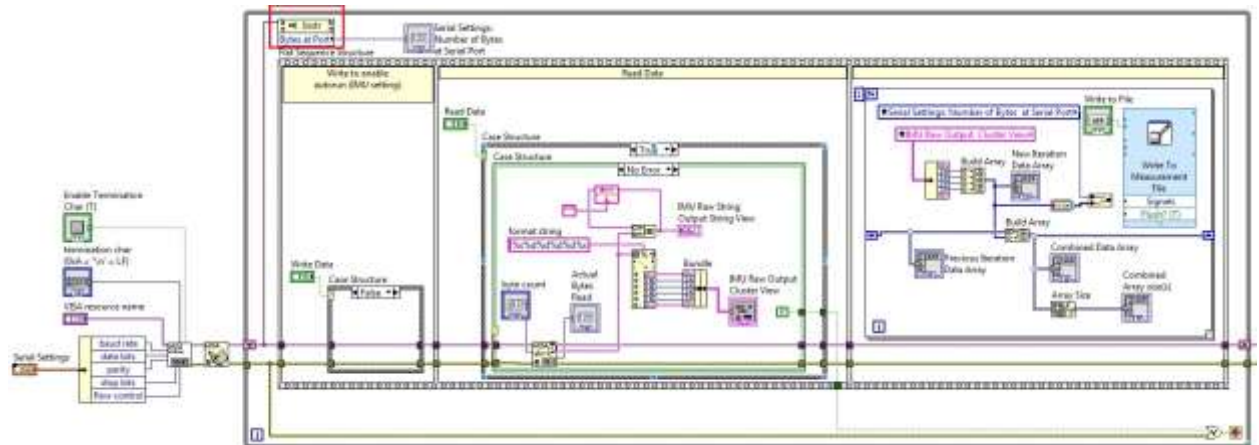As previously mentioned, various prototypes were developed the few were aimed at determining the manipulation of data read from the device. Using the textbooks provided by UTS (Bitter, Mohiuddin & Nawrocki 2006, Essick 2009, Kring & Travis 2006, Ritter 2002) to learn more about string manipulation, the Scan From String function was among one of the many tested string methods that succeeded, Figure 40 depicts the other two string manipulation methods that did not operate as intended. The Scan From String function was simple and easy to use thus was chosen out of the other methods.

The preparation phase with reference to Figure 24, involved the data collected being wired directly to the Scan From String function which converts the input string to the formatted output. The format string used to convert the string input required knowledge of the IMU output format, as indicated in Figure 13, Figure 16 and on the data sheet (refer to Appendix B − 6 Degrees Of Freedom v4 Datasheet). The readings begin with the character A and ends with Z, the numbers inside of this denote the sample count, X, Y and Z-axis accelerometer readings in that order. The operator % followed by either one of the two conversion codes d or s, decimal and string respectively, allows the single string to be separated into individual outputs.

A revision later, queues were introduced to tackle an error that caused the IMU stop its Auto run mode and display the sampling error "*Too much data. Please lower your sample frequency*". As the IMU was running at its lowest frequency of 10Hz, this fault was due to the computer, since it was not running optimally as the data is not being processed fast enough. This error also resulted in the data format of the Scan from String function to be incorrect and thus an error code "*LabVIEW: Scan failed. The input string does not contain data in the expected format.*" where the expected string was "*A 0 123 123 123 Z*". The sampling error above was displayed instead of the expected string thus leading to this Scan From String error.

The introduction of queues minimized the data pileup error from occurring however, it was deduced that the transmission of data was too slow as the problem kept occurring. This particular error had affected the projects progress greatly, as a last resort the problem was overcome by ignoring the error and re-enabling the Auto-run mode on the IMU achieved using Case Structures. Troubleshooting with LabVIEW led to no new results therefore a different approach was taken, troubleshooting via NI MAX led to the discovery that 18 bytes represented the first few outputs of the IMU. Below is a sample of data read from the IMU via NI MAX:

"*1: Read Operation, Return Count: 18 bytes: A\t3\t513\t500\t774\tZ\n*"

where each ASCII letter represented a byte and similarly with the delimiters "*\t*" and "*\n*" all sum up to 18 bytes. Thus, it was deduced that the maximum number of bytes to be read would be 24 bytes, this byte calculation accounts for four digit readings from the sample and accelerometer readings should the IMU be required to sample for that lengthy duration.

The Queue function is provided in the base development system of LabVIEW and can be found in the Queue Operations Functions, it is typically utilized to allow data availability to other functions whilst maintaining the programs synchronicity.

The purpose of producing queues is to isolate the reading and manipulation of data, this idea is modeled upon the Producer/Consumer Design Pattern template provided by the LabVIEW software.
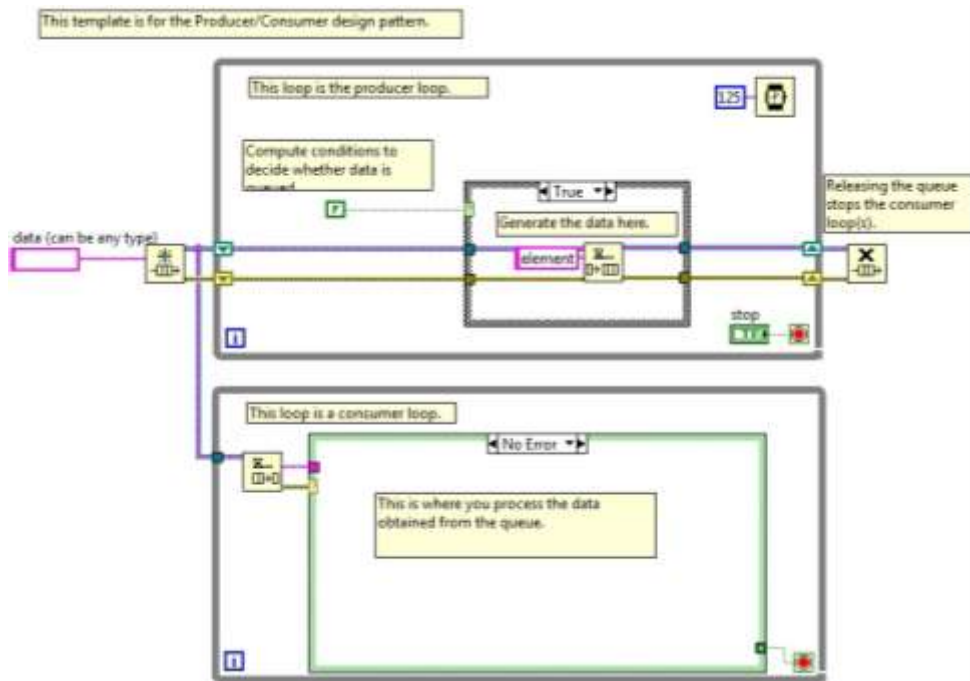
Figure 25. LabVIEW Producer/Consumer Data Pattern Template

The consumer loop seen in Figure 25 is where the data collected in the producer loop is extracted from the queue using a Dequeue Element function.

A Case Structure was introduced to the inside of the Data Collection Phase loop and a logical numerical comparison was used to determine whether the data is to be processed into queue (Figure 26) or Auto-run mode is to be enabled (Figure 27). The logic numerical comparison was based upon the discovery of the number of bytes read from a typical sample string output. This solution fixed the sampling error however the Scan From String error did not have any logical means of being fixed should the event occur, thus it was wired that the error to be reset all the time as seen in Figure 28.

An additional preprocess was incorporated to compensate for the varying Z-axis, this is due to gravity affecting readings on the Z-axis. This offset is introduced, as the test data was attained at specific height, therefore should the testing environment change the new Z-axis readings would be incorrect thus leading to skewed results. The X and Y-axis readings remain the same as the IMU is placed in such a way that the magnitudes of the readings are dependent upon the swinging motion produced by the test subject's limb.

Due to the errors in the previous stages of data collection and preparation, it was thought that the sample counts might be inconsistent thus the sample count readings were ignored and the While loop iterations were used as the sample count instead. This is possible as every iteration a single reading is processed thus one sample count.

The analysis of the K Means clustering example (refer to section 2.1.1 of

Appendix D – Machine Learning Toolkit User Manual) led to the discovery that the training parameters of the K means clustering are simple, the user is required to enter a K number of classes/clusters to be displayed on the graph. The k number of class/cluster indicators for each axis were not wired but rather were referenced by Local Variables found in the Structures category, essentially this allowed the same indicator to alter all three gait parameters simultaneously as seen in Figure 30, Figure 31 and Figure 32. It reduced the overall wiring and clutter on both the VI Front Panel and Block Diagram.

The three Sub VI's mentioned in section 11.1.1.7 were developed alongside the implementation of this system, they were originally apart of the main system however have been compartmentalized to reduce space and clutter in the programs block diagram. This was made possible as these particular functions were to be repeated for each time a pattern had to be read from file. The idea of compartmentalizing came from the observation of the internal operations of the MLT VI's and including the various prebuilt LabVIEW functions.

The process of data being extracted from external files and built into arrays produced different orientations to when they were written to file from the program. This orientation was not the correct format and was easily remedied; the solution was to transpose the built array essentially flipping the orientation of the data.

The initial evaluation between the test and trained data sets were not computed, the simple explanation was that the arrays of both sets of data varied, thus the development of a separate VI with the sole purpose of producing a fixed array size began. This separate array was to input the new readings into the last element of the array whilst deleting the first element of the array to ensure the size remained the same. The application of this array to the separate evaluation VI allowed the evaluation techniques to work, however the downside is that the display was limited to the amount of samples of each trained pattern. The development of the next sample sub VI allowed the samples to be examined based upon the users input (Figure 36).

The testing of evaluation techniques was conducted with the purpose of determining the reliability of the assessments of these techniques. The testing conditions involved running several iterations of a single pattern such as walking or running, this particular set is evaluated against itself such that the output of all the methods would return a result indicating they are a 100% match.

The results concluded that the Rand Index was a more reliable method at classifying the patterns and required less data manipulation as the functions output was of a double datatype with values ranging from '0.0' to '1.00', where '0.0' is no match and '1.00' being a 100% match between the two patterns.

The evaluation model only requires two inputs into the function, the trained model and the baseline model. The program has allowed a minimum % match to account for noise, different human anatomies and physiologies which affect the gait patterns, as each individual will have unique gait due to several factors such as their age, height, weight and injuries. The minimum % match varies greatly between walking and running as running is a faster and more dynamic movement than passive walking.

After the above was completed a solution to allowing the user to control the samples began, the testing occurred in a duplicate VI to ensure all functionality remained the same in case of any irreversible changes. After several prototypes of array manipulation, it was deduced that the evaluation could only be processed externally, thus a separate VI was created specifically for acquiring the data and evaluating the patterns sampled.

13 CONCLUSION

The Capstone project revolving around the design and programming of a Virtual Instrument (VI) on LabVIEW, a graphical user interface (GUI) programming software. The program communicates with the Inertial Measurement Unit (IMU) to record the sampled data, which is formatted and processed for the K Means clustering algorithm. The algorithm classifies the sampled data as a pattern that can be evaluated against another pattern to determine if it is an identical match. This evaluation can be used for gait analyses where the gait patterns can be evaluated to determine movement problems.

The program allows the user to write the test data to file or to an excel spreadsheet where it can be easily graphed. The program also provides a real-time visual display prior to writing to file.

The program revolving around the K Means clustering algorithm was completed within the time restrains of the project and in working order however, the programming required to reach the algorithm phase was quite intensive.

13.1 FUTURE WORKS

The LabVIEW VI was developed for its simplicity in analyzing the main forms of gait, which were mentioned earlier to be the walking and running. There are certain restrictions in the above program such as the evaluation of each pattern to be of the same number of samples, thus this limitation can be improved upon by increasing the samples for both the test and trained patterns. For the scenario when the test samples exceed the trained patterns sample size of 50 e.g. 60 samples from the test and trained, here the trained data should copy itself however it would only add first 10 samples of the trained pattern to the end of itself. This method removes the restricted evaluation window thus providing a wider perspective of the analysis. This is possible as the sampled gait patterns are equivalent to one complete cycle of walking and running, thus the addition of the first 10 samples to the end of the 50 samples would essentially create a pattern of continuous walking.

14 REFERENCES

[1]   Benocci, M., Rocchi, L., Farella, E., Chiari, L. & Benini, L. 2009, 'A wireless system for gait and posture analysis based on pressure insoles and Inertial Measurement Units', Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on, IEEE, pp. 1-6.

[2]   Tao, W., Liu, T., Zheng, R. & Feng, H. 2012, 'Gait analysis using wearable sensors', Sensors, vol. 12, no. 2, pp. 2255-83.

[3]   Salarian, A.; Russmann, H.; Vingerhoets, F.J.; Dehollain, C.; Blanc, Y.; Burkhard, P.R.; Aminian, K. Gait assessment in Parkinson's disease: Toward an ambulatory system for long-term monitoring. IEEE Trans. Biomed. Eng. 2004, 51, 1434–1443.

[4]   Saremi, K.; Marehbian, J.; Yan, X.; Regnaux, J.; Elashoff, R.; Bussel, B.; Dobkin, B.H. Reliability and validity of bilateral thigh and foot accelerometry measures of walking in healthy and hemiparetic subjects. Neurorehabil. Neural Repair 2006, 20, 297–305.

[5]   Atallah, L.; Jones, G.G.; Ali, R.; Leong, J.J.H.; Lo, B.; Yang, G.Z. Observing recovery from knee-replacement surgery by using wearable sensors. In Proceedings of the 2011 International Conference on Body Sensor Networks, Dallas, TX, USA, 23–25 May 2011; pp. 29–34.

[6]   Aminian, K.; Najafi, B. Capturing human motion using body-fixed sensors: Outdoor measurement and clinical applications. Comput. Animat. Virtual Worlds 2004, 15, 79–94.

[7]   J. Mäntyjärvi, M. Lindholm, E. Vildjiounaite, S.-M. Mäkela̅, and H. J. Ailisto, "Identifying users of portable devices from gait pattern with accelerometers," in IEEE International Conference on Acoustics, Speech, and Signal Processing, March 2005.

[8]   Gafurov, D., Helkala, K. & Søndrol, T. 2006, 'Biometric gait authentication using accelerometer sensor', Journal of computers, vol. 1, no. 7, pp. 51-9.

[9]   D. Gafurov, E. Snekkenes, and T. E. Buvarp, "Robustness of biometric gait authentication against impersonation attack," in First International Workshop on Information Security (IS'06), OnTheMove Federated Conferences (OTM'06), Montpellier, France, Oct 30 - Nov 1, 2006, Springer LNCS, to appear.

[10]   A. Sant'Anna and N. Wickström, "DEveloping a Motion Language: Gait Analysis from Accelerometer Sensor Systems," 3rd International Conference on Pervasive Computing Technologies for Healthcare, London, 1-3 April, 2009, p. 1-8.

[11]   Yang, M., Zheng, H., Wang, H., McClean, S. & Newell, D. 2012, 'iGAIT: An interactive accelerometer based gait analysis system', Computer Methods and Programs in Biomedicine, vol. 108, no. 2, pp. 715-23.

[12]   Nowlan, M.F. 2009, 'Human recognition via gait identification using accelerometer gyro forces', Available: cs-www. cs. yale. edu/homes/mfn3/proj/mfn_gait_id. pdf.

[13]   Sprager, S. & Zazula, D. 2009, 'Gait identification using cumulants of accelerometer data', 2nd WSEAS International Conference on Sensors, and Signals and Visualization, Imaging and Simulation and Materials Science, pp. 94-9.

[14]   Kennedy, R.L. 1997, Solving Data Mining Problems Through Pattern Recognition, Prentice Hall PTR.

[15]   Vathy-Fogarassy, Á. & Abonyi, J. 2013, Graph-Based Clustering and Data Visualization Algorithms, Springer.

[16]   Maulik, U. & Bandyopadhyay, S. 2002, 'Performance evaluation of some clustering algorithms and validity indices', Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 24, no. 12, pp. 1650-4.

[17]   Bandyopadhyay, S. & Saha, S. 2012, Unsupervised classification: similarity measures, classical and metaheuristic approaches, and applications, Springer Science & Business Media.

[18]   Bitter, R., Mohiuddin, T. & Nawrocki, M. 2006, LabVIEW: Advanced programming techniques, CRC Press.

[19]   Kring, J. & Travis, J. 2006, 'LabView for Everyone Graphical Programming Made Easy and Fun', Prentice Hall.

[20]   Essick, J. 2009, Hands-on introduction to LabVIEW for scientists and engineers, New York : Oxford University Press, 2009.

[21]   Ritter, D.J. 2002, LabVIEW GUI : essential techniques, New York : McGraw-Hill, c2002.

15 APPENDICES

Appendix A – LabVIEW Virtual Instrument

Appendix B – 6 Degrees of Freedom v4 Datasheet

Appendix C – Three Axis Low-g Micro machined Accelerometer (1.5 – 6g) Datasheet

Appendix D – Machine Learning Toolkit User Manual
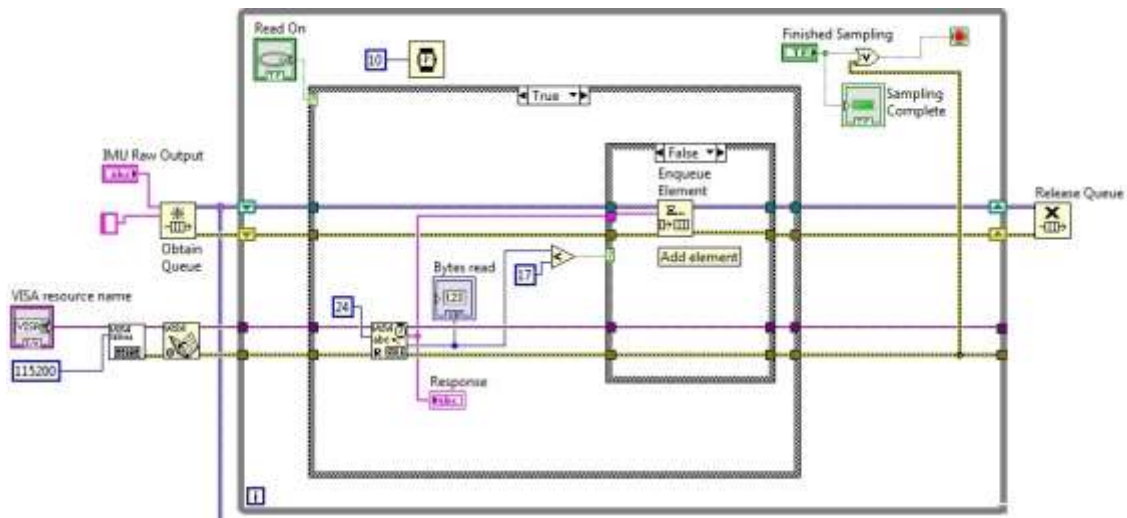
15.1 APPENDIX A – LABVIEW VIRTUAL INSTRUMENT
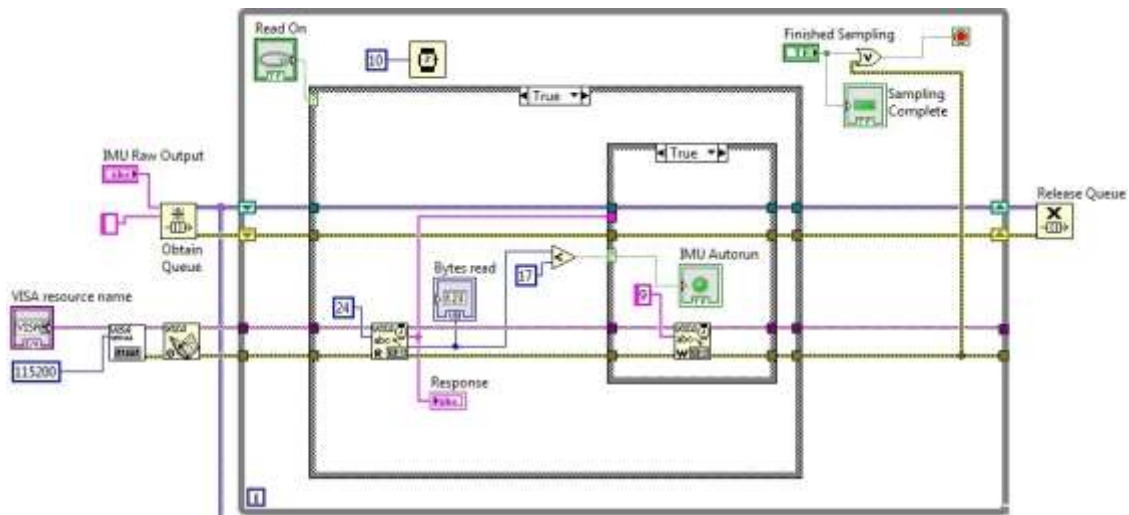

Figure 26. LabVIEW Data Collection Phase False Case

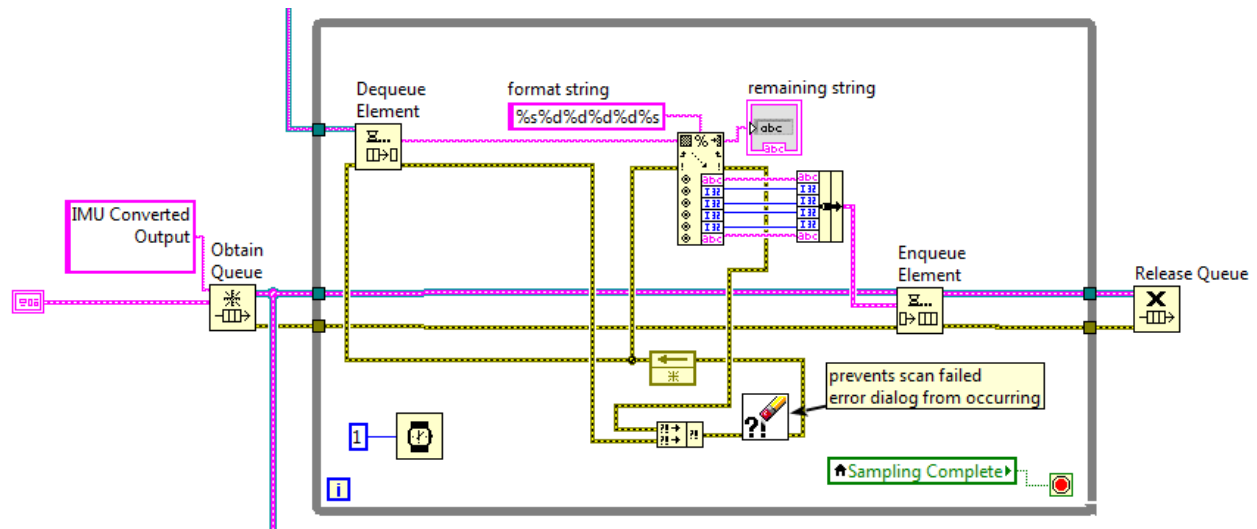*Figure 27. LabVIEW Data Collection Phase True Case*

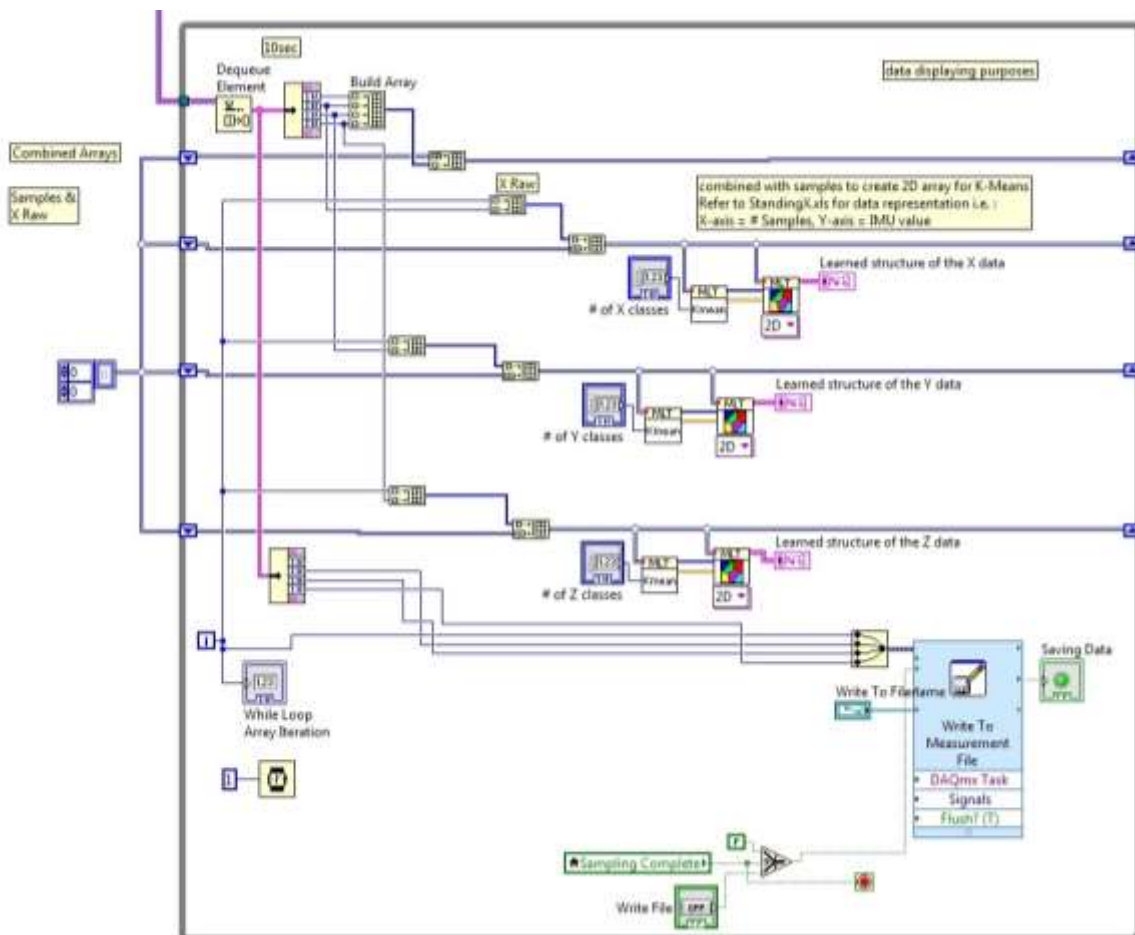*Figure 28. Data Preparation Phase*

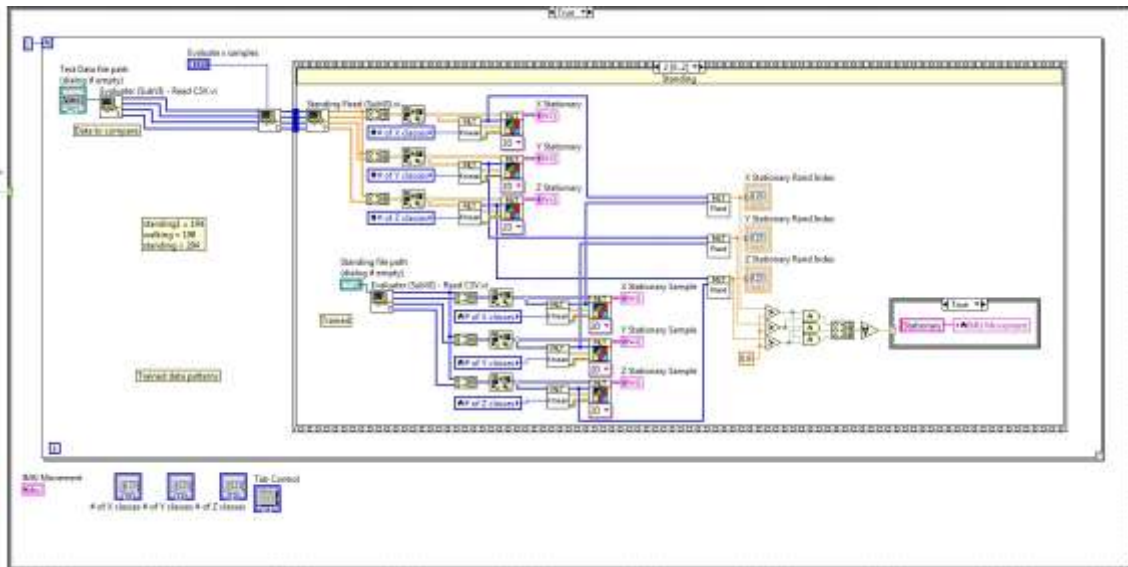*Figure 29. Data Preprocessing & Implementation & Training*

Figure 30. Algorithm Implementation, Training, Testing & Evaluation (1/3) – Stationary
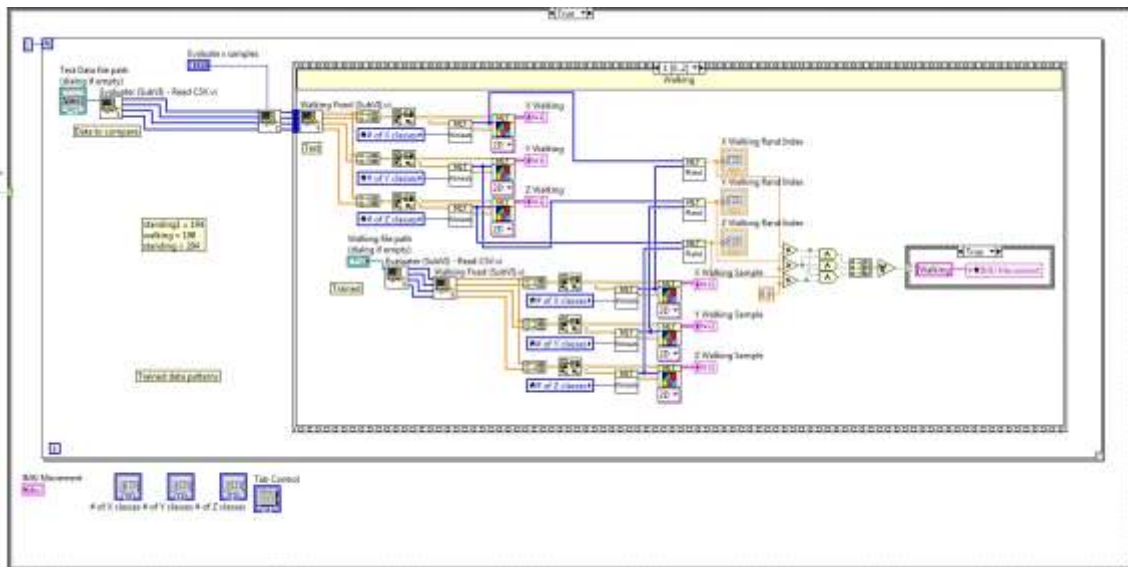

Figure 31. Algorithm Implementation, Training, Testing & Evaluation (2/3) – Walking
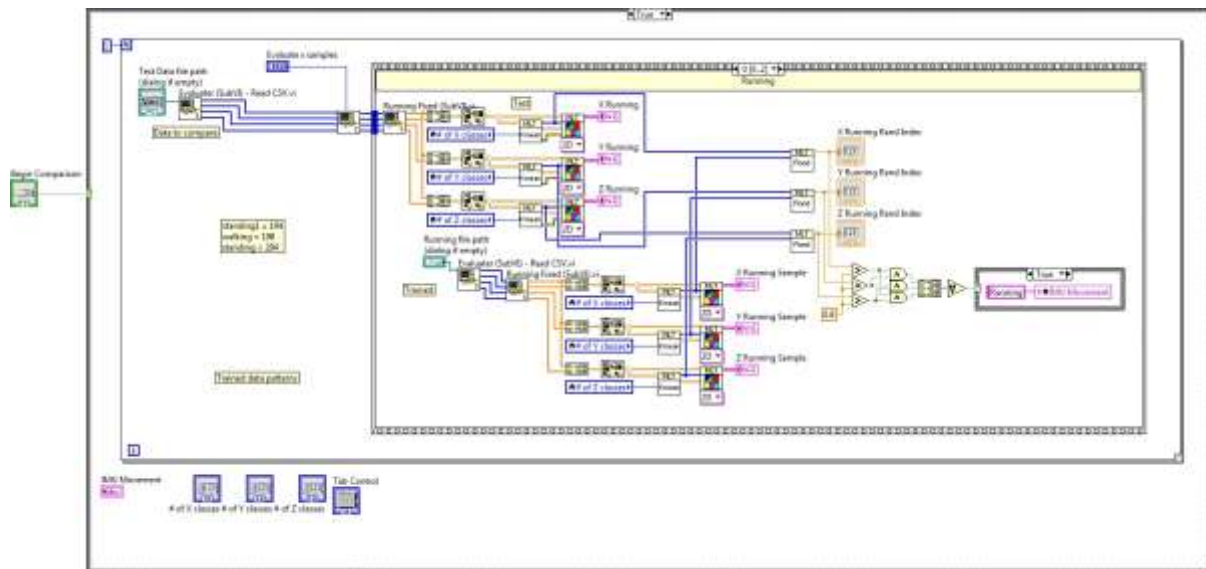
Figure 32. Algorithm Implementation, Training, Testing & Evaluation (3/3) – Running

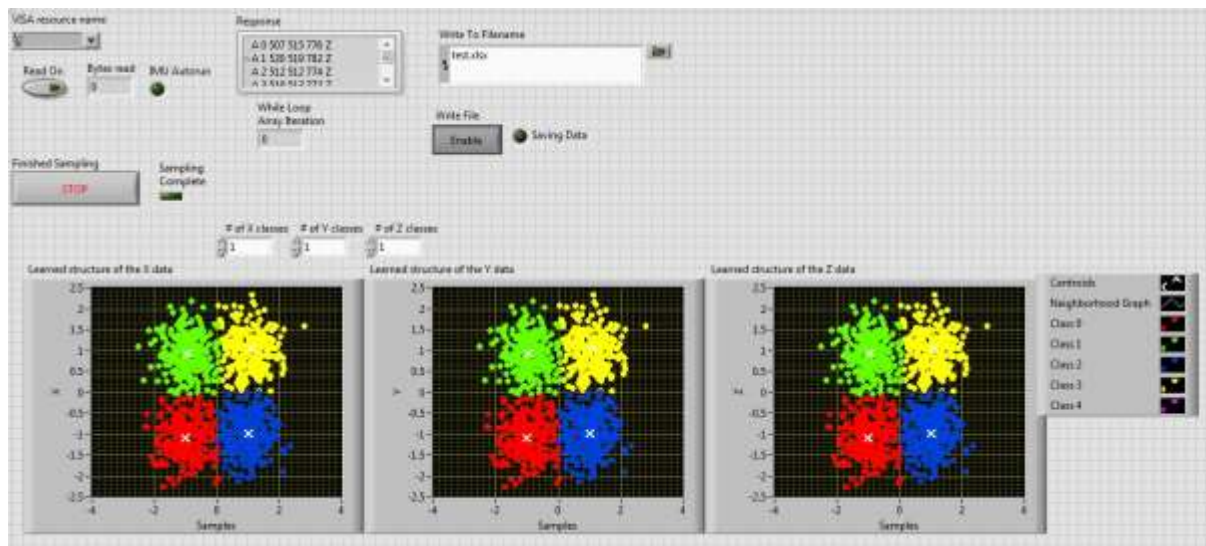*Figure 33. LabVIEW VI Sampler Front Panel*
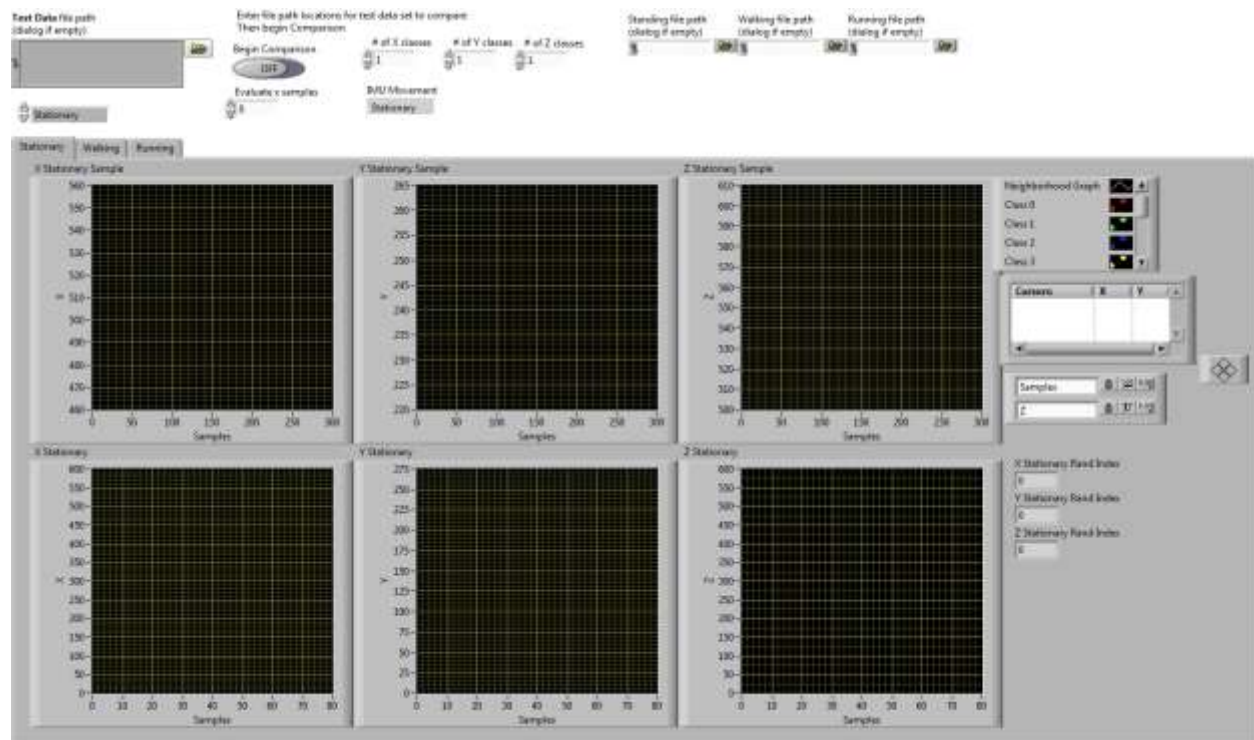
Figure 34. LabVIEW VI Evaluator Front Panel

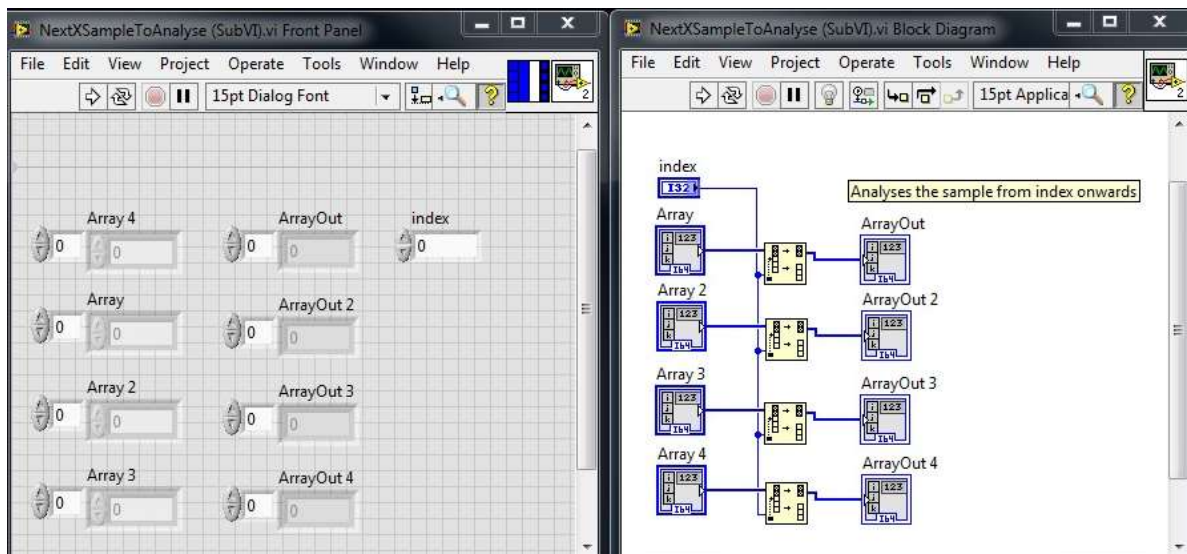*Figure 35. Read From CSV File Sub VI*
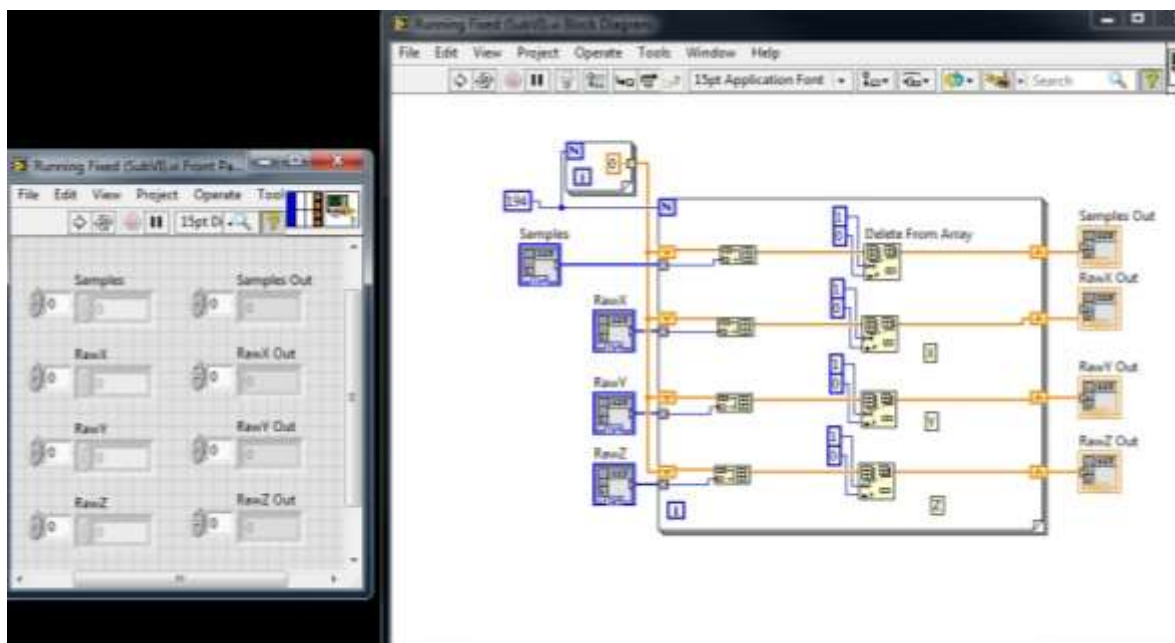
Figure 36. Next Sample Sub VI

Figure 37. Fixed Array - Running SubVI
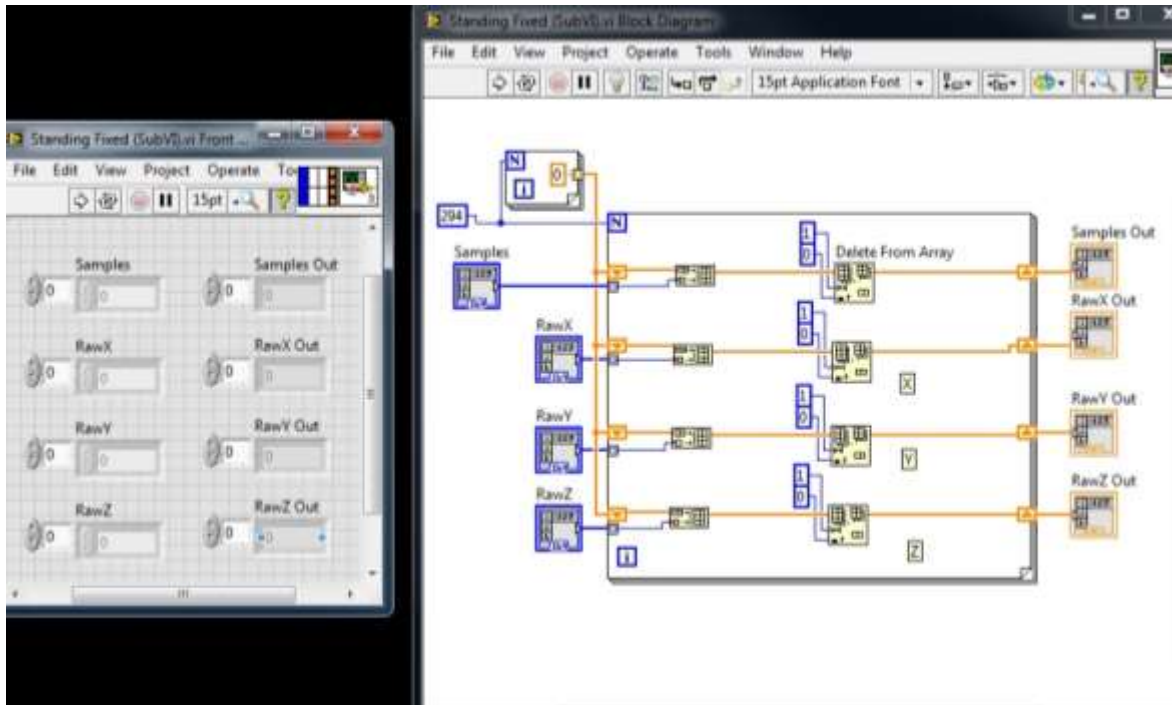
Figure 38. Fixed Array - Standing SubVI
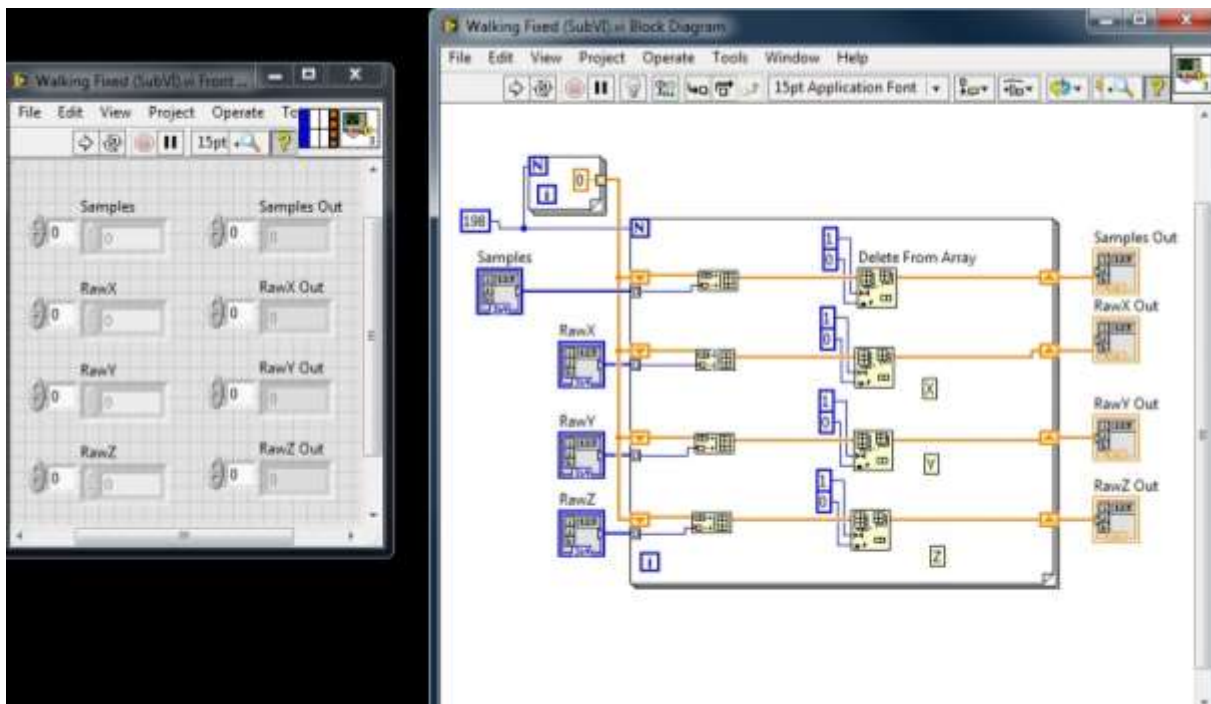
Figure 39. Fixed Array - Walking SubVI
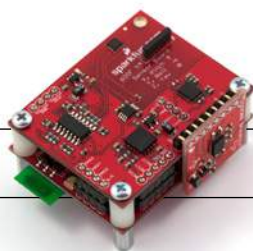
Figure 40. String Manipulation Prototype

15.2 APPENDIX B – 6 DEGREES OF FREEDOM V4 DATASHEET

sparkfun ELECTRONICS™

WEBSITE: sparkfun.com

6175 LONGBOW DRIVE, SUITE 200   ZIP CODE: 80301
BOULDER, COLORADO   USA

[303] 284.0979 [GENERAL]   P F
       443.0048

2008.04.22

IMU 6 Degrees of Freedom v4 Data Sheet

1   Overview

The 6DOF v3 gets a make-over!

The 6DOF v4 Inertial Measurement Unit (IMU) is the newest SparkFun IMU offering, bringing the best features of the v3, plus a few new functions inspired by customer feedback and suggestions.

The v4 provides 3 axes of acceleration data, 3 axes of gyroscopic data, and 3 axes of magnetic data. Each independent channel is user selectable, as is the sampling frequency. The device can also report in ASCII or binary format, and can work over a Bluetooth link or a TTL hardline. Control is provided through an LPC2138 ARM7 processor with plenty of extra memory for custom code development. Additionally, the code has been ported to the WinARM development platform. With the freely available source code, you can be doing your own development in minutes!

The IMU 6-DOF v4 uses these sensors:

- Freescale MMA7260Q triple-axis accelerometer, settable to 1.5 g, 2 g, 4 g or 6 g sensitivity
- 2 InvenSense IDG500 500 degree/second gyros
- Honeywell HMC1052L and HMC1051Z magnetic sensors

All sensor readings are available through any terminal program in either ASCII or binary format, or with the improved 6DOF v4 IMU Mixer demo application (source code also available). Additionally, all sensors are internally temperature compensated.

2   Electrical Specs

- Input voltage: 4.2V to 7V DC
- Current consumption: less that 150mA
- Frequency response:
  - Magnetic sensors: 312Hz
  - IDG500 Gyros: 140Hz
  - MMA7260Q Accelerometer:
    - 350Hz, X and Y axes
    - 150Hz, Z axis

For a full description of the sensor specifications, please see the respective manufacturer's data sheets (available at www.Sparkfun.com).

3   Hardware Overview

Like the v3, the v4 is a double-decker unit with the controller board on the bottom and the sensor board on the top. Each board has its own 3.3V regulator in order to better separate the digital and analog circuits.

3.1   Controller Board

There are a few things that the user may want to familiarize themselves with regarding the controller board.

Figure 1: Controller Board
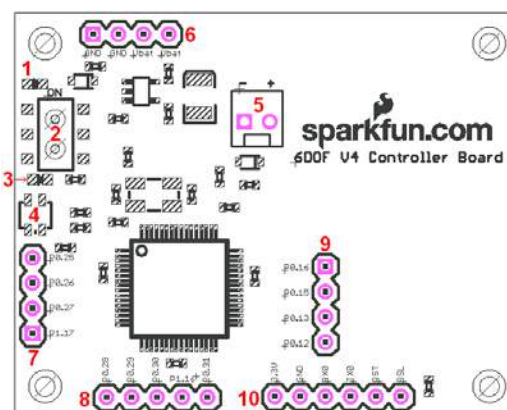
1) Power indicator LED
2) Power switch
3) Bluetooth connection indicator LED
4) Tri-color status LED
5) Power connector
6) Power port for sensor board, switched from controller board
7) Magnetic sense lines
8) Accelerometer sense lines
9) Gyro sense lines
10) Programming port and TTL serial line

To aid in custom development, we've indicated the individual port numbers of the LPC2138 processor to which the user has access. All ADC lines are 0V to 3.3V, 10-bit. For a more in depth description of the LPC2138's capabilities, please see the LPC2138 User's manual.

3.2   Sensor Board

There are a few things that the user may want to familiarize themselves with regarding the controller board.

As stated previously, the sensor board uses Honeywell HMC1052L and HMC1051Z magnetic sensors. We've also included set/reset circuitry (used for realigning each sensor's magnetic domains

Figure 2: Sensor Board

1) Power port
2) Magnetic sense lines, plus ser/reset line
3) Acceleration sense lines, plus range set lines
4) Gyro sense lines
5) Vertical-mount gyro port
6) Power indicator LED

4.1 Hard line connection

We've provided a back-door to get into the configuration over the hard line, but first the user needs to establish a hardware connection. The easiest way is to use one of our LPC programming adapters PGM-00714 or PGM-08650, but the user will still need to either solder a female header or the programming adapter itself to the 6DOF v4. We suggest using a header, but if you should choose to solder directly take care to get the orientation correct.. All of the signal lines are printed on the respective PCB's, so all you need to do is line them up.

If you should choose another means of hardware connection, you will need to know that the 6DOF v4 controller board TX and RX lines are 0 to 3.3V, not 5V, and certainly not RS-232. If you connect the controller board directly to an RS-232 line, you will likely damage the board.

From this point, we will assume that the user has provided for the hardware connection and we'll continue with the setup. Load your battery pack and plug it in to the 6DOF (or otherwise apply power to the device). Start with your 6DOF off but connected to your serial line. Open up a terminal program (Hyperterminal, Teraterm, etc) to the port to which you're connected with settings of 115,200 baud, 8 data bits, one stop bit, no parity or flow control. Now simultaneously hold down the spacebar on your keyboard and turn the 6DOF on. You should see the configuration menu come up (it will likely scroll past a few times since the keyboard repeats). Release the spacebar and choose option 6 to toggle the output port to "serial TTL". Cycle power for the setting to take effect.

4.2 Bluetooth connection

Generally speaking, a Bluetooth discovery will show the 6DOF v4 as "Firefly" with a random MAC. If you're using a Bluetooth device other than what is available from SparkFun, connect to the 6DOF v4 according to that manufacturer's directions as a serial port.

If you're using one of the Roving Networks Bluetooth devices available through SparkFun, you can directly open a serial port to the local device and issue discovery and connection commands without the interference of third-party Bluetooth drivers or other supporting software.

The first thing to do is to load your battery pack, plug it into your 6DOF v4 and turn it on. Assuming the user has a Roving Networks Bluetooth dongle (like the BlueDongle-RN-USB), the setup is very simple. Plug in your dongle and open a terminal program to the correct port, set to the same settings mentioned previously (115,200 baud, etc.). Issue the command "$$$<enter>" and the dongle will answer back with "CMD?". Then enter the command "I<enter>" and the dongle will go off and search for other Bluetooth devices for a few seconds. When it's done, it will report to you how many devices it found and give you their names and MAC addresses. The 6DOF v4 will show up as "Firefly" with a random MAC address. Now enter the command "C,<MAC address><enter>", where <MAC address> is that of the 6DOF v4, and the dongle will answer back with

before each measurement), and each channel of the magnetic sensors has a high-gain differential amplifier as suggested by the HMC105X documentation. Supporting components have been kept at a distance from the sensors so as to increase the accuracy of the measurements.

The MMA7260Q accelerometer and the IDG500 gyros have each been set up per their manufacturer's recommendations, i.e. internal clock suppression filters on their outputs. These sensors are also internally temperature compensated.

All of the sensor lines have been indicated on the PCB should the user wish to use the 6DOF v4 sensor board with another control system, and all analog outputs are 0V to 3.3V.

4   Setup

When you first power up the 6DOF v4, you will see the power indicator LEDs on both the controller board and the sensor board light up. You will also see the tri-color status LED flash a few times. During normal sampling operations, the status LED will toggle red-blue-green, each color for 64 sample frames. When in the configuration menu, all colors will be on. If the device is not in the configuration menu and not sampling, the 6DOF is in its idle state and the status LED will be off (see the section on the configuration menu auto run mode for more details on this).

The 6DOF v4 comes pre-configured to run over Bluetooth. But if you haven't got a Bluetooth module…

IMU 6 Degrees of Freedom v4 Data Sheet

"Trying". The dongle won't report the successful connection with text in the terminal window, but you'll see the connection indicator LED light up on both the 6DOF and the dongle. Now you're connected and ready to run.

4.3   6DOF v4 Mixer demo application

For a demonstration of the 6DOF v4's operation, the user can run the 6DOF v4 Mixer program. With the 6DOF powered up and connected in its idle state, close any terminal programs, open the 6DOF's serial port and start the mixer application. Select the port number to which the 6DOF is connected, set the frequency and sensitivity, hit the start button and you're off and running.

The mixer program requires that all channels are active. If you have any trouble getting the program to work right, check that all channels are active in the configuration menu.

5   Using a Terminal and the Configuration Menu

Once the novelty of the mixer program wears off, the user may want to do something slightly more useful with the v4, like attach it to something and log a file with a terminal program. What follows is a description of the configuration menu as well as a functional description of the v4's operation.

5.1   Operation from the Idle State

Upon reset and in its default configuration, the v4 will check to see if it has been configured for "auto run" mode (more on that later). If it's not in auto run mode, it goes into an idle state waiting for input. Normally this idle state serves as the start up state for the mixer application. In this state, the following inputs have the following effects:

1) "%", ASCII 37, sets the accelerometer to 1.5 g sensitivity

2) "&", ASCII 38, sets the accelerometer sensitivity to 2 g

3) " ' " (apostrophe), ASCII 39, sets the accelerometer sensitivity to 4 g

4) " (", ASCII 40, sets the accelerometer sensitivity to 6 g

5) ") ", ASCII 41, sets the sample frequency to 50Hz

6) " * ", ASCII 42, sets the sample frequency to 100Hz

7) "+", ASCII 43, sets the sample frequency to 150Hz

8) " , ", ASCII 44, sets the sample frequency to 200Hz

9) " - ", ASCII 45, sets the sample frequency to 250Hz

10) " . ", ASCII 46, sets the sample frequency to 300Hz

11) "#", ASCII 35, starts the unit running in binary mode with all channels active

12) " " (space), ASCII 32, stops the unit and returns it to the idle state (issuing another ASCII 32 will bring up the configuration menu)

Operation from this idle state will always be in binary output mode, but the user may select which channels are active. Also, configuration from this state as done in the mixer application will not be saved

in memory, whereas settings from the actual configuration menu will be saved to memory for future use. At the same time, all but the active channel settings saved in memory have no effect on operation from the idle state. It should be noted that the primary purpose of this idle state and mode of operation is to more easily interact with the v4 mixer demonstration application, but there's no reason that a user's own application couldn't use it for a quick setup. It should also be noted that the 300Hz sampling rate is not available from the mixer app's selection, the reason being that the Bluetooth begins to exhibit latency when streaming data into it at somewhere between 250Hz and 300Hz with all channels active. But the 300Hz option is available from the idle state in case the user wishes to try it. Of course, operation from the configuration menu has no maximum setting (more on that later).

5.2   Operation from the Configuration Menu

To use the v4 from a terminal program, start up your program of choice (115,200 baud, hardware flow control, 8 data bits, one stop bit, no parity), plug in your Blue Dongle and power up your v4. When the devices connect, the Blue Dongle will report the connection established back to you. The v4 is now in the aforementioned idle state. Pressing the spacebar will bring up the configuration menu (STAT0 and STAT1 will come on as well), and here's what you'll see:

```
6DOF v4 setup, version 1.0

   1)  View/edit active channel
       list
   2)  Change output mode,
       currently binary
   3)  Set Auto run mode,
       currently off
   4)  Set accelerometer
       sensitivity, currently 1.5 g
   5)  Set output frequency,
       currently 100
   6)  Change output port,
       currently Bluetooth
   7)  Save settings and run unit
```

5.3   Active Channel List

Pressing "1" will bring up the active channel list:

```
   1)  Magneto X = on
   2)  Magneto Y = on
   3)  Magneto Z = on
   4)  Accel X = on
   5)  Accel Y = on
   6)  Accel Z = on
   7)  Pitch = on
   8)  Roll = on
   9)  Yaw = on
Press the number of the channel you wish to
change,or press x to exit.
```

To change a channel from active to inactive (or the reverse), just press the number of the channel you wish to change. It's a toggling

function; pressing a number will bring up the full list again, but with the channel you wished to change in its opposite state. Press a few numbers and get a feel for it. Pressing "x" gets you back to the main menu.

5.4   Output Mode

Pressing "2" from the main menu will toggle the output mode from binary to ASCII and back again. What are these output modes, you ask?

In both output modes, the data from all active channels is framed by an "A" (ASCII 65) at the start and a "Z" (ASCII 90) at the end. Also in both modes, each channel is reported in exactly the sequence shown in the active channel list, with the addition of a sample count that immediately follows the "A" and precedes the first active measurement, which is to say:

1) Count
2) Mag X
3) Mag Y
4) Mag Z
5) Accel X
6) Accel Y
7) Accel Z
8) Pitch
9) Roll
10) Yaw

The count is two bytes that comes as MSB-LSB, and will range from 0 to 32767. If any of the channels are selected as inactive, that data is omitted from the frame and subsequent data moves up in the report sequence.

In binary mode, each active channel report comes as 2 bytes: MSB and LSB, in that sequence, and they will always be between 0 and 1023 because we're reading from 10-bit ADC's. The width of the data frame in binary mode will be 4 bytes ("A", "Z", and count are always present) plus 2 bytes for each active measurement. So for all channels active the data frame will be 22 bytes wide.

In ASCII mode, the count and active measurements are reported in ASCII so it's easier to read with a terminal program, plus all measurements and the count are delimited with TAB characters (ASCII 9) as well as a carriage return and line feed at the end of the data frame. This makes data capture and importation into a spreadsheet a relatively simple matter.

5.5   Auto Run Mode

Pressing "3" from the main menu will toggle the auto run setting. If you intend to use the v4 in ASCII mode, set this to "on". If the auto run feature is off, the v4 will always run from it's primary idle state, which means that it will always wait for a "#" to begin sampling and it will always run in binary mode.

One feature of auto run mode is that if the setting is active the v4 will begin sampling immediately upon power up, or after establishment

of a Bluetooth connection if the Bluetooth is active. Pressing the spacebar will bring up the configuration menu again.

5.6   Setting the Accelerometer Sensitivity

Pressing "4" from the main menu will bring up the following submenu:

```
Set to:
    1)  1.5g
    2)  2g
    3)  4g
    4)  6g
```

Just press the number which corresponds to your choice and the v4 will revert to the main menu with the sensitivity changed.

5.7   Setting the Output Frequency

Pressing "5" from the main menu will allow you to change the sample frequency. Simply press "i" to increase or "d" to decrease, or "x" to revert to the main menu.

The minimum frequency setting is 10Hz, and there is no maximum setting. This allows the user to experiment with smaller data frames and higher sampling rates.

5.8   Setting the output port

Pressing "6" will allow you to toggle the output port, either Bluetooth or serial TTL. After selecting this option, you will be prompted to cycle power for this setting to become active.

5.9   Save Settings and Run Unit

Pressing "9" from the main menu will save the current settings to flash and exit the configuration menu. If the auto run feature has been activated the unit will begin running immediately. If it has not been set, the unit will revert to the initial idle state and wait for additional input.

6   Bandwidth Considerations and Firmware

The 6DOF v4 does not have any filtering in firmware, though there is enough memory left in the LPC2138 flash program space to implement some filtering. The internally set output bandwidth of the MMA7260Q accelerometer is 350Hz for the X and Y axes, and 150Hz for the Z axis. There are also additional single pole low pass filters to reduce switching noise from the sensor with poles set at 1591Hz (recommended by Freescale). The internally set output bandwidth for the IDG300 gyro sensors is 120Hz, along with single pole low pass filters at 96Hz (recommended by InvenSense). The HMC1052L and HMC1051Z magnetic sensors don't have internal filtering, but each axis has an external single pole low pass filter set at 319Hz. Of course, it's a good idea for the user to consider these numbers when developing an application to ensure that the proper filtering is in place for whatever sampling rate is selected.

All source code and schematics for the 6DOF v4 are freely available from the v4's product description page. Source code is compiled with GCC using WinARM (http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects/#winarm). Programming is typically done with the SparkFun LPC programming adapter (SKU #'s: PGM-00714 or PGM-08650) and Flash Magic, which is available for free download at http://www.flashmagictool.com/.
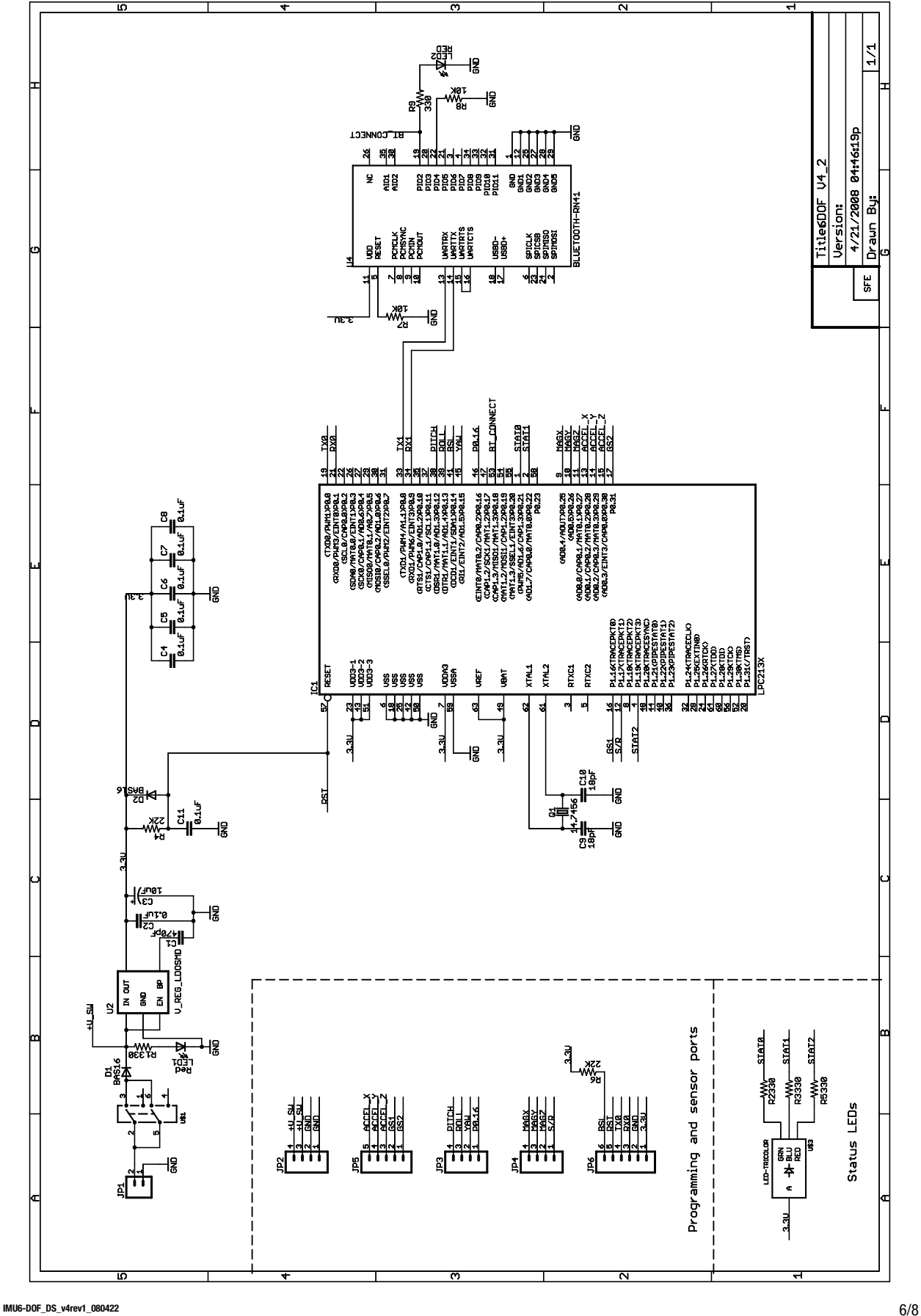
IMU6-DOF_DS_v4rev1_080422

DataSheet-6DOF-v4-Rev1.indd   5   8/17/09   2:47 PM

Title: 6DOF V4 Sensor Board

Magnetic sensors, ports and amps

Accelerometer

Power Section

Gyros

Reference Voltage Amplifier
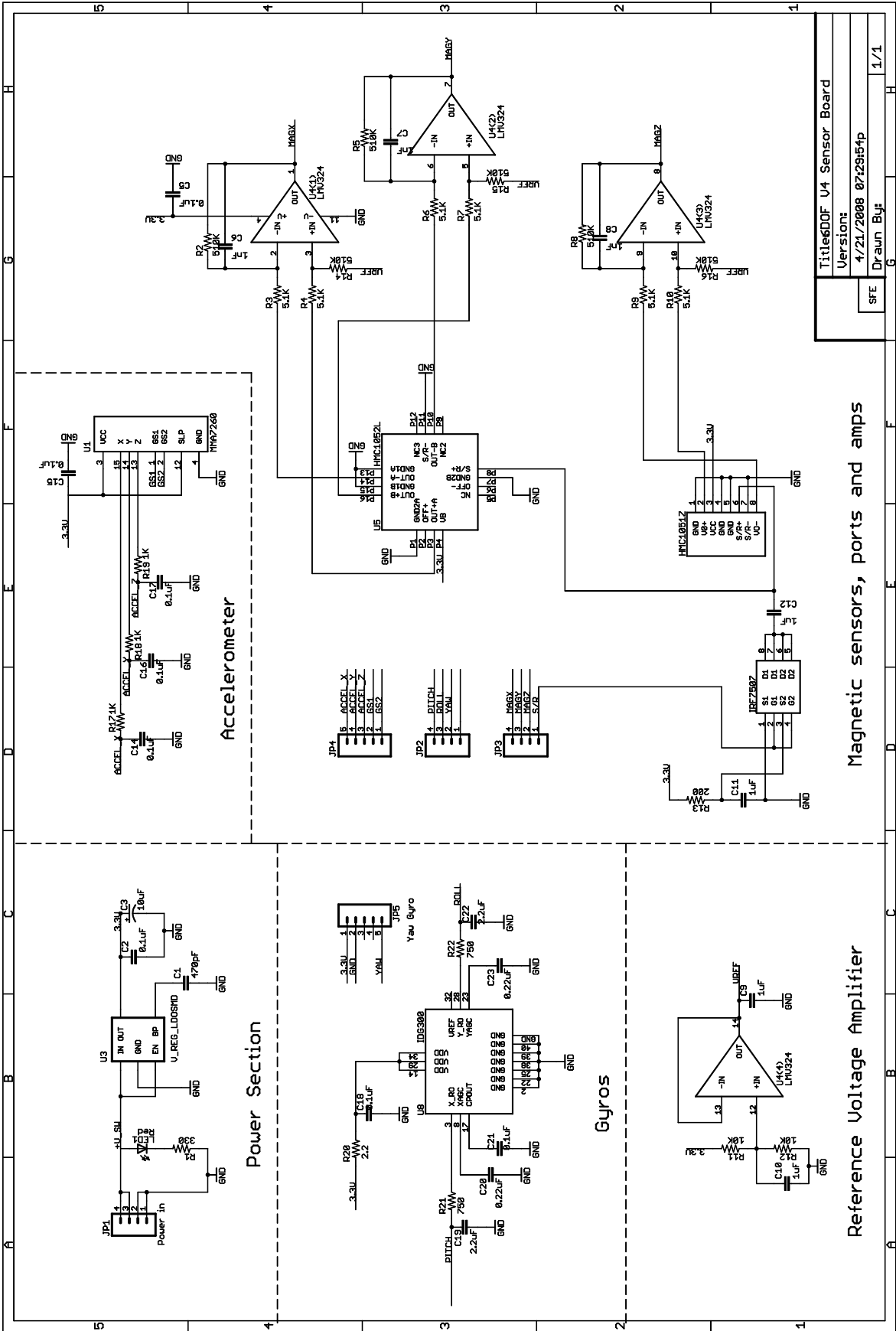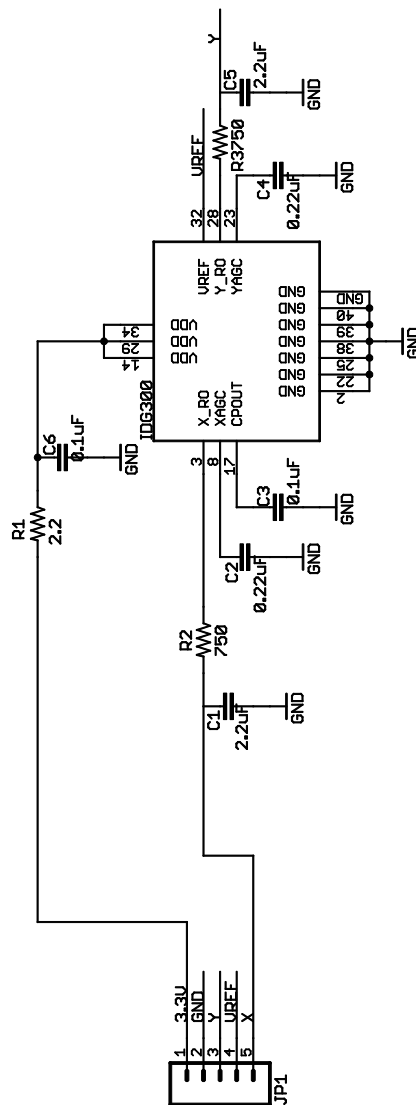
TITLE: IDG300 Breakout 2

Document Number:

REV:

SFE

Sheet: 1/1

Date: 4/21/2008 04:39:54p

15.3 APPENDIX C – THREE AXIS LOW-G MICRO MACHINED ACCELEROMETER (1.5 – 6G) DATASHEET

±1.5g - 6g Three Axis Low-g Micromachined Accelerometer

The MMA7260Q low cost capacitive micromachined accelerometer features signal conditioning, a 1-pole low pass filter, temperature compensation and g-Select which allows for the selection among 4 sensitivities. Zero-g offset full scale span and filter cut-off are factory set and require no external devices. Includes a Sleep Mode that makes it ideal for handheld battery powered electronics.

Features

- Selectable Sensitivity (1.5g/2g/4g/6g)
- Low Current Consumption: 500 μA
- Sleep Mode: 3 μA
- Low Voltage Operation: 2.2 V – 3.6 V
- 6mm x 6mm x 1.45mm QFN
- High Sensitivity (800 mV/g @1.5 g)
- Fast Turn On Time
- High Sensitivity (1.5 g)
- Integral Signal Conditioning with Low Pass Filter
- Robust Design, High Shocks Survivability
- Pb-Free Terminations
- Environmentally Preferred Package
- Low Cost

Typical Applications

- HDD MP3 Player : Freefall Detection
- Laptop PC : Freefall Detection, Anti-Theft
- Cell Phone : Image Stability, Text Scroll, Motion Dialing, E-Compass
- Pedometer : Motion Sensing
- PDA : Text Scroll
- Navigation and Dead Reckoning : E-Compass Tilt Compensation
- Gaming : Tilt and Motion Sensing, Event Recorder
- Robotics : Motion Sensing

MMA7260Q

MMA7260Q: XYZ AXIS ACCELEROMETER
±1.5g/2g/4g/6g

Bottom View

16 LEAD QFN CASE 1622-01

Top View

Figure 1. Pin Connections

ORDERING INFORMATION			
Device Name	Temperture Range	Case No.	Package
MMA7260Q	– 20 to +85°C	1622-01	QFN-16, Tube
MMA7260QR2	– 20 to +85°C	1622-01	QFN-16,Tape & Reel

freescale™
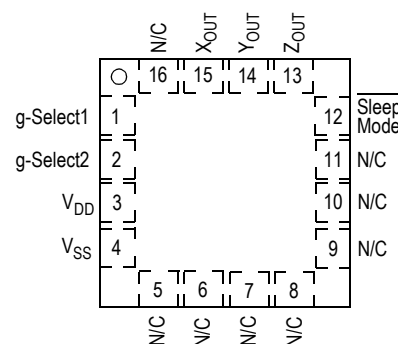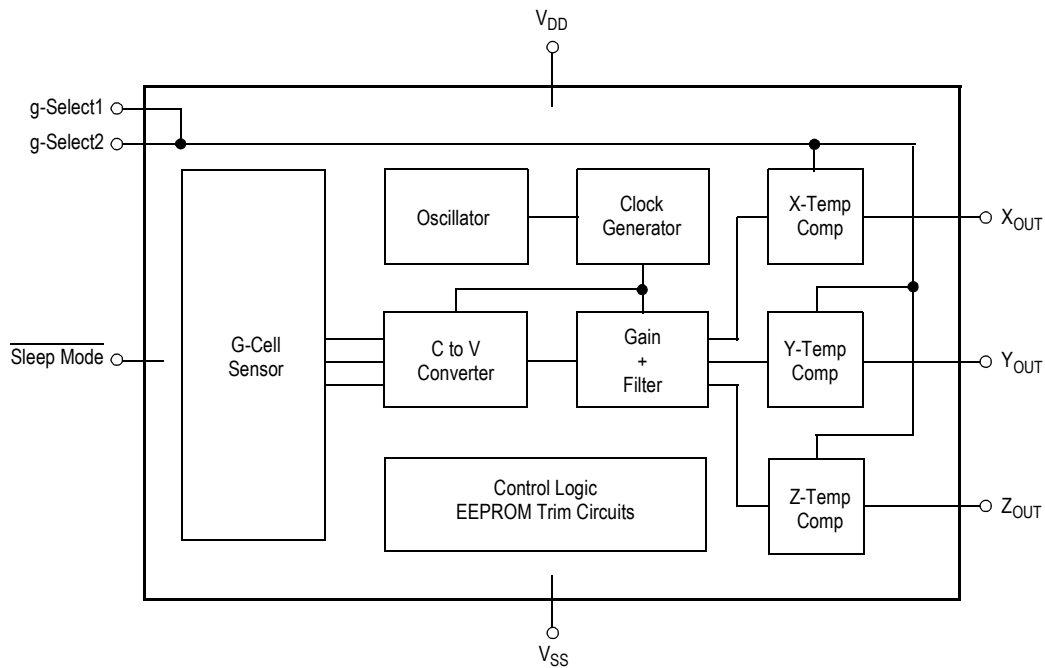semiconductor

Figure 2. Simplified Accelerometer Functional Block Diagram

Table 1. Maximum Ratings

(Maximum ratings are the limits to which the device can be exposed without causing permanent damage.)

Rating	Symbol	Value	Unit
Maximum Acceleration (all axis)	g_{max}	±2000	g
Supply Voltage	V_{DD}	−0.3 to +3.6	V
Drop Test[1]	D_{drop}	1.8	m
Storage Temperature Range	T_{stg}	−40 to +125	°C

1. Dropped onto concrete surface from any axis.

ELECTRO STATIC DISCHARGE (ESD)

WARNING: This device is sensitive to electrostatic discharge.

Although the Freescale accelerometer contains internal 2000 V ESD protection circuitry, extra precaution must be taken by the user to protect the chip from ESD. A charge of over 2000 volts can accumulate on the human body or associated test equipment. A charge of this magnitude can alter the performance or cause failure of the chip. When handling the accelerometer, proper ESD precautions should be followed to avoid exposing the device to discharges which may be detrimental to its performance.

Table 2. Operating Characteristics
Unless otherwise noted: $-20°C \leq T_A \leq 85°C$, $2.2\,V \leq V_{DD} \leq 3.6\,V$, Acceleration = 0g, Loaded output[1]

Characteristic	Symbol	Min	Typ	Max	Unit
Operating Range[2]					
Supply Voltage[3]	V_{DD}	2.2	3.3	3.6	V
Supply Current	I_{DD}	—	500	800	µA
Supply Current at Sleep Mode[4]	I_{DD}	—	3	10	µA
Operating Temperature Range	T_A	–20	—	+85	°C
Acceleration Range, X-Axis, Y-Axis, Z-Axis					
g-Select1 & 2: 00	g_{FS}	—	±1.5	—	g
g-Select1 & 2: 10	g_{FS}	—	±2.0	—	g
g-Select1 & 2: 01	g_{FS}	—	±4.0	—	g
g-Select1 & 2: 11	g_{FS}	—	±6.0	—	g
Output Signal					
Zero g (T_A = 25°C, V_{DD} = 3.3 V)[5]	V_{OFF}	1.485	1.65	1.815	V
Zero g	V_{OFF}, T_A	—	±2	—	mg/°C
Sensitivity (T_A = 25°C, V_{DD} = 3.3 V)					
1.5g	$S_{1.5g}$	740	800	860	mV/g
2g	S_{2g}	555	600	645	mV/g
4g	S_{4g}	277.5	300	322.5	mV/g
6g	S_{6g}	185	200	215	mV/g
Sensitivity	S, T_A	—	±0.03	—	%/°C
Bandwidth Response					
XY	f_{-3dB}	—	350	—	Hz
Z	f_{-3dB}	—	150	—	Hz
Noise					
RMS (0.1 Hz – 1 kHz)[4]	n_{RMS}	—	4.7	—	mVrms
Power Spectral Density RMS (0.1 Hz – 1 kHz)[4]	n_{PSD}	—	350	—	$µg/\sqrt{Hz}$
Control Timing					
Power-Up Response Time[6]	$t_{RESPONSE}$	—	1.0	2.0	ms
Enable Response Time[7]	t_{ENABLE}	—	0.5	2.0	ms
Sensing Element Resonant Frequency					
XY	f_{GCELL}	—	6.0	—	kHz
Z	f_{GCELL}	—	3.4	—	kHz
Internal Sampling Frequency	f_{CLK}	—	11	—	kHz
Output Stage Performance					
Full-Scale Output Range (I_{OUT} = 30 µA)	V_{FSO}	V_{SS}+0.25	—	V_{DD}–0.25	V
Nonlinearity, X_{OUT}, Y_{OUT}, Z_{OUT}	NL_{OUT}	–1.0	—	+1.0	%FSO
Cross-Axis Sensitivity[8]	$V_{XY, XZ, YZ}$	—	—	5.0	%

1. For a loaded output, the measurements are observed after an RC filter consisting of a 1.0 kΩ resistor and a 0.1 µF capacitor to ground.

2. These limits define the range of operation for which the part will meet specification.

3. Within the supply range of 2.2 and 3.6 V, the device operates as a fully calibrated linear accelerometer. Beyond these supply limits the device may operate as a linear device but is not guaranteed to be in calibration.

4. This value is measured with g-Select in 1.5g mode.

5. The device can measure both + and – acceleration. With no input acceleration the output is at midsupply. For positive acceleration the output will increase above $V_{DD}/2$. For negative acceleration, the output will decrease below $V_{DD}/2$.

6. The response time between 10% of full scale Vdd input voltage and 90% of the final operating output voltage.

7. The response time between 10% of full scale Sleep Mode input voltage and 90% of the final operating output voltage.

8. A measure of the device's ability to reject an acceleration applied 90° from the true axis of sensitivity.

MMA7260Q

PRINCIPLE OF OPERATION

The Freescale accelerometer is a surface-micromachined integrated-circuit accelerometer.

The device consists of two surface micromachined capacitive sensing cells (g-cell) and a signal conditioning ASIC contained in a single integrated circuit package. The sensing elements are sealed hermetically at the wafer level using a bulk micromachined cap wafer.

The g-cell is a mechanical structure formed from semiconductor materials (polysilicon) using semiconductor processes (masking and etching). It can be modeled as a set of beams attached to a movable central mass that move between fixed beams. The movable beams can be deflected from their rest position by subjecting the system to an acceleration (Figure 3) .

As the beams attached to the central mass move, the distance from them to the fixed beams on one side will increase by the same amount that the distance to the fixed beams on the other side decreases. The change in distance is a measure of acceleration.

The g-cell beams form two back-to-back capacitors (Figure 3). As the center beam moves with acceleration, the distance between the beams changes and each capacitor's value will change, $(C = A\epsilon/D)$. Where A is the area of the beam, ϵ is the dielectric constant, and D is the distance between the beams.

The ASIC uses switched capacitor techniques to measure the g-cell capacitors and extract the acceleration data from the difference between the two capacitors. The ASIC also signal conditions and filters (switched capacitor) the signal, providing a high level output voltage that is ratiometric and proportional to acceleration.
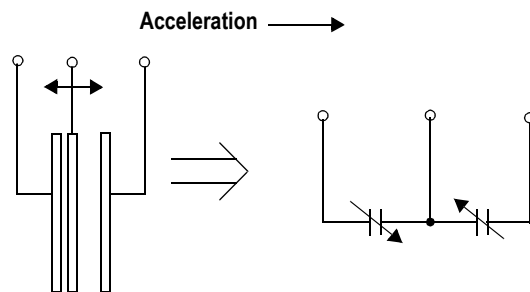
Acceleration →

Figure 3. Simplified Transducer Physical Model

SPECIAL FEATURES

g-Select

The g-Select feature allows for the selection among 4 sensitivities present in the device. Depending on the logic input placed on pins 1 and 2, the device internal gain will be changed allowing it to function with a 1.5g, 2g, 4g, or 6g sensitivity (Table 3). This feature is ideal when a product has applications requiring different sensitivities for optimum performance. The sensitivity can be changed at anytime during the operation of the product. The g-Select1 and g-Select2 pins can be left unconnected for applications requiring only a 1.5g sensitivity as the device has an internal pulldown to keep it at that sensitivity (800mV/g).

Table 3. g-Select pin Descriptions

g-Select2	g-Select1	g-Range	Sensitivity
0	0	1.5g	800mV/g
0	1	2g	600mV/g
1	0	4g	300mV/g
1	1	6g	200mV/g

Sleep Mode

The 3 axis accelerometer provides a Sleep Mode that is ideal for battery operated products. When Sleep Mode is active, the device outputs are turned off, providing significant reduction of operating current. A low input signal on pin 12 (Sleep Mode) will place the device in this mode and reduce the current to 3uA typ. For lower power consumption, it is recommended to set g-Select1 and g-Select2 to 1.5g mode. By placing a high input signal on pin 12, the device will resume to normal mode of operation.

Filtering

The 3 axis accelerometer contains onboard single-pole switched capacitor filters. Because the filter is realized using switched capacitor techniques, there is no requirement for external passive components (resistors and capacitors) to set the cut-off frequency.

Ratiometricity

Ratiometricity simply means the output offset voltage and sensitivity will scale linearly with applied supply voltage. That is, as supply voltage is increased, the sensitivity and offset increase linearly; as supply voltage decreases, offset and sensitivity decrease linearly. This is a key feature when interfacing to a microcontroller or an A/D converter because it provides system level cancellation of supply induced errors in the analog to digital conversion process.
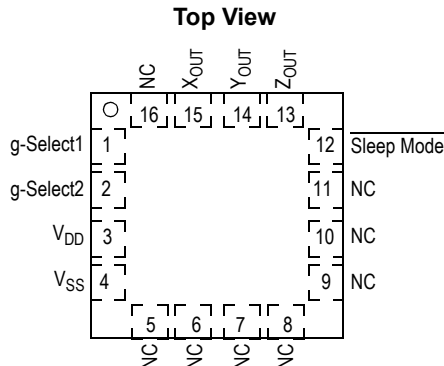
BASIC CONNECTIONS

Pin Descriptions

Top View

Figure 4. Pinout Description

Table 4. Pin Descriptions

Pin No.	Pin Name	Description
1	g-Select1	Logic input pin to select g level.
2	g-Select2	Logic input pin to select g level.
3	V_{DD}	Power Supply Input
4	V_{SS}	Power Supply Ground
5 - 7	N/C	No internal connection. Leave unconnected.
8 - 11	N/C	Unused for factory trim. Leave unconnected.
12	$\overline{\text{Sleep Mode}}$	Logic input pin to enable product or Sleep Mode.
13	Z_{OUT}	Z direction output voltage.
14	Y_{OUT}	Y direction output voltage.
15	X_{OUT}	X direction output voltage.
16	N/C	No internal connection. Leave unconnected.
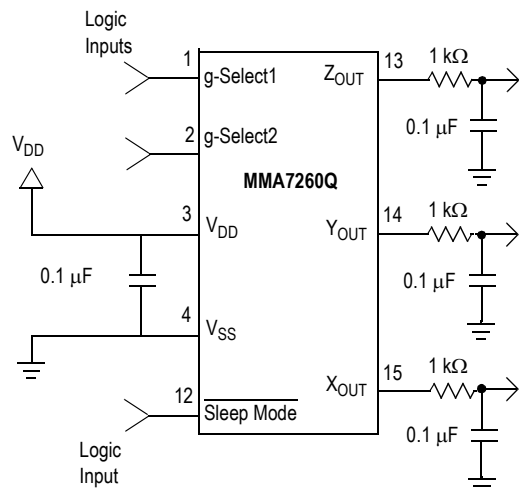
Figure 5. Accelerometer with Recommended Connection Diagram

PCB Layout
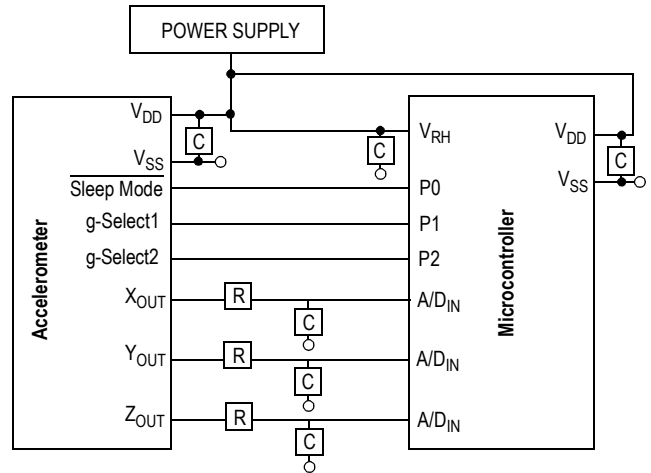
Figure 6. Recommended PCB Layout for Interfacing Accelerometer to Microcontroller
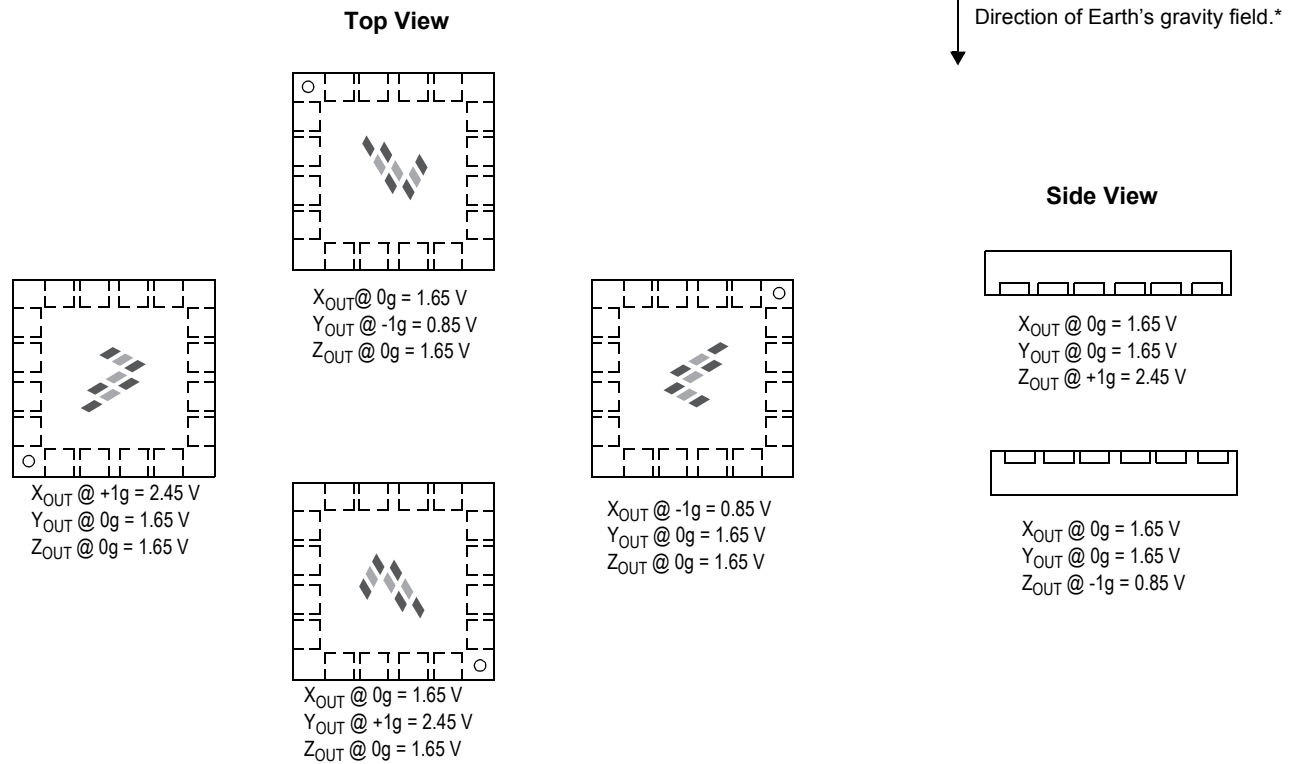
NOTES:

1. Use 0.1 µF capacitor on V_{DD} to decouple the power source.

2. Physical coupling distance of the accelerometer to the microcontroller should be minimal.

3. Flag underneath package is connected to ground.

4. Place a ground plane beneath the accelerometer to reduce noise, the ground plane should be attached to all of the open ended terminals shown in Figure 6.

5. Use an RC filter with 1.0 kΩ and 0.1 µF on the outputs of the accelerometer to minimize clock noise (from the switched capacitor filter circuit).

6. PCB layout of power and ground should not couple power supply noise.

7. Accelerometer and microcontroller should not be a high current path.

8. A/D sampling rate and any external power supply switching frequency should be selected such that they do not interfere with the internal accelerometer sampling frequency (11 kHz for the sampling frequency). This will prevent aliasing errors.

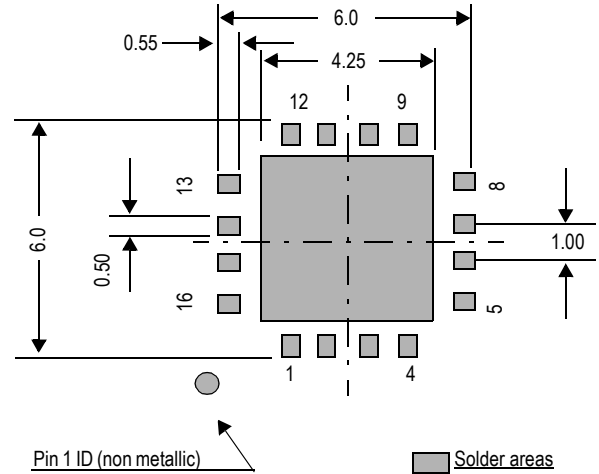**MMA7260Q**

DYNAMIC ACCELERATION

Top View

Side View

+Y

16 15 14 13
1 12
2 11
3 10
4 9
5 6 7 8

+X

-X

-Y

-Z

Top

Bottom

+Z

16-Pin QFN Package

: Arrow indicates direction of mass movement.

STATIC ACCELERATION

Direction of Earth's gravity field.*

Top View

X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ -1g = 0.85 V
Z_{OUT} @ 0g = 1.65 V

Side View

X_{OUT} @ +1g = 2.45 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT} @ 0g = 1.65 V

X_{OUT} @ -1g = 0.85 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT} @ 0g = 1.65 V

X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ +1g = 2.45 V
Z_{OUT} @ 0g = 1.65 V

X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT} @ +1g = 2.45 V

X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT} @ -1g = 0.85 V

* When positioned as shown, the Earth's gravity will result in a positive 1g output.

MMA7260Q

MINIMUM RECOMMENDED FOOTPRINT FOR SURFACE MOUNTED APPLICATIONS

Surface mount board layout is a critical portion of the total design. The footprint for the surface mount packages must be the correct size to ensure proper solder connection interface between the board and the package.

With the correct footprint, the packages will self-align when subjected to a solder reflow process. It is always recommended to design boards with a solder mask layer to avoid bridging and shorting between solder pads.

PACKAGE DIMENSIONS

DETAIL G
VIEW ROTATED 90° CLOCKWISE

C SEATING PLANE

DETAIL M

DETAIL M
PIN 1 INDEX

VIEW M-M
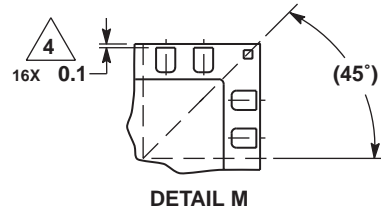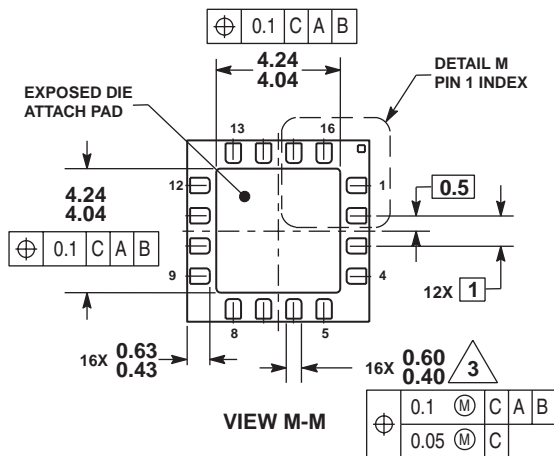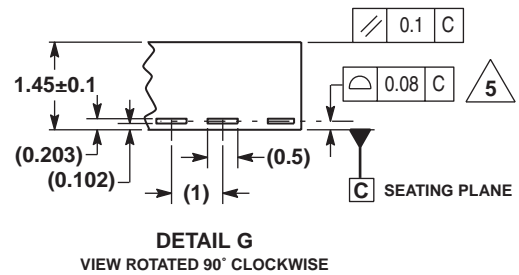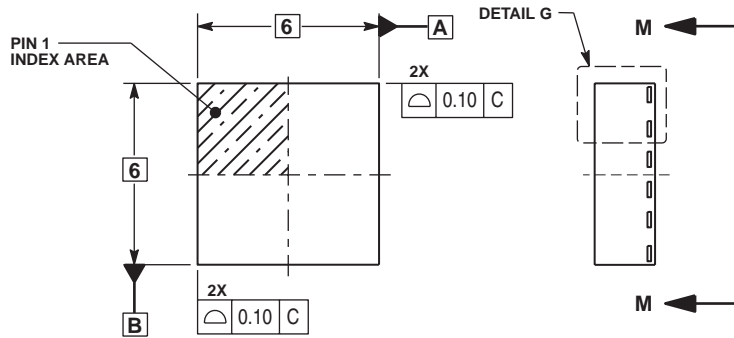
NOTES:
1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M, 1994.
3. THIS DIMENSION APPLIES TO METALLIZED TERMINAL AND IS MEASURED BETWEEN 0.25MM AND 0.30MM FROM TERMINAL TIP.
4. THIS DIMENSION REPRESENTS TERMINAL FULL BACK FROM PACKAGE EDGE UP TO 0.1MM IS ACCEPTABLE.
5. COPLANARITY APPLIES TO THE EXPOSED HEAT SLUG AS WELL AS THE TERMINAL.
6. RADIUS ON TERMINAL IS OPTIONAL.
7. MINIMUM METAL GAP 0.2MM.

CASE 1622-01
ISSUE O
16-LEAD QFN

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™
semiconductor

MMA7260Q
Rev. 1
06/2005

15.4 APPENDIX D – MACHINE LEARNING TOOLKIT USER MANUAL

LabVIEW Machine Learning Toolkit

User Manual

1. Introduction

The idea of machine learning is to mimic the learning process of human beings, i.e., gaining knowledge through experience. Machine learning algorithms allow machines to generalize rules from empirical data, and, based on the learned rules, make predictions for future data. The Machine Learning Toolkit (MLT) provides various machine learning algorithms in LabVIEW. It is a powerful tool for problems such as visualization of high-dimensional data, pattern recognition, function regression and cluster identification.

2. Features

The Machine Learning Toolkit includes algorithms, data types, validation functions, and visualization tools.

2.1 Machine Learning Algorithms
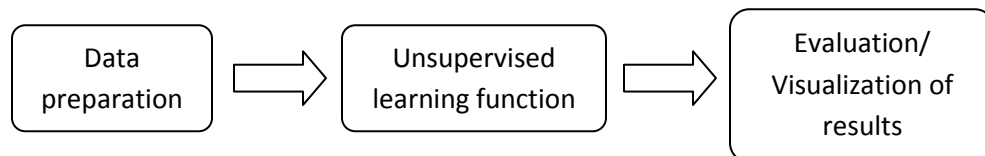
2.1.1   Unsupervised learning algorithms

Unsupervised learning refers to the problems of revealing hidden structure in unlabeled data. Since the data are unlabeled, there is no error signal fed back to the learner in the algorithm. This distinguishes unsupervised learning from supervised learning.

Clustering is one of the main and important approaches of unsupervised learning. Clustering means the assignment of class memberships to a set of objects so that similar objects are assigned into the same class and dissimilar ones are assigned into different classes. Each class often represents a meaningful pattern in the respective problem. Clustering is thereby useful for identification of different patterns in data. For example, in image processing clustering can be used to divide a digital image into distinct regions for border detection or object recognition.

List of functions:
- *k*-means
- *k*-medians
- *k*-medoids
- Fuzzy C-means
- Gaussian Mixture Model (GMM)
- Hierarchical Clustering
- Spectral Clustering
- Vector Quantization (VQ)
- Self-Organizing Map (SOM)

Conceptual diagram of usage of the Machine Learning Toolkit for unsupervised learning

```
┌──────────────┐       ┌──────────────────┐       ┌──────────────────┐
│     Data     │  ──▷  │   Unsupervised   │  ──▷  │   Evaluation/    │
│  preparation │       │ learning function │       │ Visualization of │
└──────────────┘       └──────────────────┘       │     results      │
                                                    └──────────────────┘
```

- Data preparation

  Data need to be formatted to fit the API of the unsupervised learning function the user selects.

- Unsupervised learning function application

  An unsupervised learning function is used to learn the structure of the input data.

- Evaluation/Visualization of results

  Refer to Section 2.4 for the choice of appropriate evaluation/visualization utility.

Examples:
- Example_Clustering
- Example_SOM

2.1.2   Supervised learning algorithms

Supervised learning refers to the generalization of the relationship (function) between the input data and their corresponding outputs (labels). The relationship (function) is learned through a training set of examples, each of which is a pair of an input data and a desired output. During the training, the error between the actual and the desired outputs is frequently fed back into the system for tuning the system parameters according to certain learning rule. The system "learns" by adapting itself to minimize the error. After the training, the performance of the learned relationship (function) should be evaluated on a test set (of examples) that is separate from the training set.

Supervised learning is useful for pattern recognition, function regression, etc. One example of applications is recognition of handwritten numbers. A supervised classifier can be trained with a reservoir of handwritten numbers, each with a label (the true number each image represents). Having been validated on a separate test set, the trained classifier can be used for fast and accurate recognition of future handwritten numbers.

List of functions:
- *k*-Nearest Neighbors (*k*-NN)
- Back-propagation (BP) Neural Network
- Learning Vector Quantization (LVQ)
- Support Vector Machine (SVM)

Conceptual diagram of usage of the MLT for supervised learning
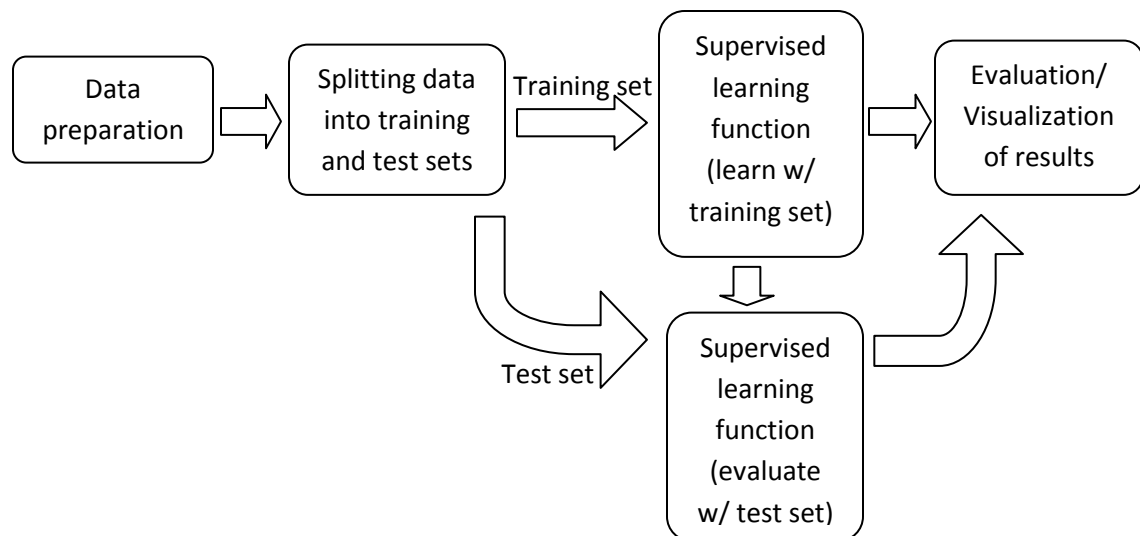
- Data preparation

    Data need to be formatted to fit the API of the unsupervised learning function the user selects.

- Splitting data into training and test sets

The MLT provides a utility (Training & Test Set.vi) to split original data into a training set and a test set with a user-specified ratio.

- Supervised learning function (learn w/ training set)

  The training set is used for the learning procedure.

- Supervised learning function (evaluate w/ test set)

  The test set is used for the evaluation of the performance.

- Evaluation/Visualization of results

  Refer to Section 2.4 for the choice of appropriate evaluation/visualization utility.

Examples:
- Example_BP Network_Classification
- Example_BP Network_Curve Fitting
- Example_LVQ
- Example_SVM

2.1.3   Dimension reduction algorithms

Dimension reduction refers to the process of reducing the number of dimension of the data. The projection of the data set in the reduced space is often desired to preserve certain important data characteristics. In some cases data analysis, such as clustering, can be done more easily and accurately in the reduced space than in the original space. One prime application of dimension reduction is face recognition, where face images represented by a large number of pixels are projected to a more manageable low-dimensional "feature" space before classification.

List of functions:
- Isometric Feature Mapping (Isomap)
- Locally Linear Embedding (LLE)
- Multidimensional Scaling (MDS)
- Principal Component Analysis (PCA)
- Kernel PCA
- Linear Discriminant Analysis (LDA)

Conceptual diagram of usage of the MLT for dimension reduction
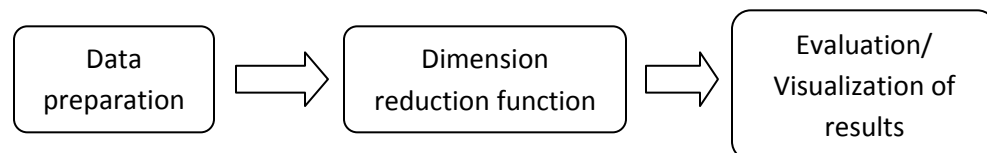
Data preparation → Dimension reduction function → Evaluation/ Visualization of results

- Data preparation

  Data need to be formatted to fit the API of the unsupervised learning function the user selects.

- Dimension reduction function

  A dimension reduction function is used to project the input data to a reduced space.

- Evaluation/Visualization of results

  Refer to Section 2.4 for the choice of appropriate evaluation/visualization utility.

Examples:
- Example_Manifold learning
- Example_LDA
- Example_Kernel PCA

2.2 Variant Data Type

For learning algorithms that require an input data type to be numeric, the data needs to be organized into a 2-D array of numeric numbers, where each row is an input sample. For learning algorithms that utilize the (dis)similarity relationships of the data samples as inputs, data samples can be any type of object. In this case, the input data needs to be organized into a 1-D array of variants. In addition, the user needs to specify the distance/kernel VI to use. Refer to Section 2.3 for the distance/kernel VI provided by the MLT.

Functions for which input data is a 1-D array of variants and a reference to a distance/kernel VI is a required input:
- k-medoids
- Hierarchical Clustering
- Spectral Clustering
- k-Nearest Neighbor (k-NN)
- Isometric Feature Mapping (Isomap)
- Locally Linear Embedding (LLE)
- Multidimensional Scaling (MDS)

Examples:
- Example_Clustering
- Example_Manifold learning

2.3 Distance/Kernel VI Reference

Some of the algorithms require the user to specify a distance/kernel VI. Refer to Section 2.2 for the list of applicable functions. The MLT provides some of the most frequently-used distances and kernel functions.

2.4 Validation & Visualization Utilities

The MLT provides validation and visualization utilities to facilitate the monitoring of the quality of learning. The utilities fall into three categories: cluster validity indices, evaluation of classification, visualization of learned results. The list of functions in each category is shown below.

Cluster validity indices:
- Rand Index
- Davies-Bouldin (DB) Index
- Jaccard Index
- Dunn Index

Evaluation of classification:
- Classification Accuracy
- Confusion Matrix

Visualization of learned results:
- Visualization (2D &3D)
- Plot SOM (2D &3D)

The applicability of each function to different algorithms is shown in Table I.

3. System Requirements

- Windows XP or later
- LabVIEW 2009 or later

4. Installation Notes

- Download and unzip the latest installer from NI Labs. Run **Setup.exe**.
- Launch LabVIEW so that the installed menus can rebuild.
- Open the diagram and go to **Addons >> Machine Learning**.

Table I. The applicability of validation and visualization utilities to different machine learning algorithms. "x" indicates that a utility is applicable to a certain algorithm.

			Validation Utility						Visualization Utility	
			Rand Index	DB Index	Dunn Index	Jaccard Index	Classification Accuracy	Confusion Matrix	Visualization (2D &3D)	Plot SOM (2D &3D)
Algorithm	Unsupervised Learning	k-means	x	x	x	x			x	
		k-medians	x	x	x	x			x	
		k-medoids	x	x	x	x			x	
		Gaussian Mixture Model	x	x	x	x			x	
		Fuzzy Cmeans	x	x	x	x			x	
		Hierarchical Clustering	x	x	x	x			x	
		Spectral Clustering	x	x	x	x			x	
		SOM							x	x
		VQ	x	x	x	x			x	
	Supervised Learning	k-NN					x	x	x	
		LVQ					x	x	x	
		SVM					x	x	x	
		BP neural network					x	x	x	
	Dimension Reduction	Isomap							x	
		LLE							x	
		LDA							x	
		MDS							x	
		PCA							x	
		Kernal PCA							x	