

A human head motion monitoring system based on an inertial measurement unit

Kanishka Jayawardene



Department of Electrical & Computer Engineering
McGill University
Montreal, Canada
May 2012

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Engineering.

© 2012 Kanishka Jayawardene

Abstract

This thesis describes the design and implementation of a human head motion monitoring system based on an inertial measurement unit. The system is to be used by physicians to characterize the head motion when engaging in day-to-day activities before and after corrective surgery is performed on the vestibular system. This system is also to be used by personnel in sports medicine to compare performance between athletes and by coaches to help athletes improve their techniques.

The design is implemented using an inertial measurement unit with an accelerometer, a gyroscope, and a magnetometer. Data can be logged on an onboard micro-SD card while transmitting data and receiving commands wirelessly. In doing so, several signal processing techniques such as finite impulse response filters and sensor fusion using Kalman filters are presented. All sensors are calibrated to ensure accuracy and reliability. In addition, this thesis focuses on pattern recognition techniques based on the Bayesian classification method to distinguish different daily activities of users.

Résumé

Cette thèse décrit la conception et le développement d'un système d'enregistrement des mouvements de la tête reposant sur une unité de mesure inertielle. Le système doit être utilisé par des médecins pour caractériser les mouvements de tête avant et après une chirurgie correctrice effectuée sur le système vestibulaire alors que le patient est engagé dans ses activités journalières. Ce système doit également être utilisé en médecine du sport afin de comparer les performances entre athlètes et les entraîneurs pour aider les athlètes à améliorer leurs techniques.

Le système comprend une unité de mesure inertielle avec un accéléromètre trois axes, un gyroscope et un magnétomètre. Les données peuvent être enregistrées sur une carte micro-SD lors de la transmission de données et recevoir des commandes sans fil. Ce faisant, plusieurs techniques de traitement du signal tels que des filtres à réponse impulsionnelle finie et la fusion de signaux en utilisant des filtres de Kalman sont présentés. Tous les capteurs sont étalonnés pour garantir l'exactitude absolue. De plus, cette thèse se concentre sur les techniques de reconnaissance de forme basée sur la méthode de classification Bayésienne pour distinguer les différentes activités quotidiennes des utilisateurs.

Acknowledgements

First, I would like to thank my supervisor, Dr. Zeljko Zilic, for giving me the opportunity to take up my thesis work under his direction. Furthermore, I would like to thank Dr. Kathleen Cullen from the Department of Physiology at McGill University for her advice on the applications of human motion monitoring.

I would also like to thank my fellow IML students Ben Nahill for his insightful advice on firmware development and Omid Sarbishei for the discussions I had with him regarding the project. In addition, I would like to thank Dr. Jerome Carriot from the Department of Physiology for his help with design decisions, getting access to measurement equipment used to calibrate the sensors, and collecting head motion data from various subjects. My thanks also go to Walter Kurcharski for his help with machine shop requirements.

I am also grateful to Sébastien Cros, the Canadian national women's short track coach, and numerous athletes from Speed Skating Canada for taking time out of their busy schedules to help us test the device while providing valuable feedback.

My special thanks go to my parents and my brother who have provided me with great support throughout my studies. Without their encouragement and love, this thesis would not have been possible.

Finally, my gratitude goes to the Natural Science and Engineering Research Council (NSERC) and McGill University for the financial support they provided for my graduate studies.

List of Acronyms

AHB	AMBA high performance bus
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced peripheral bus
CMSIS	Cortex Microcontroller Software Interface Standard
DMA	Direct memory access
FatFS	File allocation table file system
GPIO	General Purpose input output
GUI	Graphical user interface
HAL	Hardware abstraction layer
iNEMO	i N E rtial M odule
ISR	Interrupt service routine
NVIC	Nested vectored interrupt controller
RCC	Reset and clock control
RTOS	Real-time operating system

Contents

1	Introduction	1
1.1	Motivation.....	1
1.1.1	Vestibular System	1
1.1.2	Human Motion Monitoring.....	4
1.2	Design Requirements	5
1.3	Thesis Contributions and Organization.....	5
1.3.1	Contributions	5
1.3.2	Thesis Organization.....	7
2	Background	9
2.1	Related Work	9
2.1.1	Recent Research and Development.....	10
2.1.2	Commercial Body Motion Monitors	12
2.2	Overview of Hardware to be Used.....	15
2.2.1	Microcontroller	15
2.2.2	Inertial Measurement Unit (IMU)	16
2.2.3	Wireless Transceiver	19
2.3	Calibration and Signal Processing	22
2.3.1	The Least Square Method for Sensor Calibration	22
2.3.2	FIR Filter	24
2.3.3	Sensor Fusion using Complimentary and Kalman Filters	25
2.3.4	Tilt Angle Compensation	27
2.4	Statistical Analysis	29
2.4.1	Pattern Recognition	29
2.4.2	Training Algorithm	31
2.4.3	Other Pattern Recognition Algorithms	32
3	Overview of the Design Platform.....	34
3.1	High-level Design.....	34
3.2	Drivers	36
3.2.1	Data Storage.....	36
3.2.2	Wireless Connectivity.....	39
3.3	Low-level Software – Real-time Operating System	40
4	Design Implementation.....	42

4.1	Sensor Integration and Calibration	42
4.1.1	Accelerometer	43
4.1.2	Gyroscope	46
4.1.3	Magnetometer	49
4.1.4	Sensor Noise Distribution	50
4.2	Drivers - Data Storage	51
4.2.1	Flash Memory	51
4.2.2	SD Card	52
4.3	Drivers - Wireless Connectivity	54
4.4	RTOS-based Task Organization	56
4.4.1	Task Organization and Real-time Behaviour	56
4.4.2	Task Implementation	58
4.5	Filter Design	60
4.5.1	FIR Filter	60
4.5.2	Sensor Fusion	61
4.6	Chapter Summary	62
5	Applications.....	64
5.1	DizzyFIX Assistant.....	64
5.2	Activity Recognition	65
5.2.1	Classification	65
5.2.2	Identification	69
5.2.3	Patient Monitoring.....	70
5.3	Speed Skater Tracking	71
5.3.1	Data Acquisition	72
5.3.2	Head Orientation Estimation	75
5.3.3	Speed Estimation	77
5.3.4	Performance Comparison	81
5.3.5	Graphical User Interface	82
6	Conclusions and Future Work.....	84
6.1	Conclusion.....	84
6.2	Future work.....	85
	Appendix A.....	87
	Appendix B.....	89
	Appendix C.....	91
	References	93

List of Figures

Figure 1.1: The location of the vestibular system in the ear – available on [2].....	2
Figure 1.2: Structure of the vestibular system – open source on [3]	3
Figure 1.3: Roll, pitch, and yaw planes of motion with respect to human head [1].....	3
Figure 2.1: A photograph of the ST Microelectronic’s iNEMO platform.....	17
Figure 2.2: A photograph of the (a) stamp module, (b) STM32W-EXT evaluation board	20
Figure 2.3: The structure of the complimentary filter	26
Figure 2.4: The Kalman filter algorithm – adapted from [37]	27
Figure 2.5: Pattern recognition process – adapted from [45]	29
Figure 2.6: The expectation maximization (EM) algorithm [47]	32
Figure 3.1: Levels of abstraction of the design platform	35
Figure 3.2: Block diagram of the iNEMO module – adapted from [21]	35
Figure 3.3: The micro-SD card and STM32F10RE pin out – adapted from [21]	37
Figure 3.4: Fat FS layers – adapted from [42]	38
Figure 3.5: DMA block diagram – adapted from [40]	40
Figure 4.1: iNEMO module in protective case	43
Figure 4.2: Histogram of the accelerometer readings after calibration	45
Figure 4.3: Turntable used to calibrate the gyroscope	47
Figure 4.4: Comparison of the expected gyroscope reading and the measurements values about the x-axis	48
Figure 4.5: Comparison of the yaw angle computation using raw and calibrated magnetometer data and gyroscope data.....	50
Figure 4.6: Accessing the micro SD card via the SDIO interface	53

Figure 4.7: Packet structure for wireless transmission.....	54
Figure 4.8: Connections between the STM32W (wireless module) and the STM32F10E (iNEMO) micro-controllers.....	56
Figure 4.9: Top level state diagram.....	56
Figure 4.10: Keil logic analyzer signals	61
Figure 5.1: Accelerometer readings in <i>mg</i> for different activities	66
Figure 5.2: Local minima observed in vertical acceleration.....	67
Figure 5.3: Distribution of peak acceleration and period	69
Figure 5.4: Speed skating track (not to scale) [48].....	71
Figure 5.5: Speed skaters wearing the iNEMO	72
Figure 5.6: Accelerometer readings of four laps.....	73
Figure 5.7: Gyroscope readings of four laps	74
Figure 5.8: Head orientation estimation.....	76
Figure 5.9: Head orientation estimation (zoomed in)	76
Figure 5.10: Speed in each half lap	78
Figure 5.11: Speed as a function of time – improved using the Kalman filter.....	80
Figure 5.12: Comparing the head motion in two half laps	81
Figure 5.13: GUI showing the acceleration as a function of position on the skating track	82

List of tables

Table 2.1: Comparison of commercial body motion monitors	14
Table 2.2: Inertial measurement platforms comparison	18
Table 2.3: Wireless transceiver comparison	21
Table 3.1: Functions used from the FatFs module [42]	39
Table 4.1: Mean values of acceleration data in <i>mg</i>	45
Table 4.2: Kurtosis values of sensor noise	51
Table 5.1: Statistical characteristics of supervised and unsupervised training	68
Table 5.2: Sample data set to be classified	69
Table 5.3: Comparison of speeds (in m/s)	78

1 Introduction

1.1 Motivation

1.1.1 Vestibular System

The vestibular system in the inner ear is the natural human sensory mechanism that detects linear and rotational motion that the head experiences [1]. Not only that, it is also the driving force behind clear vision. In order to maintain the focus on an object, the eyes need to be fixed with respect to the target, and the vestibular system provides precise measurements that allow the muscles that control the eye movement to stabilize the image [2]. Distinct features in vestibular signals observed when people engage in various daily activities are used to design distinct pattern recognition systems that are unique to those activities. Furthermore, neck and head injuries resulting in concussions can affect athletes' spatial perceptions, and the ability to accurately capture head-motion data assists physicians during the rehabilitation process. A cross section of the ear is shown in Figure 1.1, and the vestibular system is marked with a red square.

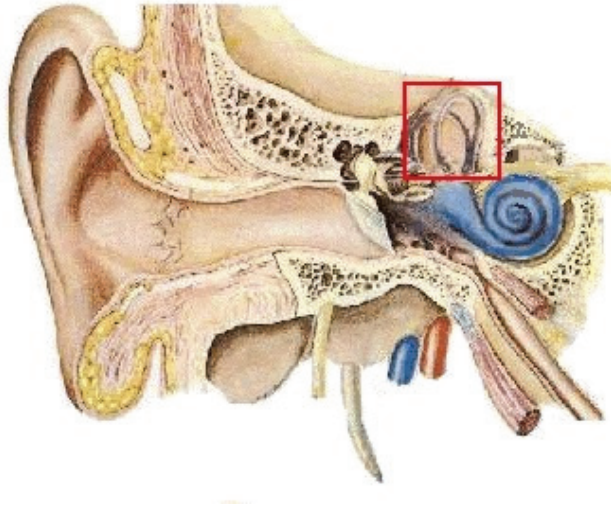


Figure 1.1: The location of the vestibular system in the ear – available on [2]

Head motion displays six degrees of freedom with translations and rotations about each of the three axes in three dimensional space. Translations are sensed by the saccule and utricle organs, and they sense the magnitude and the direction of gravity in a similar fashion to an accelerometer. The semicircular canals contain hair cells that are similar to the ones used for hearing, and when the head rotates, the fluids in these canals gain momentum and stimulate these hair cells with the response being proportional to the rate of rotation [2]. These organs are illustrated in Figure 1.2, and PC, SC, and HC refer to posterior, superior, and horizontal canals respectively [3].

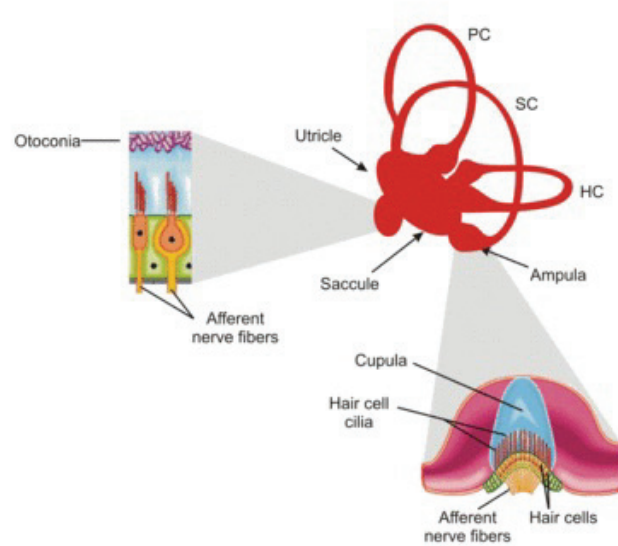


Figure 1.2: Structure of the vestibular system – open source on [3]

In order to make vestibular signals easier to analyze, the coordinate system shown in Figure 1.3 is used.

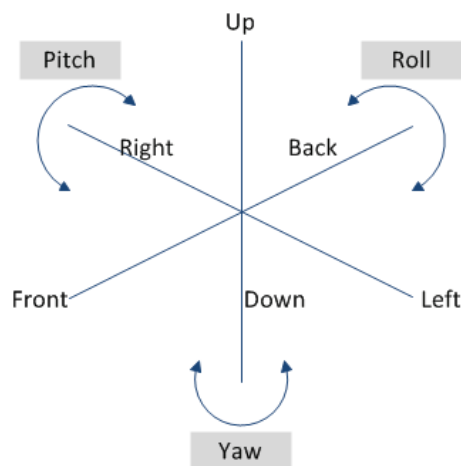


Figure 1.3: Roll, pitch, and yaw planes of motion with respect to human head [1]

1.1.2 Human Motion Monitoring

The functionality of the vestibular system can be affected by external influences such as head trauma when people are involved in automobile or sports accidents. Many patients need to undergo corrective surgery to regain the full control of their spatial perceptions. The ability to compare vestibular signals before and after surgery greatly assists physicians during the rehabilitation process [1].

Furthermore, the analysis of head motion can be used in sports medicine. Knowing which head motions are associated with the best performance in activities such as running, speed skating and skiing can help athletes and coaches improve their performance.

Generally, camera based motion capturing systems are used to monitor head motion. However, such laboratory based monitoring significantly reduces the distance a person can walk and the length of time the tests can be conducted [4]. Furthermore, camera based systems pose significant challenges when they are deployed to monitor outdoor sports such as skiing. Tracking speed skaters using multiple camera systems has been proposed in literature [5]. However, these camera-based systems are associated with high costs and arduous set-ups.

The need for a simpler vestibular signal replication system arises, and wearable sensors offer a good solution. Since the vestibular system detects linear and rotational motion, inertial measurement units with accelerometers and gyroscopes can be used to monitor the head motion of both patients and athletes.

1.2 Design Requirements

In order to replicate vestibular signals, it is necessary to obtain precise accelerometer and gyroscope measurements at high sampling rates. The device needs to have a small form factor and light weight allowing it to be worn on the head for long periods of time. It is also important that the device has long battery life to maximize the time patients and athletes can be monitored before recharging the device. In doing so, data should be stored on an onboard memory module such as a Secure Digital (SD) card. In order to allow the physicians to monitor patients in a clinical environment, wireless streaming of sensory information becomes very useful.

1.3 Thesis Contributions and Organization

1.3.1 Contributions

The work undertaken for this thesis can be summarized by the following problem statement and the contributions.

Problem Statement:

The objective of this thesis is to implement a standalone inertial measurement system that researchers and physicians in physiology and sports medicine can use to monitor patients and athletes.

Contributions:

1. This thesis implements firmware for the iNEMO inertial measurement unit and the STM32W wireless module and optimizes them to be used to replicate vestibular signals. A sensor calibration scheme is included in the firmware to ensure reliability and accuracy.
2. An investigation of how head motion can be used to distinguish daily activities is presented, and an activity recognition scheme based on the Bayesian classification technique is developed. The system is to be used by physicians to characterize the head motion when engaging in day-to-day activities before and after corrective surgery is performed on the vestibular system. Studies of this nature have not been performed in literature to the best of the author's knowledge.
3. A tool to assist Speed Skaters is implemented, which includes:
 - a. Monitoring the head motion of athletes
 - b. Estimation of speeds
 - c. A graphical tool to compare inertial data of each lap

In doing so, Kalman filters are used for sensor fusion and for speed estimation. Knowing which head motions are associated with the best performance can help athletes improve their performance. The use of head motion data based on inertial measurements to track speed skaters is not found in literature to the best of the author's knowledge.

1.3.2 Thesis Organization

The work presented in this thesis has two major branches; namely, firmware development and applications. Firmware development includes the configuration of accelerometers, gyroscopes, and magnetometers; implementation of a sensor calibration scheme; real-time operating system based task organization; and the implementation of data storage and wireless connectivity. The application development focuses on the study of head motion related to day-to-day activities and the implementation of a graphical user interface as a tool for speed skaters to analyze their head motion statistics.

Given the design requirements, this thesis is organized as follows. Chapter 2 performs a thorough analysis of recent developments on sensor modules that can be worn on different parts of the body and their applications. This chapter also compares several commercially available body motion monitors and assesses their strengths and shortcomings. The last section of the chapter inspects hardware platforms that can be used to develop the head motion monitoring system and compares their specifications. Wireless transceivers that can be used to establish wireless sensor networks are also evaluated.

Chapter 3 provides an overview of the design platform along with the high-level design tasks. Furthermore, details of the existing firmware employed to make the design process smoother are presented. These include the firmware for data storage, wireless connectivity, and the real-time operating system.

Chapter 4 presents the details of design implementation. First, a calibration scheme for accelerometers, gyroscopes, and magnetometers is presented. Next, the results of driver implementation and task organization are presented. The next section dedicated to low-level software development, which includes signal processing using finite impulse response filters, complimentary filters, and Kalman filters.

Chapter 5 focuses on applications of the head motion monitor, and it presents an activity recognition scheme based on head motion data. The device is also to be used to track the head motion of speed skaters, and such data is analyzed in this chapter to provide useful feedback to physicians, coaches, and athletes.

Chapter 6 presents a summary of the thesis and proposes future extensions to the project.

2 Background

In order to provide background to the work done on this project, a review of recent research and development in human body motion monitoring and related applications is presented in this chapter. Also included in this section is an overview of which microcontroller was to be used, as well as an evaluation of possible inertial measurement units (IMU) and wireless transceivers considered during the design process. Section 2.3 introduces the theoretical background that is used throughout this thesis. An overview of the least square method, finite impulse response filters, Kalman filters, and tilt angle compensation methods is presented. Section 2.4 presents background information about statistical analysis of head motion tracking data.

2.1 Related Work

The review of related work is performed in two parts. First, an overview of developments found in academic research is presented. Applications such as the use of IMUs to monitor core body motion and to analyze walking are commonly found in literature. Next, commercially available body motion monitors are compared, and it is shown that most of them are used to monitor core body and limb motion. The use of IMUs to monitor head motion during sports such as speed skating is not found in literature to the best of the author's knowledge.

2.1.1 Recent Research and Development

Monitoring human body motion has been of great interest to people of various disciplines over the past few years. Research has been directed towards applications in healthcare, entertainment, and education [4]. Body area sensor networks (BASN) and wireless sensor networks (WSN) provide non-invasive methods to track the physical behavior of people while using low-power radios to receive and transmit data. Barth et al. [6] have proposed a body area sensor platform called “TEMPO” (Technology Enabled Medical Precision Observation) that measures linear acceleration and rate of rotation about three axes. “TEMPO” is used by many research groups, but one of the shortcomings is that all collected data needs to be transferred wirelessly to a host machine in order to be stored. This can be a limiting factor when the head motion tracking device is to be worn throughout the day while engaging in day-to-day activities.

Wearable sensor modules can be used for *gait analysis*, and Chen et al. [4] have proposed a gait evaluation system called “LEGSys” (Locomotion Evaluation and Gait system), which uses 9-degree-of-freedom inertial sensors to analyze leg movement. The advantage of this system is that it allows gait analysis experiments to be conducted outside of laboratory environments in order to capture natural gait patterns. Najafi et al. [7] have used the LEGSys device to provide experimental results and to analyze the reliability of the system. The accelerometers used in the LEGSys system have a maximum range of $\pm 2g$, and more notably, the weight of the system is nearly 200g where the sensor, the data unit, and the battery weigh 10g, 80g, and 104g respectively. While such

weight is not a concern for a device that is worn on legs, the weight is one of the major design constraints when the device is to be worn on a person's head.

Another aspect of gait analysis is estimating walking speed. Vathsanam et al. have proposed such speed estimation by using a Gaussian Process-based Regression (GPR) technique and have compared the performance with Bayesian Linear Regression (BLR) and Least Squares Regression (LSR) [8]. The limitation here is that experiments were performed in a laboratory, so the resemblance to natural behavior has been removed from the results they have presented. Neural network based speed estimation from tri-axial accelerometer data has been proposed by Yoonseon et al. where they use an accelerometer with a range of $\pm 3g$ [9]. Although the sensor module is only 23g, there is an additional data collection device that must be worn on a belt as well. S. Chen et al. have proposed a TEMPO based gait analysis technique in [10] and [11]. These works of literature draw attention to the study of gait analysis focusing on knee joints angle.

Activity and gesture recognition is another common application of wearable sensors. Ravi et al. [12] have analyzed several classification methods for accelerometer based activity recognition, and some of these include decision trees, decision tables, k-nearest neighbors, and naïve Bayes. Suutala et al. have presented a daily activity recognition system based on support vector machines (SVM) [13].

Accelerometer-based athlete monitoring is presented in [14] – particularly for swimmers and rovers. The periodic nature of these activities was used to characterize performance. Accelerometers were also used in boxing gloves to analyze punch forces,

speed, and the number of punches [15]. These results were used to predict fight outcomes in real-time.

Monitoring athletes such as speed skaters using a camera-based technique has been proposed by Liu et. al [5]. However, these camera-based systems are associated with high costs and arduous set-up. Furthermore, camera-based systems pose significant challenges when they are deployed to monitor outdoor sports such as skiing. Inertial measurement units (IMUs) with accelerometers and gyroscopes provide a low-cost alternative to camera-based methods for monitoring athletes. Estimating translational speeds is a significantly more challenging task for speed skating. In general, IMU speed predictions, obtained using integration methods, are fused with the speeds obtained from GPS data [16]. However, GPS data is not available for indoor use, which is the case with speed skaters.

2.1.2 Commercial Body Motion Monitors

There are several body motion monitors on the market today, each characterized by a number of strengths and weaknesses. One of the most notable devices is the Opal body motion monitor by APDM. This lightweight inertial measurement device is sold in the form factor of a wrist watch and is capable of detecting high speed rotations, but the input range of the accelerometer is limited to only +/- 6g, and these devices cost nearly \$2400 [17].

Xsens MTw devices have excellent input ranges for both accelerometers and gyroscopes, but the noise coming from the accelerometer readings is very high. The biggest drawback of Xsens MTw is that it only allows wireless streaming of data [18]. As per one of the requirements of observing natural statistics of head motion, data storage is of utmost importance.

Motion Node Bus has similar sensor specifications to that of Opal and provides both data logging and wireless streaming. The downside is that the complete system, which includes a sensor module, a controller, and a battery pack, has a total weight of 270g [19]. The weight of the system is what keeps the Motion Node Bus from being used by physicians extensively. The Mensense Nano IMU also comes with good sensors, but it can only be used with a wired connection to a portable computer to collect data [20].

KinetiSense by Cleveland Medical Devices Inc. is a good choice of body motion device, but just like Opal, its accelerometer has a small input range. Also, the input noise is very high [21].

Table 2.1 summarizes the main features the commercially available body motion sensors described in this section.

Table 2.1: Comparison of commercial body motion monitors

Device		Opal [17]	Xsens MTw [18]	Motion Node Bus [19]	Memsense Nano [20]	KineticSense [21]
Physical	Dimensions	48.5x36x12 mm	34.5x57.8 x 14.5	35 x 35 x 15	46.5 x 13.8 x 22.8	
	Weight (g)	22	27	10 + 80 + 180 *	20 +	30
Accelerometer	Sample Rate (Hz)	128	100	100	50	128
	Resolution (bits)	14		14		12
	Range (+/- g)	6	16	6	10	5
	Noise (μ g/VHz)	128	305.8	150	210	1341.64
Gyroscope	Sample Rate (Hz)	128	100	100	50	128
	Resolution (bits)	14		14		12
	Range (deg/s)	1500	1200	2000	1200	1100
	Noise (deg/s/VHz)	0.07 - 0.126	0.05		0.56	0.58
Data storage	Memory (GB)	8	none	4	none	
	Equivalent time	28 days				30 hours
Battery life	Data logger (hours)	16	0	4.75 - 7	none	15
	Wireless (hours)	8	3.5			3
Wireless	Wireless range (m)	10	20 - 50	100		30
	Wireless data rate (kbps)					57.6

* sensor + controller + battery

2.2 Overview of Hardware to be Used

This section presents an evaluation of hardware that can be used to develop the head motion monitoring system and how the design decisions were made to meet the requirements presented in section 1.2. In doing so, the characteristics of microcontrollers based on the Cortex-M3 architecture are described in section 2.2.1. Several IMUs and wireless transceiver modules are compared in sections 2.2.2 and 2.2.3 respectively.

2.2.1 Microcontroller

ARM Holdings plc's Cortex family 32-bit microprocessor architecture is used industry wide due to its high performance and low power consumption. Microcontrollers based on the ARM architecture are used extensively in consumer electronic devices such as mobile phones, televisions, ebook readers, and tablets [23].

One of the attractive features of Cortex processors is that they are supplied with a vendor independent hardware abstraction layer (HAL) called Cortex Microcontroller Software Interface Standard (CMSIS). CMSIS governs how vendors should design their microcontrollers based on the processor core, and as a result, moving firmware from one vendor's microcontroller to another becomes easier. In addition, CMSIS provides support for real-time operating systems (RTOS), which can maximize the processor performance [23].

Interrupt-based firmware development is highly encouraged in embedded systems, and Cortex-M processors are equipped with a nested vectored interrupt controller (NVIC) to enhance the processor's interrupt handling capabilities. As the name suggests, the address of the function to be executed (also known as the interrupt handler) is stored in a vector table, and the processor refers to the table when an interrupt has taken place [23].

2.2.2 Inertial Measurement Unit (IMU)

There are various inertial measurement development units/modules on the market, and it is important to choose a unit that has a small form factor and that has uncompromised performance. Analysis of head movement requires accelerometers and gyroscopes, and it is also important to note that gyroscopes require an order of magnitude more power than accelerometers [6]. Interfaces to connect a wireless transceiver and a micro SD card are essential to meet the needs described in section 1.2.

ST Microelectronic's STEVAL-MKI062V2: **iN**ertial **M**odule, also known as the iNEMO, provides a viable option as the main development unit for this project. The embedded processor is an ARM Cortex-M3 architecture based STM32F103 microcontroller. This unit also includes a LSM303DLH geomagnetic module that provides accelerometer data in an input range of +/-8g with 14-bit fixed point accuracy [24]. Roll and pitch measurements are provided by an LPR430AL gyroscope while yaw axis data are provided by an LPR430AH gyroscope. The roll and pitch gyroscope is capable of detecting

rotations up to $1200^{\circ}/\text{sec}$ while the yaw axis gyroscope can detect rotations up to $300^{\circ}/\text{sec}$ [24]. One of the major advantages of the iNEMO is the availability of a micro SD card slot, which is connected to the microprocessor via the SDIO interface. In addition, STMicroelectronics provides a lot of example code and drivers to access the accelerometer, the gyroscope, and the SD card, which can be used as a base for firmware development. Figure 2.1 shows the iNEMO along with a Canadian 10 cents coin for size comparison.

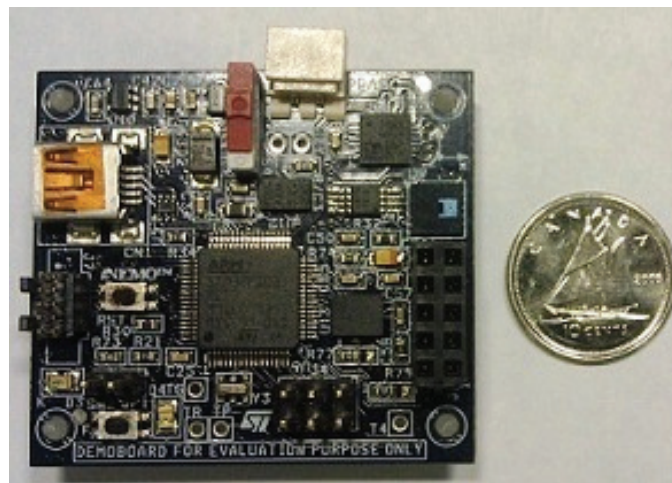


Figure 2.1: A photograph of the ST Microelectronic's iNEMO platform

The next generation of the iNEMO, which is yet to be released to the market, is presented in [25]. The IMM board is a surface mount device (SMD) with a very small form factor. Although accelerometer and gyroscope measurement ranges have both been improved, the lack of a micro SD card slot and a debugging interface such as a JTAG connector give rise to the need for a custom PCB.

SEN-08726 from Sparkfun and ADIS16400 from Analog devices offer inertial measurement units with respectable accelerometers and gyroscopes; however, both units lack a micro SD slot, which is a design requirement for this project. Table 2.2 summarizes the specifications of the inertial measurement units considered for the human head motion monitoring system.

Table 2.2: Inertial measurement platforms comparison

	Device	ST iNEMO [24]	ST IMM [25]	SEN – 08726 [26]	Analog ADIS16400 [27]
Hardware	Processor	ARM CM-3	ARM CM-3	ARM 7	BF 533
	Clock (MHz)	72	72	60	600
	RAM (kB)	64	64	32	128
	Dimensions (mm)	30 x 40 x 15	20x17x5	45 x 51 x 25	23 x 23 x 23
	Weight (g)	20		27	16
	Programming	JTAG /SWD	n/a	UART	
Accelerometer	Sample Rate (Hz)	50, 100, 400, 1000	100	150	330
	Resolution (bits)	14	14	14	14
	Range (+/- g)	8	16	6	18
Gyroscope	Sample Rate (Hz)	100	100	140	330
	Resolution (bits)	14	14	14	14
	Range (deg/s) [Yaw]	300	2000	500	300
	Range (deg/s) [Pitch/Roll]	1200	2000	500	300
Data storage and transmission	SD card slot	yes	none	none	none
	Interface for wireless	UART/SPI	UART	UART	SPI
Price (\$)		281.49	n/a	299.95	437.85

Comparing the IMUs presented in this section, it is evident that the iNEMO and ST IMM are both great choices. The lack of an SD card slot and the fact that the ST IMM device is yet to be released to the market makes its use limited. On the other hand, iNEMO's accelerometer and gyroscope sensing ranges are adequate for human head motion monitoring [1], and the availability of a micro SD card slot, along with an interface for a wireless transceiver makes the iNEMO the best choice IMU for this project.

2.2.3 Wireless Transceiver

In a clinical environment, it is beneficial for physicians to see the precise head movement of patients. It is also of equal importance to start recording data simultaneously when multiple devices are used. Wireless transceivers are required to facilitate such operation of the head motion monitor. The characteristics of several wireless transceivers are compared in order to determine the best fit wireless transceiver to be used with the iNEMO.

STM32W wireless transceivers offer a low power solution to both data streaming and providing commands to the motion monitor. The advantage is that the transceiver is supplemented by an onboard Cortex-M3 microprocessor, which can be used to reduce the computations that the main processor on the motion monitor needs to carry out. In addition, ST Microelectronics provides libraries that handle a lot of error corrections associated with wireless transmission [28]. The other advantage of STM32W is that it is bundled with an evaluation board that assists firmware development. Once all software

is tested, a stamp module of the size 25x20x3mm can be removed and placed with the motion monitor. The STM32W-EXT evaluation board and the stamp module are shown in the following figure along with a Canadian 10 cents coin for size comparison.

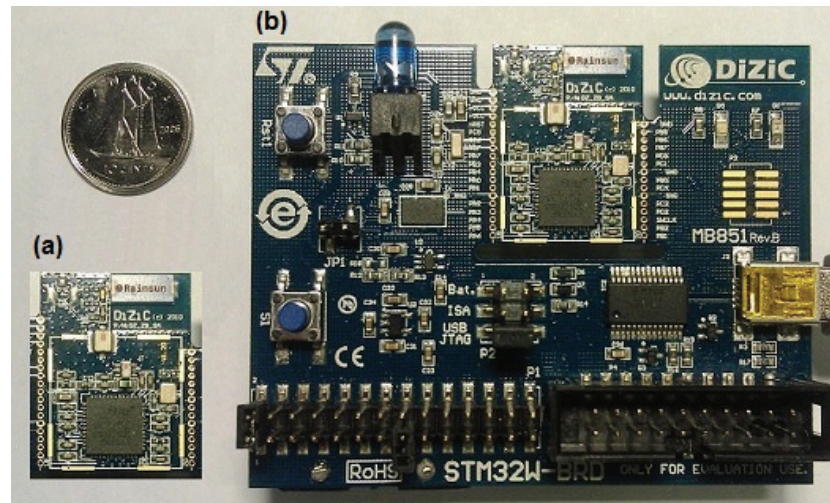


Figure 2.2: A photograph of the (a) stamp module, (b) STM32W-EXT evaluation board

XBee ZB is an extensively used wireless transceiver that provides users with the opportunity to configure the device with the use of a graphical user interface (GUI). The downside to its simplicity and low cost is that it draws more current than other devices that offer the same transmission distance [29].

EcoMote is the smallest and the lightest device evaluated, and it provides a data rate of 1Mbps, which is four times as high as the data rate of STM32W and XBee [30]. The drawback is that the cost of a unit is \$2000, which is the highest hurdle in developing a low cost head motion monitor. Xtreme OEM from Digi International Inc. is capable of

long-distance transmission; however, the data rate is very low and the current consumption is high. The transmission current of Xtreme OEM is 150mA whereas the STM32W uses just 24mA for transmission [28], [31].

The next best contender for the wireless transceiver solution is Texas Instruments' CC2500 based MSP430 Wireless Development Tool (MSP430 WDT). Similar to the STM32W kit, the MSP430 WDT kit is comprised of an onboard microcontroller, which can be used to perform computations that are necessary for wireless transmission [32].

A comparison of the wireless transceivers evaluated is tabulated in Table 2.3.

Table 2.3: Wireless transceiver comparison

	Device	STM32W [28]	XBee ZB [29]	EcoMote [30]	Xstream OEM [31]	TI CC2500 [32]
Radio	Protocol	ZigBee	ZigBee			ZigBee
	Frequency (GHz)	2.4	2.4	2.4	0.900 and 2.4	2.4
	Data rate (kbps)	250	250	1000	19.2	500
	Rx current (mA)	20	38	21	50	16.6
	Tx current (mA)	24	35	10	150	21
	Range (m)	40	40	10	450	40
Microprocessor	Model	STM32 Cortex-M3	n/a	8052	n/a	MSP 430
	Frequency (MHz)	24	n/a	16	n/a	16
	Bits	32	n/a		n/a	16
Physical	Dimensions (mm)	70x55x15 (*stamp at 25x20x3)	32x22x3	13x11x7	40x71x9	
	Weight (g)	15	20	1.8	24	40
	Battery	2xAAA	external 3V	40mAh LiPoly	external 5V	2xAAA or external

The analysis of the features of the wireless transceivers presented in the above table suggests that STM32W and TI CC2500 are the best two choices. The STM32W device holds the advantage because it uses a Cortex-M3 32-bit microprocessor, which is also used in the iNEMO. This reduces the time spent on firmware development, so the STM32W device was chosen as the wireless transceiver for the head motion monitoring system.

2.3 Calibration and Signal Processing

The objective of this section is to present the theoretical background used for sensor integration. First, the least square method used to calibrate the iNEMO sensors is described. Then, basic theory behind the implementation of finite impulse response (FIR) filters and Kalman filters is discussed. FIR filters are required to filter out the high frequency noise of accelerometers and magnetometers and the low frequency drifts of gyroscopes. Kalman filters are used to fuse readings from multiple sensors in order to obtain meaningful outputs such as orientation angles and translational speeds. Next, equations used to compute orientation angles of the IMU and rotation matrices used to transform the coordinate system of IMU data are presented.

2.3.1 The Least Square Method for Sensor Calibration

The iNEMO-based head motion monitor can improve the quality of life of many people, and sensor reliability is of utmost importance when it is used in medical applications

[33]. The sensitivity of sensors can degrade over time, and usually the factory calibration varies from device to device. Therefore, it is necessary to calibrate and re-calibrate sensors periodically [33]. The least square method is commonly used to estimate parameters and to fit a function to a data set using an over-determined system of equations [34]. The optimum parameters found by the least squares method minimize the sum of residuals, which is given by,

$$S = \sum_{i=1}^N (y_i - x_i)^2 \quad (1)$$

Where y_i and x_i refer to the i^{th} value of the known fitting function and the data set respectively [34]. Data points y_i and x_i can be placed in matrices, and a matrix B can be defined to contain the parameters that need to be estimated. The matrix relationship can simply be stated as:

$$Y = X \cdot B \quad (2)$$

The calibration parameter matrix denoted by X can be computed as

$$B = [X^T \cdot X]^{-1} \cdot X^T \cdot Y \quad (3)$$

Equation (3) can be used to determine calibration parameters for the accelerometer and the gyroscope. In the case of the accelerometer, the compensated values are related to the raw measurements as seen below [35]:

$$\begin{bmatrix} A_{x1} \\ A_{y1} \\ A_{z1} \end{bmatrix} = \begin{bmatrix} ACC_{11} & ACC_{12} & ACC_{13} \\ ACC_{21} & ACC_{22} & ACC_{23} \\ ACC_{31} & ACC_{32} & ACC_{33} \end{bmatrix} \cdot \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} + \begin{bmatrix} ACC_{10} \\ ACC_{20} \\ ACC_{30} \end{bmatrix} \quad (4)$$

A_{x1} , A_{y1} , and A_{z1} are the known acceleration values while A_x , A_y , and A_z are the raw measurements and ACC_{x0} are the bias values. The entries on the diagonal (ACC_{11} , ACC_{22} , ACC_{33}) are the gain values of each of the axes, and the off diagonal entries refer to the

amount of crosstalk between axes. As suggested in [35], the above relationship can be rearranged to be more suitable for least square computation as follows:

$$[A_{x1} \quad A_{y1} \quad A_{z1}] = [A_x \quad A_y \quad A_z \quad 1] \cdot \begin{bmatrix} ACC_{11} & ACC_{21} & ACC_{31} \\ ACC_{12} & ACC_{22} & ACC_{32} \\ ACC_{13} & ACC_{23} & ACC_{33} \\ ACC_{10} & ACC_{20} & ACC_{30} \end{bmatrix} \quad (5)$$

The relationship described by eq. (5) can also be used to calibrate the gyroscope and the magnetometer on the iNEMO.

2.3.2 FIR Filter

The accelerometer and the gyroscope in the iNEMO are micro electro-mechanical systems (MEMS) based devices. Often, sensor readouts are corrupted by noise inherent to MEMS processes, as well as by noise caused by external influences such as mechanical vibrations [36]. In order to ensure the reliability of recorded data, it becomes necessary to filter out the high frequency noise of accelerometers and magnetometers and the low frequency drifts of gyroscopes. These filters need to be stable and have linear phase. Therefore, finite impulse response (FIR) filters based on the Kaiser Windowing technique are to be implemented [37].

The order of the FIR filter can be approximated by [37],

$$M \approx (A - 7.95)/(14.36 \Delta f) \quad (6)$$

Where A refers to the maximum sideband ripple and Δf is the normalized transition band defined by,

$$\Delta f = (f_c - f_r)/(f_s/2) \quad (7)$$

f_c , f_r , and f_s refer to cutoff, first stop band, and sampling frequency respectively. The parameter β used to define the shape of the Kaiser window is defined as [37]

$$\beta = 0.1102(A - 8.7) \quad (8)$$

2.3.3 Sensor Fusion using Complimentary and Kalman Filters

Complimentary Filter

In order to obtain the correct orientation at any given point, it is necessary to effectively fuse the measurements of the accelerometer, gyroscope, and the magnetometer. A complimentary filter combines the angle estimates obtained using accelerometers and integrating gyroscope readings by weighing them based on the dynamics of the system. Accelerometer based angles are filtered using a low pass filter, and gyroscope based angles are filtered using a high pass filter before summing the two angle estimates [39]. The structure of the complimentary filter is illustrated in Figure 2.3.

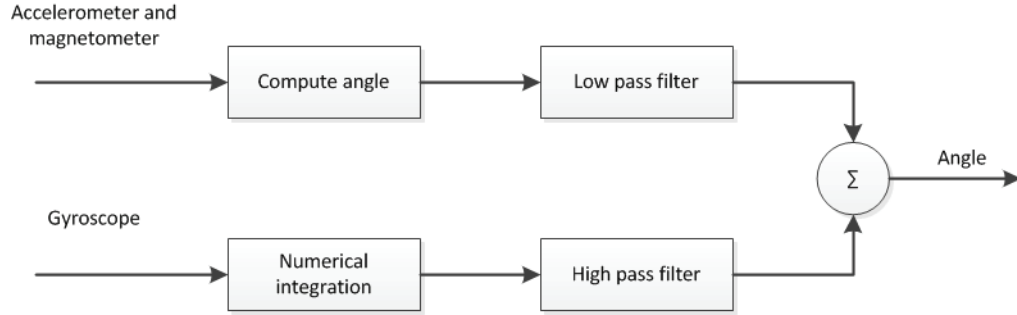


Figure 2.3: The structure of the complimentary filter

Kalman Filter

A standard Kalman filter estimates the states of a discrete-time process described by the following equations [40]:

$$x_{k+1} = A_k x_k + B_k u_k + w_k \quad (9)$$

$$y_k = H_k x_k + v_k \quad (10)$$

Where,

- A , B , and H are state transition, input, and output matrices respectively
- x_k and y_k are the state vector and the output at the k^{th} time step respectively
- u_k is the known control signal (input)
- w is the process noise with covariance matrix Q
- v is the measurement noise with covariance matrix R

The Kalman algorithm is described in Figure 2.4

Predict – time update
<ol style="list-style-type: none"> 1. Project the state ahead $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$ 2. Project the error covariance $P_k^- = AP_{k-1}A^T + Q$
Correct – measurement update
<ol style="list-style-type: none"> 1. Kalman gain $K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$ 2. Error of the estimated state $\hat{x}_k = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-)$ 3. Update the error covariance $P_k = (I - K_k H)P_k^-$

Figure 2.4: The Kalman filter algorithm – adapted from [40]

The Kalman filter can be used to fuse gyroscope, accelerometer, and magnetometer measurements to estimate the orientation angles. In addition, the Kalman algorithm can be used to estimate the speeds of speed skaters by fusing lateral and forward acceleration measurements as described in section 5.3.3.

2.3.4 Tilt Angle Compensation

The iNEMO-based human motion tracking device is to be used by persons of various disciplines; therefore, it is not easy to guarantee that they will always align the device

perfectly with the global x, y, and z coordinates. As such device should be intelligent enough to compute the initial orientation using data from the first few seconds of measurement and adjust subsequent measurements accordingly. It is assumed that the device is stationary when the device is turned on. Therefore, the orientation angles can be estimated using the accelerometer readings as shown by the following equations [36]. The roll and pitch angle axes are as defined in Figure 1.3.

$$Roll = \alpha = \tan^{-1} \left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}} \right) \quad (11)$$

$$Pitch = \beta = \tan^{-1} \left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}} \right) \quad (12)$$

The heading angle, defined as the rotation from the magnetic north, cannot be computed using only accelerometer data, and therefore, magnetometer data are also used [36].

$$M'_x = M_x \cos(\alpha) - M_y \sin(\beta) \sin(\alpha) + M_z \cos(\beta) \sin(\alpha) \quad (13)$$

$$M'_y = M_y \cos(\beta) + M_z \sin(\alpha) \quad (14)$$

$$\gamma = 90 - \tan^{-1} \left(\frac{M'_x}{M'_y} \right), \text{ when } M'_y > 0 \quad (15)$$

$$\gamma = 270 - \tan^{-1} \left(\frac{M'_x}{M'_y} \right), \text{ when } M'_y < 0 \quad (16)$$

The following rotation matrices can then be used to project the sensor readings onto the global coordinate system for further processing [10].

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix} \quad (17)$$

$$R_y(\beta) = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \quad (18)$$

$$R_z(\gamma) = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (19)$$

2.4 Statistical Analysis

Head motion data collected using the iNEMO during different day-to-day activities is used to analyze the characteristics of vestibular signals. Statistical analysis is the basis for activity recognition, gait analysis, and comparing head motion data from one person to another. Similarly, statistical analysis can be used to differentiate the head motion of patients before and after corrective surgery. The theoretical background used for activity recognition is presented in this section.

2.4.1 Pattern Recognition

One of the major concepts of activity recognition is *pattern recognition* [48], and this process involves three major steps as seen in Figure 2.5.



Figure 2.5: Pattern recognition process – adapted from [48]

Once data has been collected, patterns need to be observed, and distinguishing features of each of the activities need to be extracted. The simplest method to identify a given task is to use the naïve Bayes classifier. The algorithm assumes that features that make up a class are independent, which makes computation simple and effective, and the algorithm performs very well even with dependent features [48]. The sample data set is referred to as evidence (E), and the probability of E being in class c is given by

$$p(c|E) = \frac{p(E|c) \cdot p(c)}{p(E)} \quad (20)$$

E is classified as class c if

$$f_b(E) = \frac{p(c|E)}{p(\bar{c}|E)} \geq 1 \quad (21)$$

Where $f_b(E)$ is called a Bayesian classifier and \bar{c} is the complement of class c [48]. The probabilities based on all features of the evidence sample can be computed as follows:

$$p(E|c) = \prod_i^n p(x_i|c) \quad (22)$$

Where $p(x_i|c)$ refers to the probability of each feature given the class. The probability density function for normal distribution can be used to compute each of the total probabilities in a matrix sense as follows.

$$p(X) = \frac{1}{(2\pi)^{n/2} \sqrt{\det(\Sigma)}} \cdot e^{-\frac{1}{2}(X-\mu_x)^T \Sigma^{-1} (X-\mu_x)} \quad (23)$$

Where,

- μ_x is the mean vector of the features

- Σ is the variance-covariance matrix
- n is the number of elements in the feature vector

2.4.2 Training Algorithm

The classification model used to define Gaussian parameters up to this point is called *supervised training*. Classification was done based on the knowledge that a particular training set belonged to a particular class, and this knowledge is known as *a priori*. However, the long term goal is for the iNEMO should be able to identify head motions of different users autonomously. If the available training data is not labelled, it should still be possible to separate data into classes, and this method is called *unsupervised training* [49]. An expectation maximization (EM) algorithms can be used to iteratively classify training data to facilitate unsupervised training.

The EM algorithm used in this project is based on the algorithm proposed by X. Guorong et al. in [50]. Given that there are M classes, the prior probabilities of all classes are assumed to be equal such that $P(c_m) = 1/M$. There are two main steps that are repeated until a stable solution is reached. Namely, the estimation and the maximization steps.

The estimation step computes the probability that each data point x_i belongs to class c_m^- with the parameter set λ . Each class is assumed to have a mean vector of μ_m and a covariance matrix of Σ_m , and the $-$ and $+$ signs refer to the old and the new values.

The maximization step involves updating the mean vector, the covariance matrix, and the prior probabilities.

Estimation
$P(c_m^- x_i, \lambda^-) = \frac{P(c_m^- \lambda^-) \cdot p(x_i \mu_m^-, \Sigma_m^-)}{\sum_{j=1}^M P(c_j^- \lambda^-) \cdot p(x_i \mu_j^-, \Sigma_j^-)}$
Maximization
<p>1. Update mean:</p> $\mu_m^+ = \frac{\sum_{i=1}^M x_i \cdot P(c_m^- x_i, \lambda^-)}{\sum_{i=1}^M P(c_m^- x_i, \lambda^-)}$ <p>2. Update covariance:</p> $\Sigma_m^+ = \frac{\sum_{i=1}^M P(c_m^- x_i, \lambda^-) \cdot (x_i - \mu_m^+) \cdot (x_i - \mu_m^+)^T}{\sum_{i=1}^M P(c_m^- x_i, \lambda^-)}$ <p>3. Update prior probabilities:</p> $P(c_m^+ \lambda^+) = \frac{1}{M} \sum_{i=1}^M P(c_m^- x_i, \lambda^-)$

Figure 2.6: The expectation maximization (EM) algorithm [50]

2.4.3 Other Pattern Recognition Algorithms

Several other pattern recognition algorithms are found in literature. Some of them include: support vector machines (SVM), logistic regression, k-means, and k-nearest neighbors. The SVM algorithm allows non-probabilistic classification, but one of the

short-comings is that training has to be supervised [13]. Since one of the long-term objectives of the design is to autonomously identify the head motion of different users, it is necessary to use an algorithm that supports unsupervised training. Logistic regression uses an iterative method to determine the maximum likelihood of regression coefficients, which results in a lack of convergence in some cases [51]. The k-means algorithm is initialized by randomly assigning input data into ' k ' classes. The means of each class are first computed. Next, each data sample is assigned to the class with the nearest mean [52]. The disadvantage of this method is that different initial classes can result in different final classes, and the process needs to be repeated to ensure reliable classification. Therefore, the Bayesian classification method along with the expectation maximization algorithm due to their simple, yet accurate implementation.

3 Overview of the Design Platform

The objective of this chapter is to present the high-level design of the iNEMO-based head motion monitor and to introduce existing firmware that was used to implement the system. First, the drivers needed to implement data storage and wireless connectivity are introduced. Next, an overview of the real-time operating system used in this project is presented.

The software development for the iNEMO and the STM32W wireless module was conducted using the Keil uVision 4 and IAR Embedded Workbench 6.21 Integrated Development Environments (IDE). Low-level drivers to access microcontroller peripherals such as I2C, ADC, DMA, SDIO, and UART were included with the installations of the IDEs, and these drivers were used to make the code as readable as possible.

3.1 High-level Design

The design of the iNEMO-based head motion monitor platform was performed in several levels of abstraction. Figure 3.1 illustrates how the overall design is organized. The areas highlighted in dark grey (also marked with *) are the main contributions of this thesis, and these are presented in Chapter 4. The areas highlighted in light grey (also marked with +) are the tasks that needed to be implemented with the aid of existing firmware to enable the seamless operation of the overall system, and these are presented in the remainder of Chapter 3.

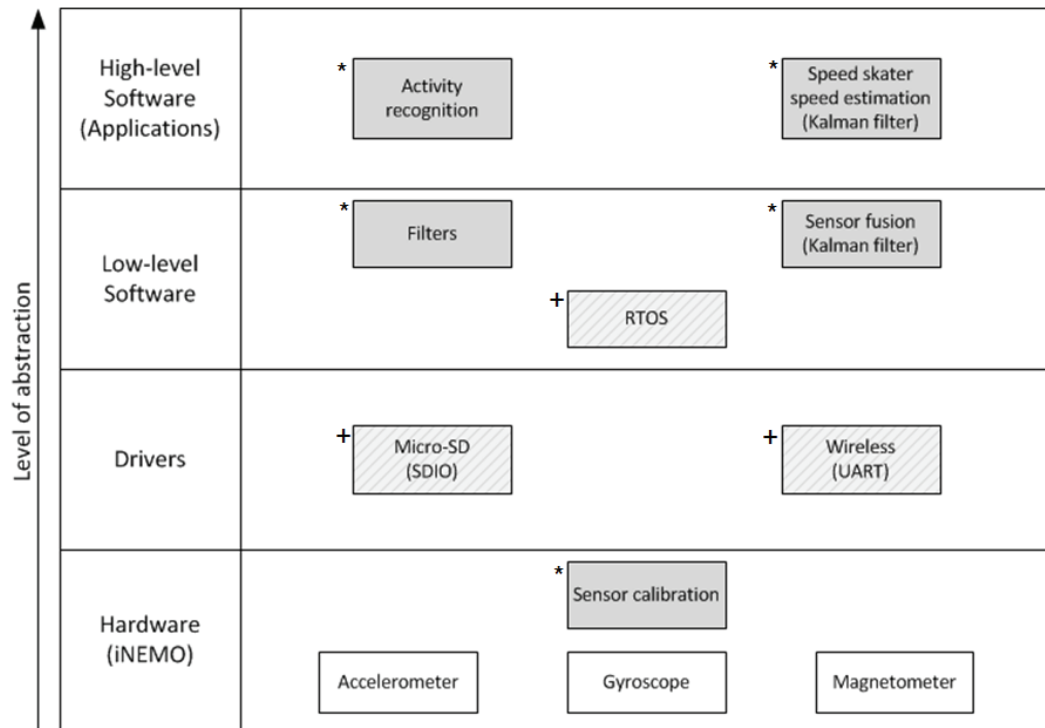


Figure 3.1: Levels of abstraction of the design platform

The platform designed in this thesis is based on the iNEMO, and the following figure illustrates its block diagram. Only the peripherals used in this project are shown.

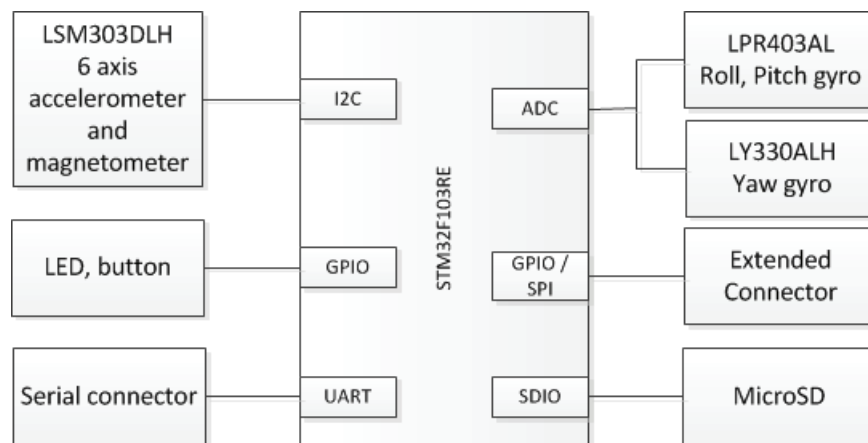


Figure 3.2: Block diagram of the iNEMO module – adapted from [24]

The sensors on the iNEMO need to be calibrated to ensure data accuracy and reliability, especially since the head motion monitor is to be used for medical applications. The calibration scheme is implemented in section 4.1. This platform is to be used for long-term head motion monitoring as well as for real-time monitoring in clinical environments. Therefore, driver-level implantations of data storage and wireless connectivity are presented in sections 3.2, 4.2, and 4.3. Significant amount of signal processing is required to draw useful conclusions from acquired data; therefore, filter design and sensor fusion are discussed. High-level application development to monitor patients and athletes, presented in Chapter 5, uses aforementioned tasks as its foundation.

3.2 Drivers

3.2.1 Data Storage

Flash Memory

The STM32F103RE microcontroller found on the iNEMO module is equipped with 512kB of flash memory, which is generally used to store program code. However, space unused by program code can be used to store sensor data and processed information. The maximum space available in memory-bank 1 is from address 0x08008000 to 0x807FFFF, which results in 491519-bytes of available space [43].

SD-Card Interface

The SDIO (secure digital input output) peripheral found in the STM32F103RE is wired to a micro-SD slot on the iNEMO module. Figure 3.3 illustrates the pin-out of the micro-SD card and the SDIO interface of the STM32F103RE microcontroller.

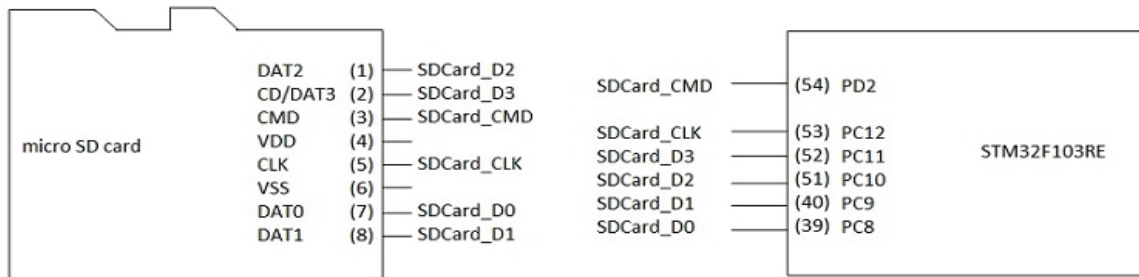


Figure 3.3: The micro-SD card and STM32F103RE pin out – adapted from [24]

The signals are prefixed with “SDCard_” for easier identification, and each device’s pins are presented within parentheses. The figure also suggests that GPIO C pins 8 to 12 are to be used to send data to the micro-SD card while GPIO D 2 pin is to be used to send commands.

The motion measurement system was designed with physicians, patients, and athletes being the intended end-users; therefore, data collection should be made as easy as possible. With this design requirement in mind, a file system was employed to store data on the micro SD card, and as a result, the user can simply insert the card into their computer to get required data files.

The two file systems, the Keil Flash File System (FFS) [44] and the FatFs library by ChaN [45], were considered for this project. The Keil FFS can be used to create Windows-compatible FAT8, FAT16, and FAT32 file systems, and is provided with MDK (Microcontroller Development Kit) Professional IDE distributions [44].

The open-source FatFs library is a module that is independent of the disk input-output layer, which made it an ideal candidate for this project [45]. Figure 3.4 illustrates the layers of software involved in implementing the FatFs module.

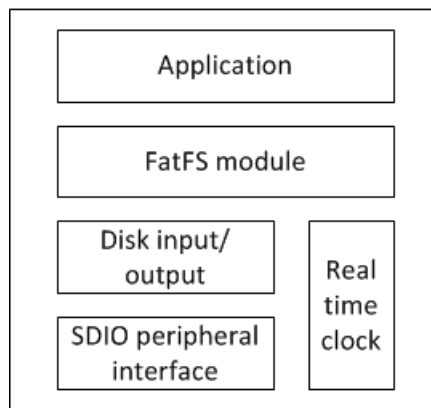


Figure 3.4: Fat FS layers – adapted from [45]

File input-output operations of the FatFs module are defined in a similar manner to those of standard C, and the following functions were used to store sensor information on the micro SD card.

Table 3.1: Functions used from the FatFs module [45]

Function	Description
f_mkfs	Create a FAT file system on the logical drive
f_open	Open a file object with mode flags such as FA_READ, FA_WRITE, FA_OPEN_ALWAYS, FA_CREATE_NEW
f_close	Close a file
f_write	Write data in a buffer to a file object
f_printf	Write strings to a file object
f_putc	Write a character to a file
f_sync	Flush cached information of file write commands

3.2.2 Wireless Connectivity

The iNEMO based human motion tracking device is required to transmit data wirelessly to a host PC when it is used in a clinical environment. In addition, a wireless sensor network can be established, and data from multiple body parts can be collected. In doing so, it is imperative that devices can synchronize with each other. The host computer should be able to send wireless commands to each sensor node, so they may perform actions such as streaming or logging data or both, recognizing human activities, and evaluating orientations.

Figure 3.5 illustrates the DMA block diagram configured for wireless transmission using the USART peripheral.

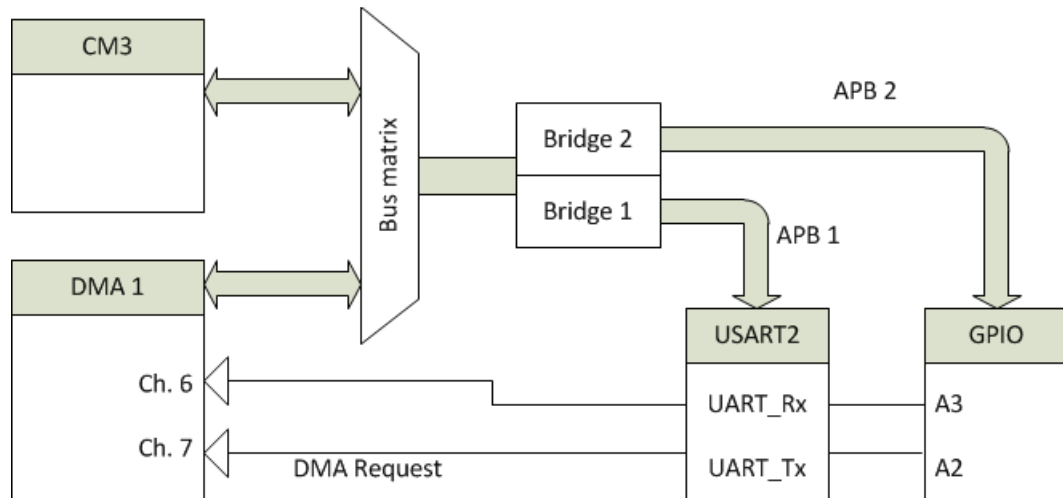


Figure 3.5: DMA block diagram – adapted from [43]

As shown in the figure, DMA channel 6 was connected to the UART_Rx pin and GPIO A3 pin while DMA channel 7 was connected to UART_Tx pin and GPIO A2 pin. The USART and the GPIO peripherals are wired to advanced peripheral bus (APB) 1 and 2, so the respective reset and clock control (RCC) drivers needed to be enabled as well.

3.3 Low-level Software – Real-time Operating System

Many simple embedded systems execute using the super-loop concept [41]. All functions are executed in a predetermined order within an infinite loop, and real-time requirements are achieved using interrupts. However, in complex projects, only using

interrupts to meet timing requirements can lead to complex interrupt service routines (ISRs), and if interrupts are nested, it becomes even more challenging. In addition, memory management and the development of fault-tolerant procedures can become very tedious, and as a result, the scalability of the project decreases [42].

A real-time operating system (RTOS) assists firmware development by handling all the resource management, so the developer can dedicate more time to application development. Furthermore, RTOS-based task scheduling results in optimum program flow and facilitates task concurrency [41].

Two RTOSs, FreeRTOS and Keil's RTX RTOS, were considered for this project; however, due to its open source nature, FreeRTOS was used.

4 Design Implementation

As mentioned in Chapter 3, the overall platform was designed in several layers of abstraction. Chapter 4 presents the details of the design implementations of the contributions illustrated in Figure 3.1 on page 35. At the hardware level, a sensor calibration scheme to ensure the accuracy of collected data is presented. Next, the implementation and the test results of driver-level tasks to implement data storage and wireless connectivity are presented. The low-level software development such as RTOS-based task management and filter design are presented. Last, the implementation of sensor fusion using Kalman filters is discussed.

4.1 Sensor Integration and Calibration

The first step in implementing the platform described in Figure 3.1 was to configure the sensors on the iNEMO. Absolute sensor data is a design requirement imposed by the physicians and researchers in physiology, and as a result, it is vital that the accelerometer, the gyroscopes, and the magnetometer on the iNEMO are properly calibrated [33]. Sensor calibration was performed using the least square method introduced in section 2.3.1.

4.1.1 Accelerometer

Based on the set-up introduced in section 2.3.1, a least square problem was defined to compute calibration parameters that handle misalignment, sensitivity, and bias [8]. The acceleration readings from six positions were used to construct the ‘known’ matrix used in the least squares method. These positions were x-axis up/down, y-axis up/down, and z-axis up/down.

The iNEMO was placed in a plastic case along with a lithium ion battery in order to facilitate the calibration process and to place on the helmets of Speed Skaters. The latter is discussed in section 5.3. The vertices of the box were machined to be 90° , and the iNEMO was precisely aligned with the box’s straight edges with the aid of a caliper. The box was placed on a level surface, verified using an engineer’s spirit level, and data samples for positions related to the x-axis and the z-axis were recorded. The following figure illustrates the device in the protective case.

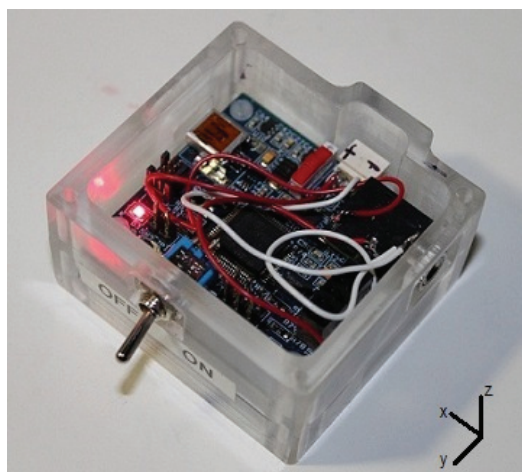


Figure 4.1: iNEMO module in protective case

Collecting data for the y-axis up and down positions was challenged by the presence of the switch and the protrusion on the back side. However, the clamp seen in Figure 4.3 and an engineer's spirit level was used to ensure the y-axis of the iNEMO properly aligned with the earth's gravity vector.

For each accelerometer position, 3000 data samples at 100Hz were collected, and a least square parameter estimation problem was set up according to the description in section 2.3.1. The system was solved using Matlab, and the following parameter matrices were obtained.

$$\begin{bmatrix} ACC_{11} & ACC_{12} & ACC_{13} \\ ACC_{21} & ACC_{22} & ACC_{23} \\ ACC_{31} & ACC_{32} & ACC_{33} \end{bmatrix} = \begin{bmatrix} 0.9593 & 0.0044 & 0.005 \\ 0.0487 & 0.9624 & 0.0068 \\ -0.0046 & -0.0149 & 0.9607 \end{bmatrix}$$

$$\begin{bmatrix} ACC_{10} \\ ACC_{20} \\ ACC_{30} \end{bmatrix} = \begin{bmatrix} -0.0178 \\ 0.0118 \\ 0.0203 \end{bmatrix}$$

The above calibration parameters apply to accelerometer readings in g . The validity of the calibration parameters was verified by applying them to a new set of measurements with known accelerations of -707 , -707 , and 0 mg in x , y , and z directions respectively. To do so, the box shown in Figure 4.1 was mounted at an angle of 45° using the clamp shown in Figure 4.3, and this angle was verified using a set square. The following is a histogram of the calibrated data set, and it should be noted that these values were sampled at 100Hz and filtered using a low pass FIR filter with a cut-off frequency of 20Hz.

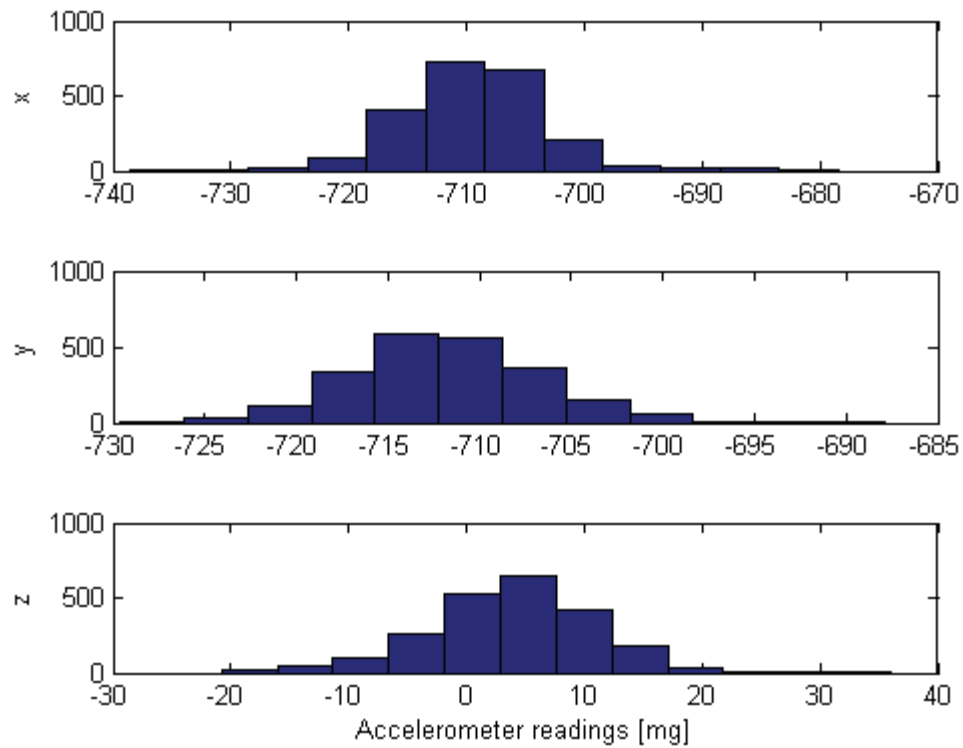


Figure 4.2: Histogram of the accelerometer readings after calibration

The preceding figure suggests that accelerometer readings exhibit a near Gaussian distribution, which will be further discussed in section 4.1.4. The mean values of the readings are tabulated below:

Table 4.1: Mean values of acceleration data in *mg*

Axis	X	Y	Z
Pre-calibration	-722.9606	-720.5112	-13.0563
Post-calibration	-709.5025	-711.8439	3.8047
Error (%)	0.39	0.69	n/a

It is evident that the least square calibration method gives parameters that result in accelerometer readings that are within less than 1% of the expected mean values. The computed calibration parameters were included in the firmware, so the end user does not have to recalibrate the device prior to using it to collect head motion data. However, it should be noted that these calibration parameters are valid only for the iNEMO used in this experiment and that any time a new device is used, a new set of calibration parameters is required.

4.1.2 Gyroscope

A turntable that can provide oscillations up to 10Hz with a maximum angular rate of 100dps was used to calibrate the three-axis gyroscope. Figure 4.3 shows the placement of the iNEMO on the turntable with the aid of a clamp, so the measurements from each of the three axes could be obtained.

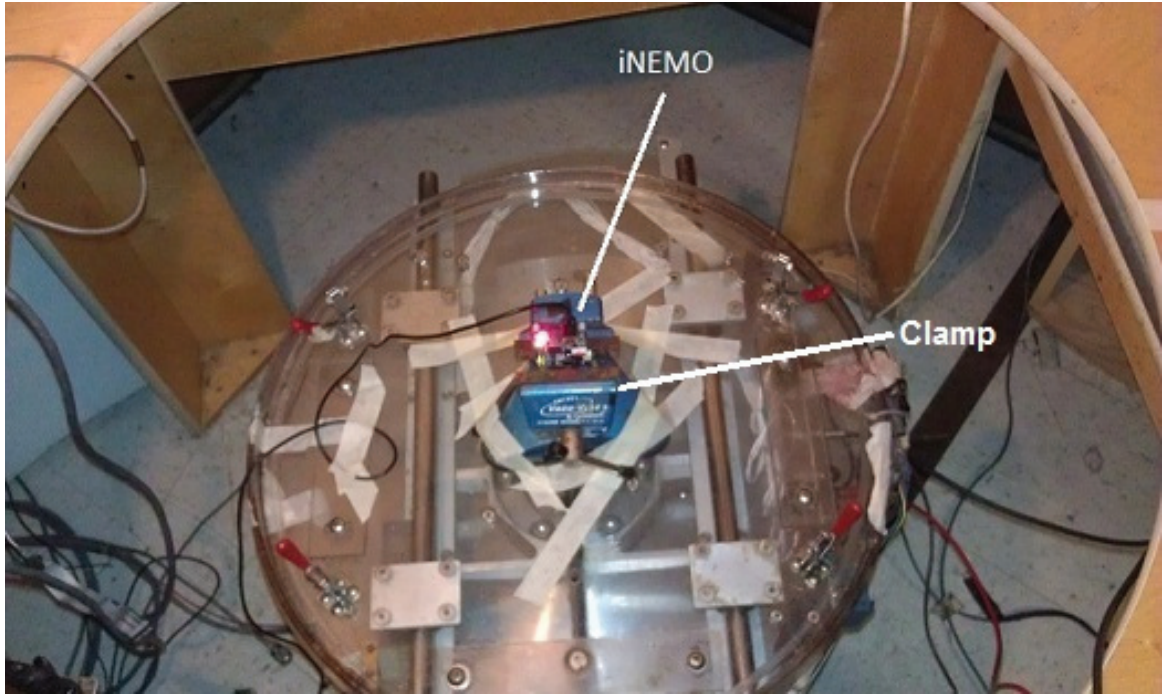


Figure 4.3: Turntable used to calibrate the gyroscope

Similar to section 4.1.1, the relationship between the calibration parameters that need to be determined, measurements, and the known values is as follows.

$$\begin{bmatrix} G_{x1} & G_{y1} & G_{z1} \end{bmatrix} = \begin{bmatrix} G_x & G_y & G_z & 1 \end{bmatrix} \cdot \begin{bmatrix} Gyro_{11} & Gyro_{21} & Gyro_{31} \\ Gyro_{12} & Gyro_{22} & Gyro_{32} \\ Gyro_{13} & Gyro_{23} & Gyro_{33} \\ Gyro_{10} & Gyro_{20} & Gyro_{30} \end{bmatrix} \quad (24)$$

G_{x1} , G_{y1} , and G_{z1} are the known acceleration values while G_x , G_y , and G_z are the raw measurements and $Gyro_{x0}$ are bias values.

Figure 4.4 illustrates the comparison between the iNEMO measurement values about the x-axis and the expected values obtained using a precise optical gyroscope. The turntable was oscillated at 1Hz, 2Hz, and 10Hz with amplitudes of 38dps, 35dps, and 90dps respectively. The experiment was repeated for both the y-axis and the z-axis in

order to generate the required matrices for the least square parameter estimation problem.

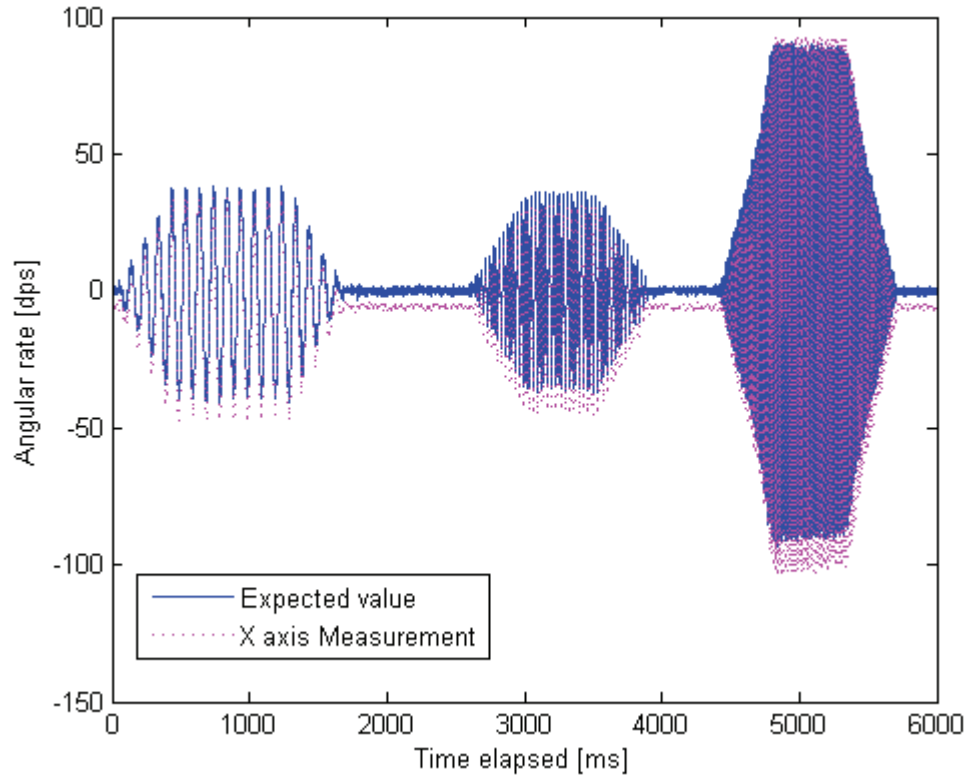


Figure 4.4: Comparison of the expected gyroscope readings and the measurements values about the x-axis

The computed set of gyroscope calibration parameters is as follows:

$$\begin{bmatrix} Gyro_{11} & Gyro_{12} & Gyro_{13} \\ Gyro_{21} & Gyro_{22} & Gyro_{23} \\ Gyro_{31} & Gyro_{32} & Gyro_{33} \end{bmatrix} = \begin{bmatrix} 0.9451 & 0.0197 & 0.0003 \\ -0.0082 & 0.9304 & -0.0097 \\ 0.02 & 0.0427 & 1.0017 \end{bmatrix}$$

$$\begin{bmatrix} Gyro_{10} \\ Gyro_{20} \\ Gyro_{30} \end{bmatrix} = \begin{bmatrix} 5.5561 \\ -12.8228 \\ -0.1553 \end{bmatrix}$$

Similar to the case with the accelerometer, the computed calibration parameters were used in firmware development, and the end user does not have to perform additional calibration prior to using the device.

4.1.3 Magnetometer

The magnetometer on the iNEMO is the least reliable of the sensors since external magnetic fields can easily interfere with the earth's magnetic field. Regardless, gravitational forces alone cannot be used to determine the heading angle, and it is therefore essential to use magnetometer measurements combined with gyroscopic measurements to compute the heading angle. In order to compute calibration parameters for the magnetometer, the iNEMO was rotated about the z-axis, and the heading angle was compared with the angle computed from the gyroscope.

Locating a reliable reference that provided raw magnetometer measurements in Gauss was deemed a challenging task. However, the gyroscope was calibrated in the preceding section, allowing its results to be used as a reference to calibrate the magnetometer. Therefore, the heading angles computed using equations (13) to (16) in section 2.3.4 were compared with the angles computed using the gyroscope. Figure 4.5 illustrates the comparison of heading angles computed using raw and calibrated magnetometer and gyroscope data. The starting position of rotation was determined to be 130° using a handheld compass. The device was turned 300° clockwise, 600° counterclockwise, and 300° clockwise to return to the starting position.

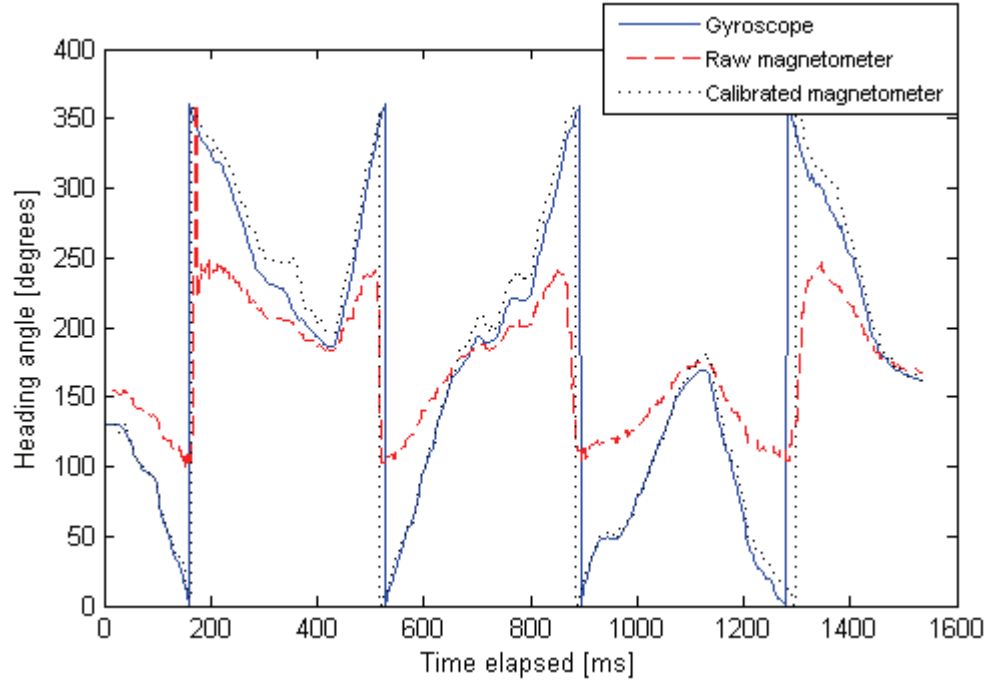


Figure 4.5: Comparison of the yaw angle computation using raw and calibrated magnetometer data and gyroscope data

4.1.4 Sensor Noise Distribution

Once all sensors were calibrated, the *kurtosis* values of the sensor data were compared. Kurtosis is the measure of peakedness of a distribution, and any Gaussian distribution is defined as having a value of 3 [38]. Kurtosis values lower than 3 refer to distributions that have flatter shapes than that of Gaussian distributions. The readout distribution of an ideal sensor is an impulse with a mean at the expected value [33]. Therefore, higher kurtosis values can be associated with more reliable sensors. Table 4.2 shows the kurtosis values of sensors based on 50 000 samples at 100Hz for each sensor axis.

Table 4.2: Kurtosis values of sensor noise

Sensor axis	X	Y	Z
Accelerometer	67.6616	29.1857	8.8683
Gyroscope	1240	5199	146.7
Magnetometer	4.3924	3.117	3.131

Table 4.2 suggests that all sensor readings have kurtosis values that are higher than 3. Very high kurtosis values observed in gyroscope noise data suggest that they are very reliable in the short term. Long term drift issues associated with gyroscopes were not addressed in this experiment. The magnetometer readings have flatter distributions, which confirm the less reliable measurements.

4.2 Drivers - Data Storage

As mentioned before, storing sensor data is of utmost importance for offline analysis. Two storage methods were evaluated: the microcontroller's flash memory and a FAT file system implemented SD card. The performance and the ease of use of each of the two methods were analyzed, and the implementation of a FAT file system was deemed the best data storage option.

4.2.1 Flash Memory

Configuring the iNEMO to store sensor data on the flash memory was a straightforward task using the drivers provided with the Keil IDE. The more challenging task involving the

use of flash memory was that of recovering stored data. Since it was not possible to simply plug the device into a PC to read the flash memory, the iNEMO was configured so that stored data could be sent out using the UART peripheral. The UART peripheral was selected because it allows for the simplest form of communication with a PC. In addition to the configurations of the microcontroller, it was also necessary to develop a program to run on a PC that decoded the data stream received via the computer's USB port. A Python script was developed to implement this task.

Accelerometer, gyroscope, and magnetometer measurements are read as signed 16-bit values (2 bytes). Considering that each sensor has three axes (a total of nine 16-bit values), 18 bytes are required for each sample. The desired sampling frequency is 100Hz; therefore, the time before memory runs out is calculated as follows:

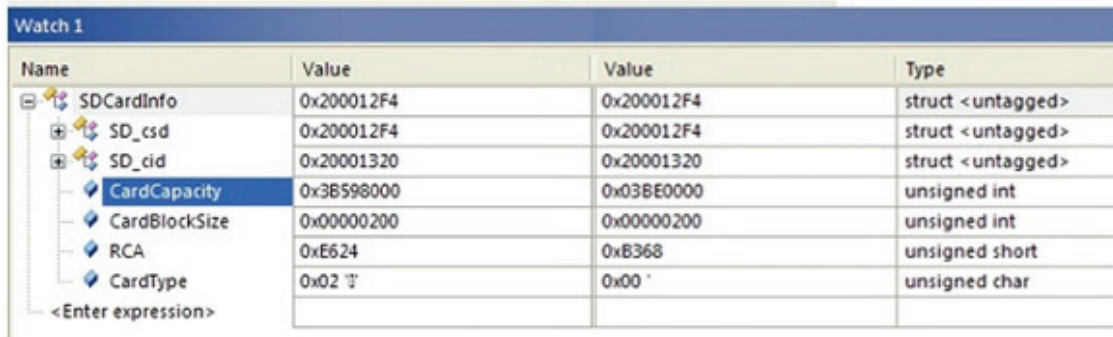
$$t_{max} = 491519 \text{ bytes} * \frac{1 \text{ sample}}{18 \text{ bytes}} * \frac{1 \text{ s}}{100 \text{ samples}} = 273 \text{ s}$$

This is a mere four and a half minutes of data recording, hence it was necessary to explore data compression techniques to optimize the amount of data stored on flash memory.

4.2.2 SD Card

The SDIO interface was configured using the code listings found in Appendix A. The NVIC was configured to handle interrupts from the SDIO interface, and prior to initializing the SDIO peripheral, GPIO pins needed to be mapped and the AHB (AMBA high performance bus) and DMA2 clocks were initialized.

Once initialized, information related to micro-SD cards connected to the interface was read. Figure 4.6 shows the watch window of the Keil IDE when SD cards of different sizes were connected.



Name	Value	Value	Type
SDCardInfo	0x200012F4	0x200012F4	struct <untagged>
SD_csd	0x200012F4	0x200012F4	struct <untagged>
SD_cid	0x20001320	0x20001320	struct <untagged>
CardCapacity	0x3B598000	0x03BE0000	unsigned int
CardBlockSize	0x00000200	0x00000200	unsigned int
RCA	0xE624	0xB368	unsigned short
CardType	0x02 'I'	0x00 '	unsigned char
<Enter expression>			

Figure 4.6: Accessing the micro SD card via the SDIO interface

The first column indicates that the card capacity was 0x3B598000, which is 995 721 216 bytes or 1GB. A second micro-SD card was also tested, and the second column indicates that its size was 0x03BE0000, which is 62 783 488 bytes or 64MB.

As with flash memory storage, retrieving data stored on a raw SD card was challenging since operating systems such as Windows are not equipped with programs to access raw SD cards. As a result, it was necessary to develop code that could save data in a file format that PC users can easily access. The FatFS file system was used to resolve this issue. The functions provided in the FatFS module, summarized in Table 3.1, were used to create data files with extensions such as .dat and .txt on the micro-SD card.

4.3 Drivers - Wireless Connectivity

In order to ensure efficient data transfer while allowing the system to be immune to packet losses, a packet structure with a synchronization word (sync word), the length of payload, a command, the payload, and a checksum was defined. The sync word was chosen to two bytes (16 bits) in order to ensure the receiver can easily recognize the start of the data sequence of the transmitted wireless packet. Longer sync words reduce data corruption; however, they reduce the throughput. This structure can be easily visualized from the following figure.

pkt_tx_buffer	<struct>
sync	" "
[0]	'.' (0x00)
[1]	'.' (0x00)
length	'.' (0x00)
cmd	CMD_STOP_RECORD
payload	<struct>
acc	<struct>
x	0.0
y	0.0
z	0.0
gyro	<struct>
roll	0.0
pitch	0.0
yaw	0.0
checksum	0

Figure 4.7: Packet structure for wireless transmission

It may be possible to define a more sophisticated data packet protocol to handle error detection and correction; however, the current structure is sufficient for the project in discussion. The enum type commands can be seen in Listing 4.1.


```
typedef enum {
    CMD_STOP_RECORD          = 0x0,
    CMD_START_RECORD         = 0x1,
    CMD_STOP_TRANSMIT        = 0x2,
    CMD_START_TRANSMIT       = 0x3,
    CMD_COMPUTE_ORIENTATION   = 0x4,
    CMD_ERROR                = 0xF
} user_command;
```

Listing 4.1: User commands

Moving data from one memory location to another is a processor intensive task; therefore, it is more efficient to use general purpose DMA (direct memory access) controllers on STM32F103 processors [43]. DMA and USART peripherals were configured using the code listings in Appendix A.

An RTOS task was created to initiate the data transfer from the main memory to the USART peripheral. This task waits on a semaphore given by the DMA1_Channel7 interrupt handler once the DMA controller completes transferring the previous set of data.

The use of DMA reduces the number of actions the processor needs to perform, and the DMA controller takes over the data transfer once it is enabled. On the other hand, if the processor were to handle all the data transfers, data needs to be transferred one byte at a time to the USART port.

The STM32W wireless transceiver was connected to the iNEMO module using header J4, which is hard-wired to the USART 2 peripheral. Figure 4.8 illustrates the connections the author had to make between the iNEMO and the wireless module. Although both

devices use STM32 processors, their peripherals are slightly different, so care was taken when porting firmware developed for one device to the other.

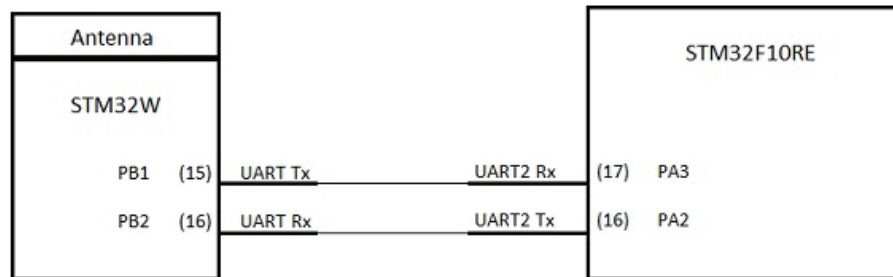


Figure 4.8: Connections between the STM32W (wireless module) and the STM32F10E (iNEMO) micro-controllers

4.4 RTOS-based Task Organization

4.4.1 Task Organization and Real-time Behaviour

With the aid of the RTOS, the main program tasks such as configuring sensors and peripherals, processing sensor data, and wireless connectivity were organized as seen in Figure 4.9.

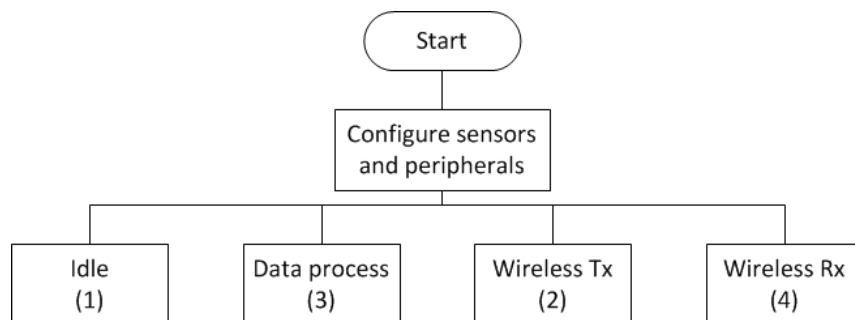


Figure 4.9: Top-level task organization

The priorities of each of these tasks are provided in parentheses. The *wireless Rx* task has the highest priority since it needs to decode any command such as start or stop recording or data streaming needs to be handled immediately. The *wireless Tx* task has lower priority than the *data process* task since reading sensor data should not be interrupted by wireless transmission.

In order to verify that the system can correctly be scheduled, its real-time behavior needs to be analyzed. The system is identified as a mixed *soft and firm real-time system* since tasks that do not execute to completion before their deadlines do not cause system failures. The failure of the *data process* task to complete before its deadline can degrade the quality of the collected measurements. The worst case scenario for the *data process* task is failing to store all sensor data at a particular time stamp, and since data acquisition is performed at 100Hz, the system can afford to miss data points sporadically. Therefore, the task period is set to 9ms. The *wireless transmission* task is invoked at 10Hz, and all memory management is handled by the DMA controller. It was experimentally verified that the processor only requires 8 μ s to initiate the DMA controller, so the deadline for this task is very relaxed. The *wireless receive* task, on the other hand, imposes stricter deadlines. For instance, if the received command is intended to stop the data transmission, any failure to meet the timing deadline renders the command futile. Furthermore, since *wireless receive* is the highest priority task, it is necessary to limit the task period in order to prevent unnecessary wait times incurred by the processor. In order to ensure high priority tasks can use the computing resources

when they need them the most, a pre-emptive scheduling policy is followed in this design [42].

4.4.2 Task Implementation

Each task has its own internal states, and different tasks can be in different states at a given time due to concurrency.

When the iNEMO is powered on, the sensors and the peripherals are configured, and the *idle* task is initiated. The *idle* task consists of four states where the signal *Record* (the compliment of which is *Record*) is used to start or stop data logging as illustrated in the following figure.

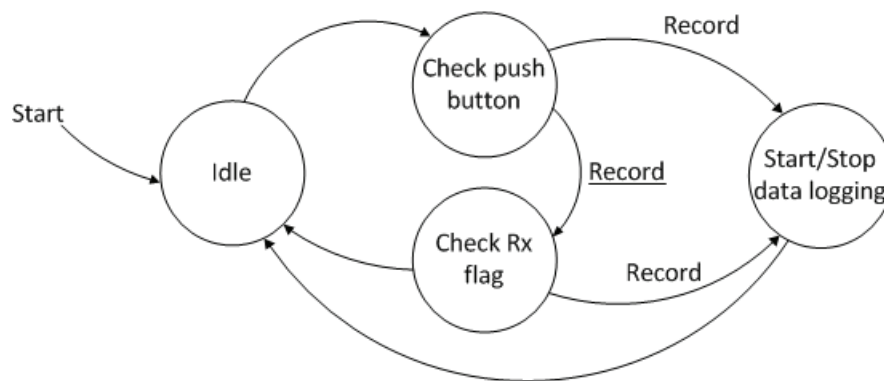


Figure 4.10: The state diagram of the *Idle* task

The *data process* task consists of three states. The signal *CO* (compute orientation) dictates whether the accelerometer and gyroscope measurements are directly stored on the micro-SD card or whether the measurements are used to estimate the orientation

angles of the head, which are later used by application-level software. Figure 4.11 illustrates the state diagram of the *data process* task.

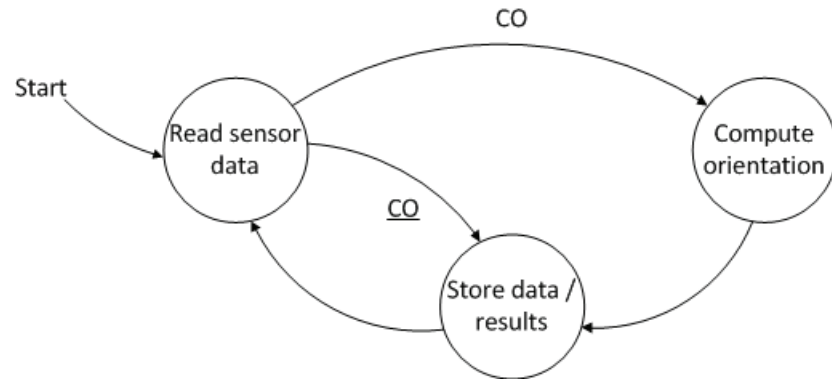


Figure 4.11: The state diagram of the *Data process* task

The wireless transmission task consists of only two states since the DMA controller is configured to handle all memory access tasks. The state diagram is illustrated in Figure 4.12. The signal *Tx* and its complement $\overline{T_x}$ are used to initiate the transfer of either measurement data or computed orientation data depending on the state the *data process* task is in.

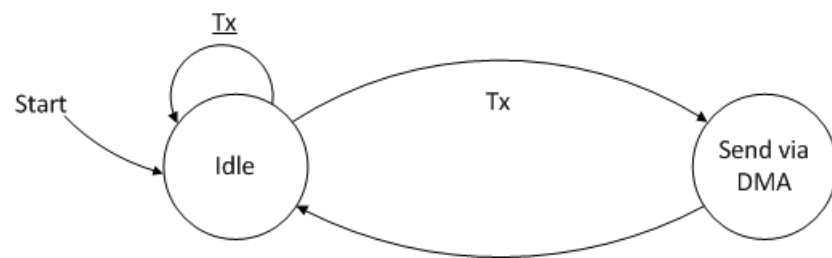


Figure 4.12: The state diagram of the *Wireless transmission* task

The state diagram of the *wireless receive* task is illustrated in Figure 4.13. The signal *RxStart* (the complement is $\overline{RxStart}$) is asserted using a semaphore given by DMA, and the state machine

makes the transition to the *wait* state until the reception is complete or the allocated time runs out. If the reception is complete, the incoming message is processed to reveal the command that it contains.

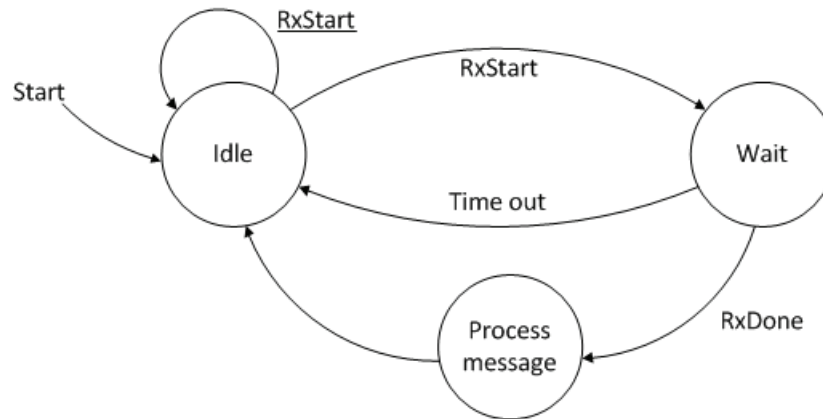


Figure 4.13: The state diagram of the *Wireless receive* task

4.5 Filter Design

The filter theories described in section 2.3.2 were implemented to remove sensor noise as well as to fuse them to get orientation information, which is used in application development presented in chapter 5.

4.5.1 FIR Filter

It is necessary to filter raw sensor data when they are used directly to compare head motion during different activities [36]. Accelerometer and magnetometer signals were

sampled at 100Hz, and the low pass filter was designed based on the technique described in section 2.3.2.

The FIR filter was designed to have a maximum sideband ripple ($-A$) of -50dB, cutoff (f_c) and first stop band (f_r) frequencies of 6Hz and 11Hz. The order (M) and the parameter β were determined to be 29.8 and 4.55 respectively. These parameters were adjusted with the aid of the Matlab filter design toolbox, and the final values were $M=28$ and $\beta = 4.11$. The CMSIS DSP library was used to implement the filter, and Figure 4.14 illustrates input and output signals of the FIR filter as seen on the logic analyzer window of the Keil IDE, and it shows a delay of 140ms in the output signal.

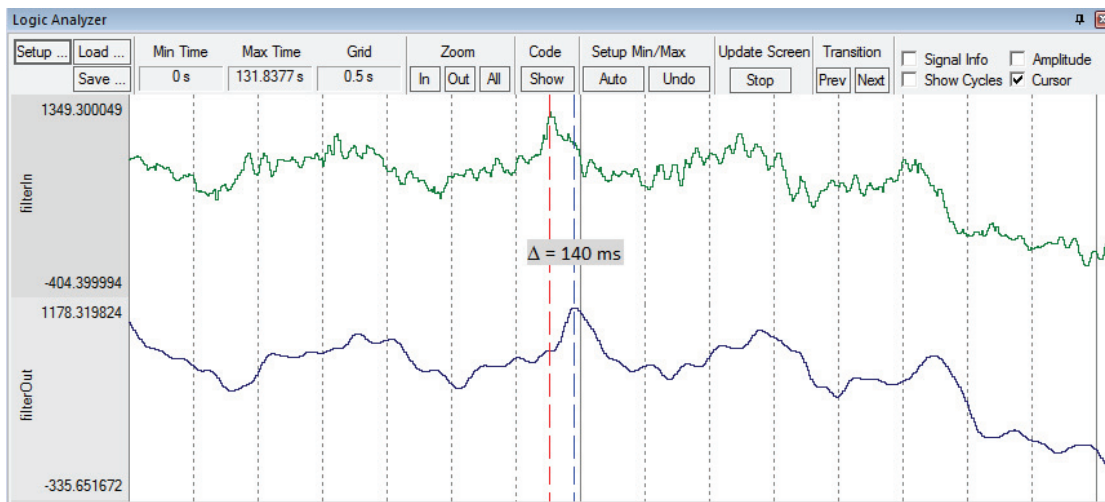


Figure 4.14: Keil logic analyzer signals

4.5.2 Sensor Fusion

In application development, it is necessary to combine the measurements of multiple sensors to obtain useful information. In order to compute the orientation of the iNEMO,

a Kalman filter was designed as described in 2.3.3. The Kalman filter states included orientation angles as well as gyroscope bias values [40], allowing the gyroscope to be periodically recalibrated. The angle about the x-axis was updated as follows:

$$\theta_k^x = \theta_{k-1}^x + (\omega_x - bias_x) * \Delta t \quad (25)$$

Where θ_k^x and ω_x refer to the angle and the angular rate at the k^{th} time step respectively.

The state vector was defined as:

$$x = [\theta_x \quad bias_x \quad \theta_y \quad bias_y \quad \theta_z \quad bias_z]^T \quad (26)$$

The prediction stage of the Kalman filter was implemented using only the gyroscope readings. The fusion of accelerometer and magnetometer readings was implemented in the correction stage. A detailed implementation of the Kalman filter is presented in Appendix B

An additional Kalman filter was used to compute the travelling speeds of speed skaters, and the two sensory data sets fused in that case were the x-axis and the y-axis of the accelerometer. This will be further discussed in section 5.3.3

4.6 Chapter Summary

The implementation details of the design platform presented in Figure 3.1 were presented in this chapter. The sensor calibration scheme improved the accuracy of all

the accelerometer, the gyroscope, and the magnetometer on the iNEMO. The drivers for data storage and wireless transmission were used to make the iNEMO-based head motion monitoring system user-friendly. RTOS-based task management and low-level software development, which includes filter implementations, were used to make the application development easier.

5 Applications

The iNEMO-based head motion monitoring system can be used for various applications. Since one of its features is to compute orientation angles while wirelessly streaming data to a host computer, the head motion can be replicated on a screen in real-time. This capability was exploited to develop a DizzyFIX assistant. In addition, the accelerometer, gyroscope, and magnetometer measurements were used to characterize vestibular signals resulting from various day-to-day activities. The Bayesian classification method was used to classify activities based on inertial data. Furthermore, head motion data was used to develop a training tool for speed skaters. Metrics such as traveling speeds were computed with the aid of Kalman filters.

5.1 DizzyFIX Assistant

DizzyFIX is a treatment provided by Clearwater Clinical Ltd. for vertigo and dizziness. People suffering from dizziness are asked to perform a series of head movements to help treat the main cause of dizziness called Benign Paroxysmal Positional Vertigo (BPPV) [46].

DizzyFIX works by forcing the semicircular canals in the vestibular system to experience a systematic sequence of movements. The iNEMO-based head movement tracking device is designed to replicate the stimuli experienced by the vestibular system, so it can easily be adapted for use with DizzyFIX. Both patients and physicians benefit from seeing

head motion data in real time; therefore, the motion is replicated on a computer screen with the use of the 3D animation engine, Panda 3D, in addition to providing raw data [47]. The Python script used to read input from a serial port and to set roll, pitch, and yaw angles can be found in Appendix C.

5.2 Activity Recognition

The iNEMO-based head motion tracking device was used to collect inertial data from eight healthy individuals engaging in various activities such as biking, driving a car, jumping, walking, running, and sprinting. First, it was necessary to investigate whether head motion data shows distinct characteristics depending on the activities people engage in. This knowledge can be extended to differentiate the head motion of healthy people from that of patients who have a compromised vestibular system.

5.2.1 Classification

Accelerometer readings for different activities are shown in Figure 5.1, and distinct patterns can be observed for each of the activities. These measurements were adjusted according to the tilt angles to correct for the mounting errors. The acceleration resulting from gravity was also removed.

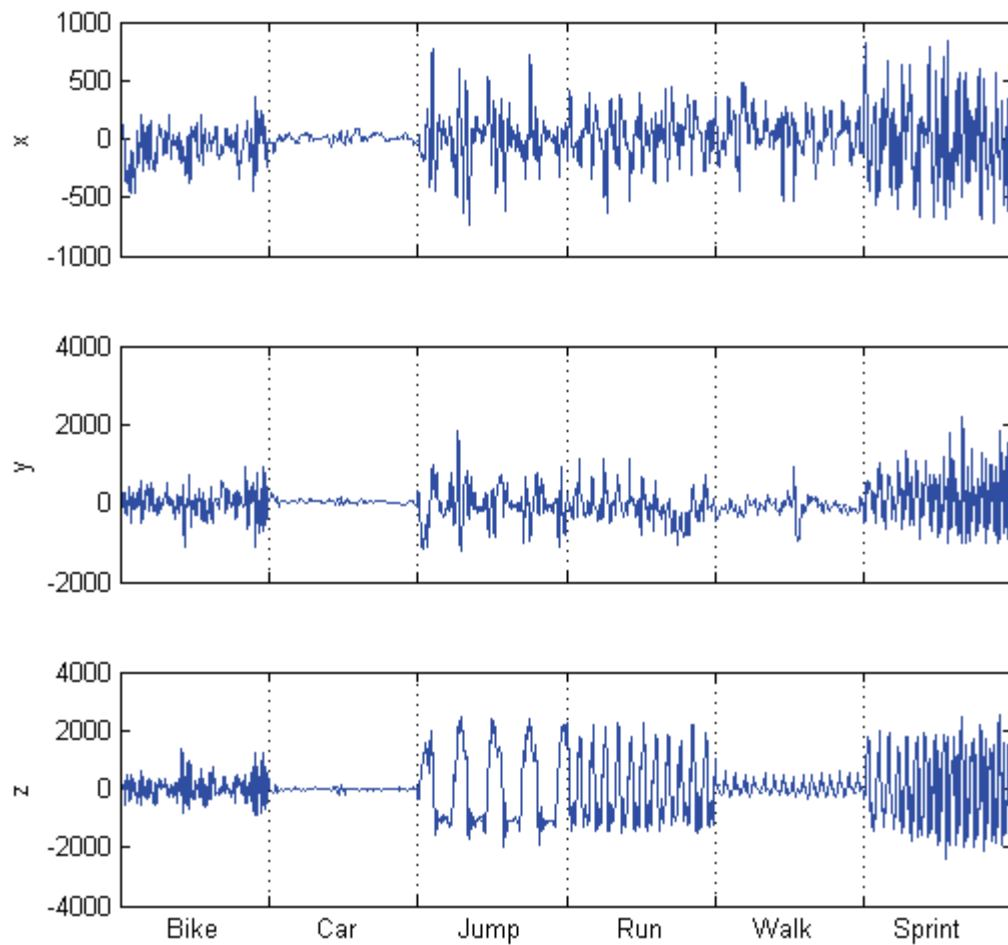


Figure 5.1: Accelerometer readings in mg for different activities

Figure 5.2 illustrates the profile of the gravity-compensated vertical acceleration profile when a subject is engaged in walking and running. Running can be distinguished from walking by the presence of two closely situated acceleration extrema. The first extreme point occurs when the heel strikes the ground and the second extreme point occurs when the foot pushes off the ground. The magnitude of the acceleration spikes and the

time difference between these spikes were characterized to develop an activity recognition scheme.

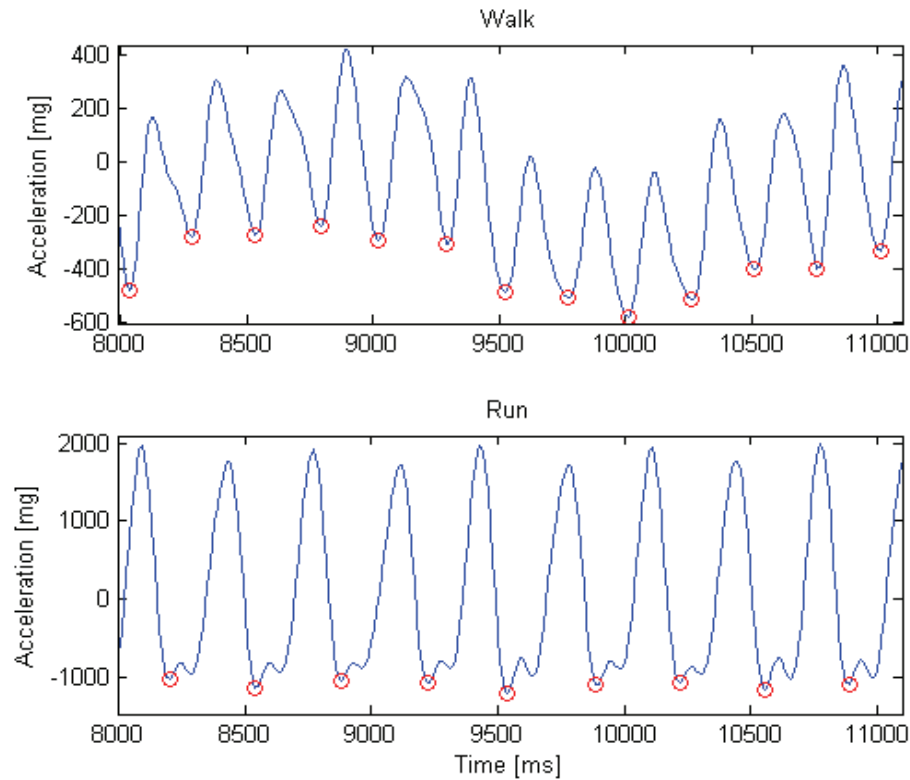


Figure 5.2: Local minima observed in vertical acceleration

Observations such as the ones shown in Figure 5.2 were used to define classes as suggested in section 0. Gaussian parameters such as mean, variance, and covariance of features were computed directly from the available data sets to perform supervised training. In addition, the expectation maximization (EM) algorithm described in section 2.4.2 was implemented in Matlab since one of the objectives was to automate the classification process by means of unsupervised training. The EM algorithm was tested

with training data from the eight individuals, and the results were comparable to what was obtained using supervised training as seen in Table 5.1.

Table 5.1: Statistical characteristics of supervised and unsupervised training

Activity		Walk		Run	
		Supervised	Unsupervised	Supervised	Unsupervised
Acceleration peak	Mean (mg)	369.3	368.36	1043.1	1043.7
	Variance (mg ²)	10460	9632.8	12431	12205
Period	Mean (ms)	263.58	263.59	356.3	356.3
	Variance (ms ²)	931	917.5	1040	1057
Covariance (mg ms)		-521	-527.1	-213	-251

The collected data samples are illustrated in Figure 5.3, where Gaussian contours corresponding to means, standard deviations, and covariances of walking and running are shown.

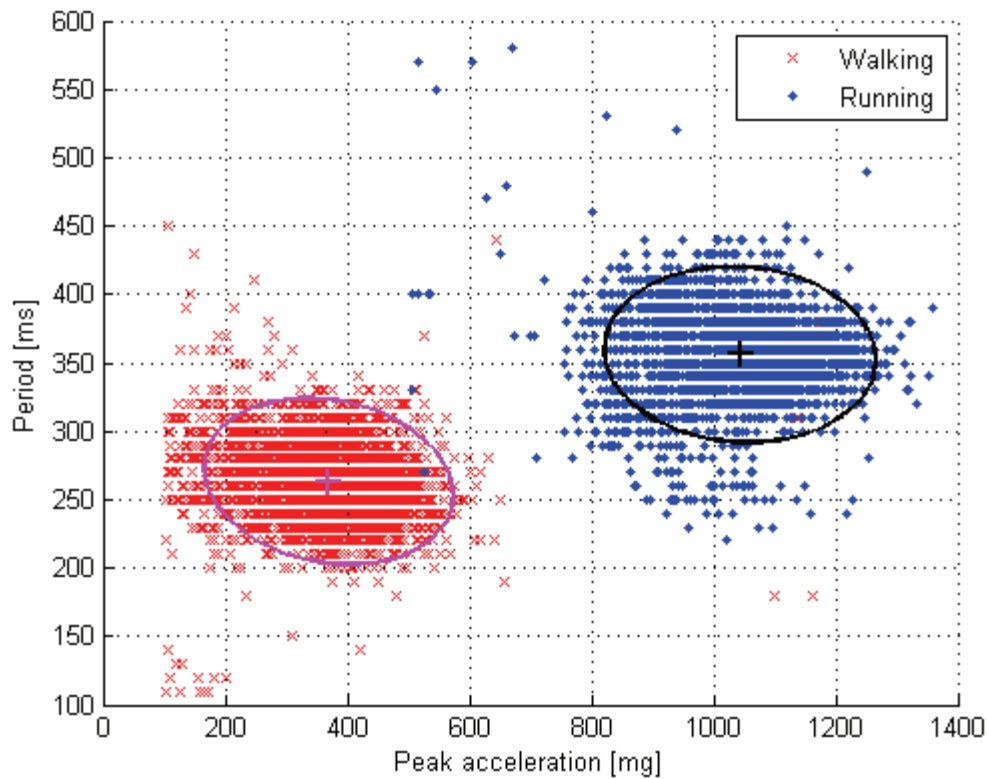


Figure 5.3: Distribution of peak acceleration and period

5.2.2 Identification

The following is a sample set of evidence in a two-second interval taken from an individual whose data was not used as a training set.

Table 5.2: Sample data set to be classified

Acceleration peak[mg]	626.5329	537.332	661.9986	605.5914	547.5217	650.6915
Period [ms]	470	400	480	570	550	430

The mean values are 604.95mg and 483.3ms. The probability densities were computed using equations (17) to (20), and the naïve Bayesian classifier was set up as follows:

$$f_b(E) = \frac{p(E|run)}{p(E|walk)} = \frac{2.0039e - 11}{5.0956e - 19} = 3.9327e7 > 1$$

The data sample was then classified as belonging to a running event. The experiment was repeated with 30 data sets belonging to running and walking events to give statistical significance to the test. Only one of the samples was incorrectly classified, resulting in a false acceptance rate of 0.03.

5.2.3 Patient Monitoring

One of the challenges associated with testing the iNEMO-based head motion monitor was locating patients who had undergone vestibular surgery. Data collected from healthy individuals presented in section 5.2.1 form a solid base, upon which further improvements to the head motion monitoring system can be built. The differences between patterns of healthy individuals and patients engaging in the same activity are not expected to be as distinct as the case with the comparison of running and walking. Therefore, it is necessary to explore classification algorithms beyond naïve Bayes, and such a task will be an excellent future extension to this thesis.

5.3 Speed Skater Tracking

Another application of the iNEMO-based head motion monitoring system is the study of athletes. Knowing which head motions are associated with the best performance in speed skating can help athletes and coaches improve their performance. In this case, the main objective is to assist competitive sporting persons to improve their performance by understanding where the physical movement was not optimized for the best performance. The differences between training for short and long track competitive events also need to be established and taken into account in order to achieve usability for all speed skaters. Testing of the device was conducted in partnership with the Canadian national women's speed skating team. Data samples were collected from six athletes training for the 500m short track event. The dimensions of a speed skating track, which is the size of an ice hockey rink, are presented in the following figure.

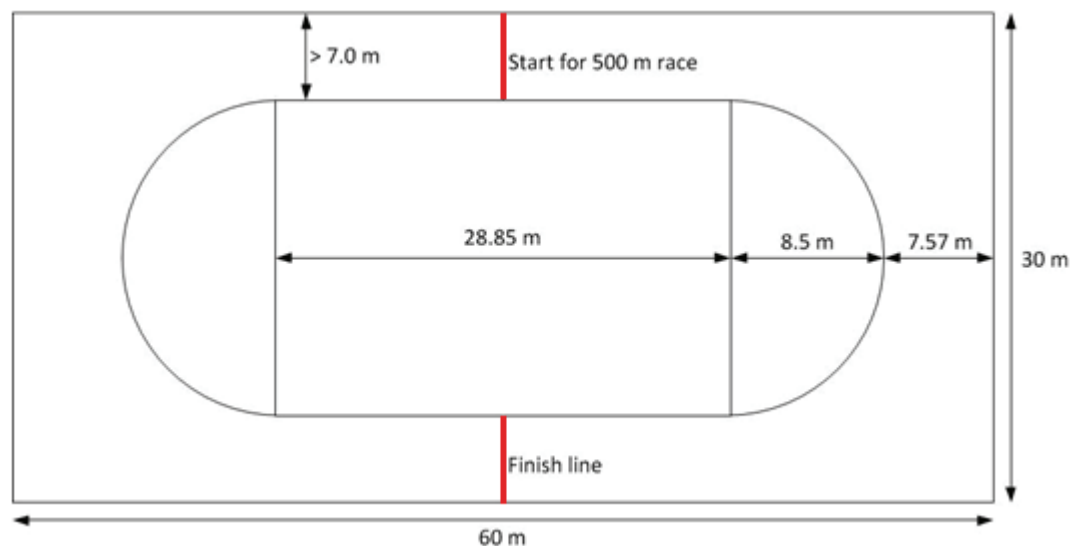


Figure 5.4: Speed skating track (not to scale) [53]

5.3.1 Data Acquisition

The plastic box fabricated to aid sensor calibration was also used to mount the iNEMO-based head motion monitor on the helmets of athletes. A plastic flap was attached to the underside of the case to allow the device to be attached to a helmet using masking tape, and Figure 5.5 shows speed skaters wearing the device. The skater motion was captured on video, against which the results obtained using the iNEMO were compared.



Figure 5.5: Speed skaters wearing the iNEMO

Efforts were put in to make sure that the device axes were aligned with global x , y , z axes. However, the analysis was not affected by the initial position because the measurements were easily rotated to the global coordinate system using the rotation matrices described in section 2.3.4.

Figure 5.6 shows the measurements of one skater after completing four laps, and the results shown were transformed using rotation matrices and filtered using the filter designed in section 4.5.1. In addition, the measurements were corrected for gravity based on the initial head position.

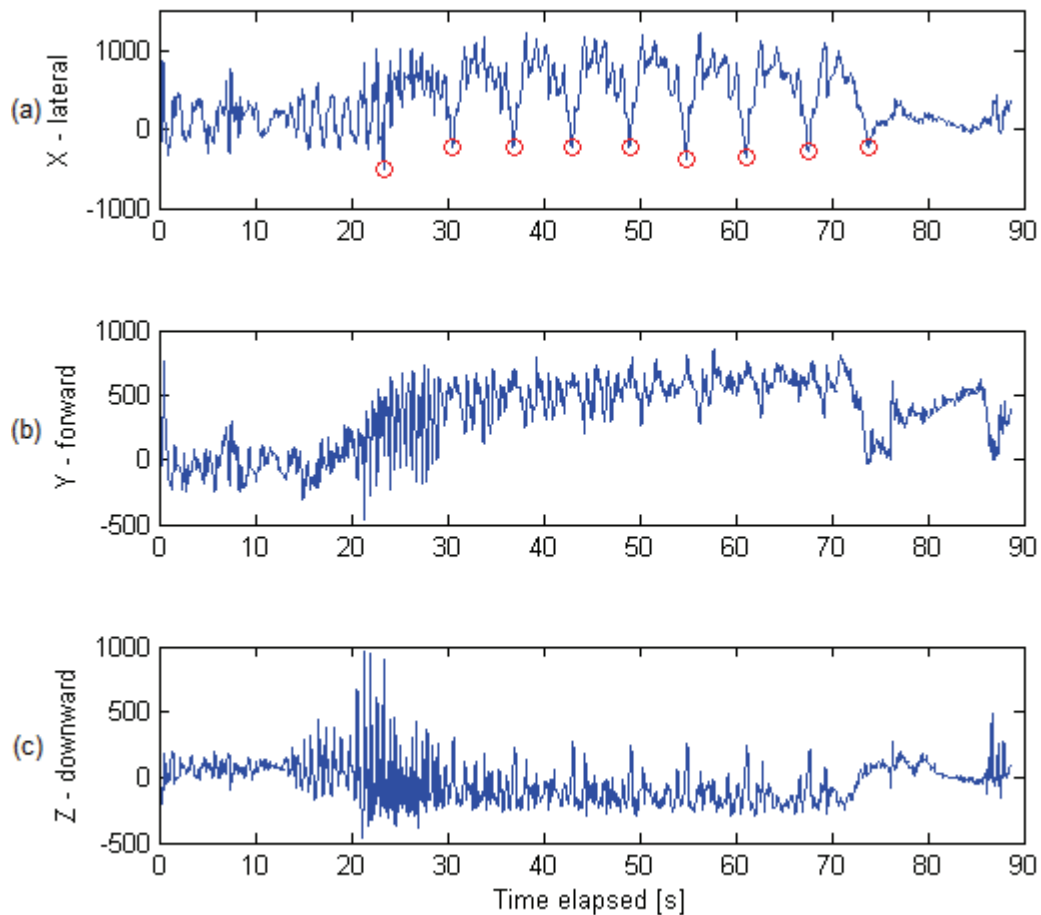


Figure 5.6: Accelerometer readings of four laps in mg

Figure 5.6a suggests that the lateral head movement (x-axis) is very periodic, and the local minima are identified as the events when skaters transfer their weight to the

outside of the track and straighten their heads when they are on the lines marked with red in Figure 5.4. This is also the position on the track that results in the lowest centripetal force acting on the skaters. It can also be concluded from y-axis data that the skaters tilt their heads forward while engaging in fast skating.

Figure 5.7 illustrates the angular rates of motion in each of the pitch, roll, and yaw axes as defined in section 1.1.1.

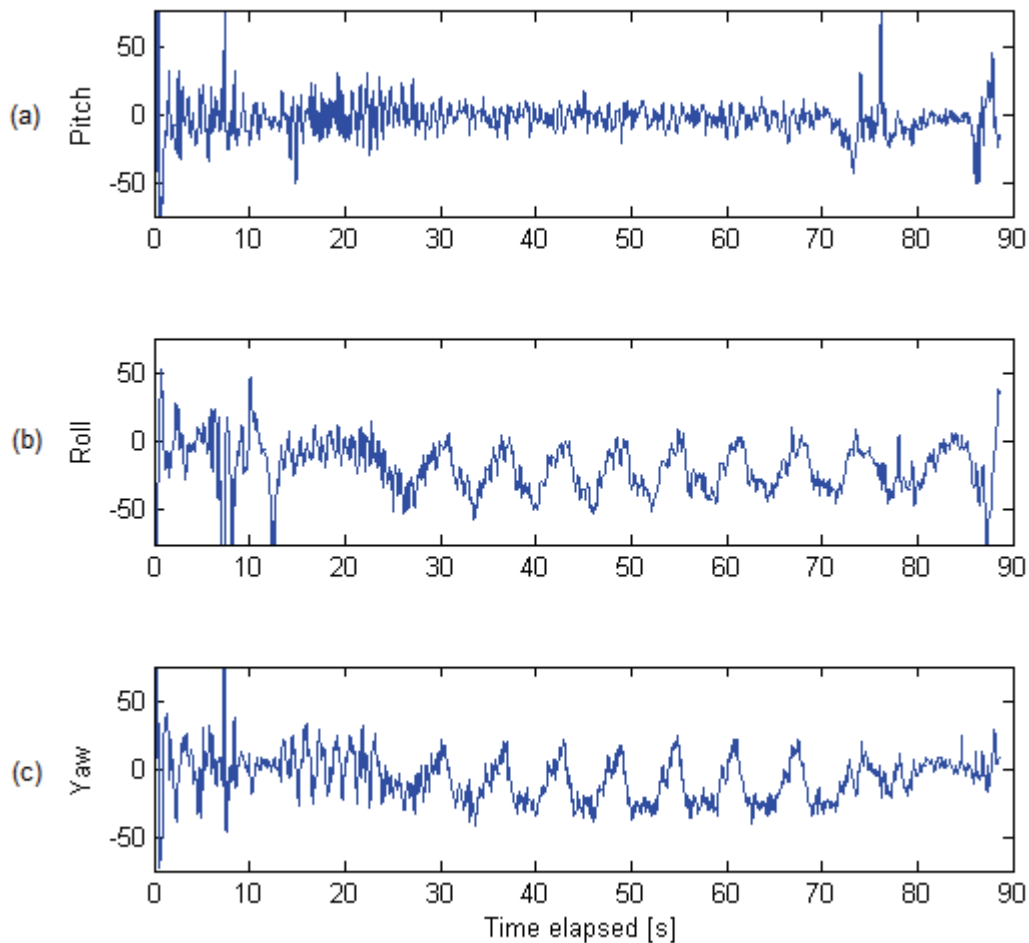


Figure 5.7: Gyroscope readings of four laps

It is interesting to note that there is a significant amount of periodic roll and yaw rotations when the athletes are skating around the track. Although the angular rates are high, they only correspond to very small amplitudes of actual head movement, and a camera based motion tracking system is likely to miss these events.

5.3.2 Head Orientation Estimation

The complimentary and Kalman filters designed in section 4.5.2 were used to estimate the orientation of each of the athletes' heads. The orientation estimate was then used to project the acceleration vector recorded by the accelerometer to the forward direction in order to be used for speed estimation. The pitch and roll angle computations provided satisfactory results as seen in Figure 5.8.

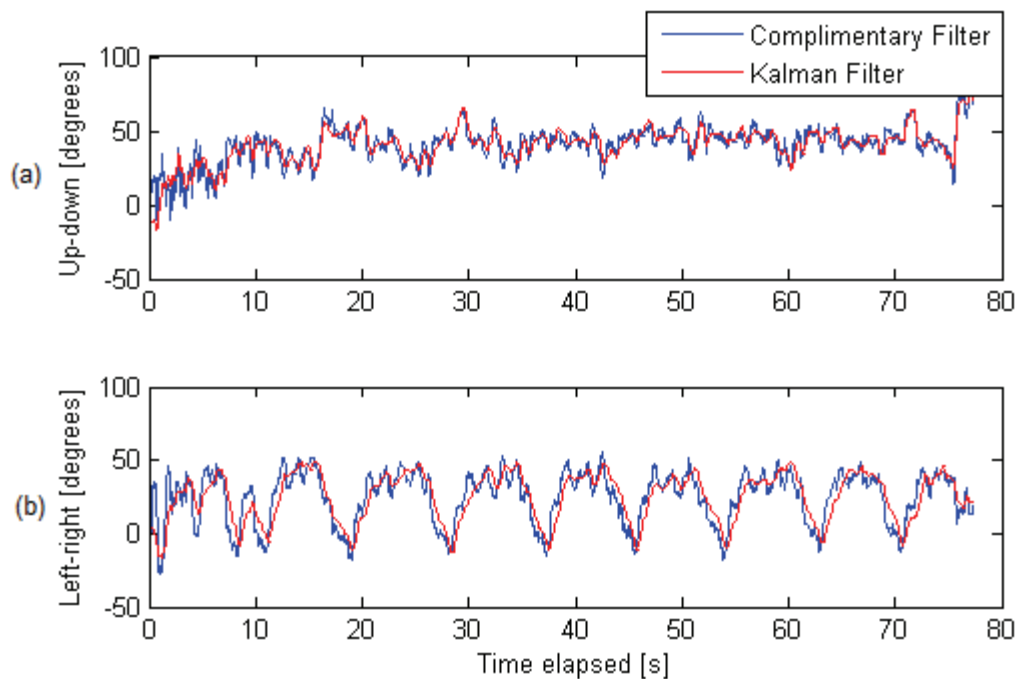


Figure 5.8: Head orientation estimation

Figure 5.9 shows a magnified version of the head orientation estimation, from which it is evident that the Kalman-based orientation estimation yields smoother results than that of the complimentary filter.

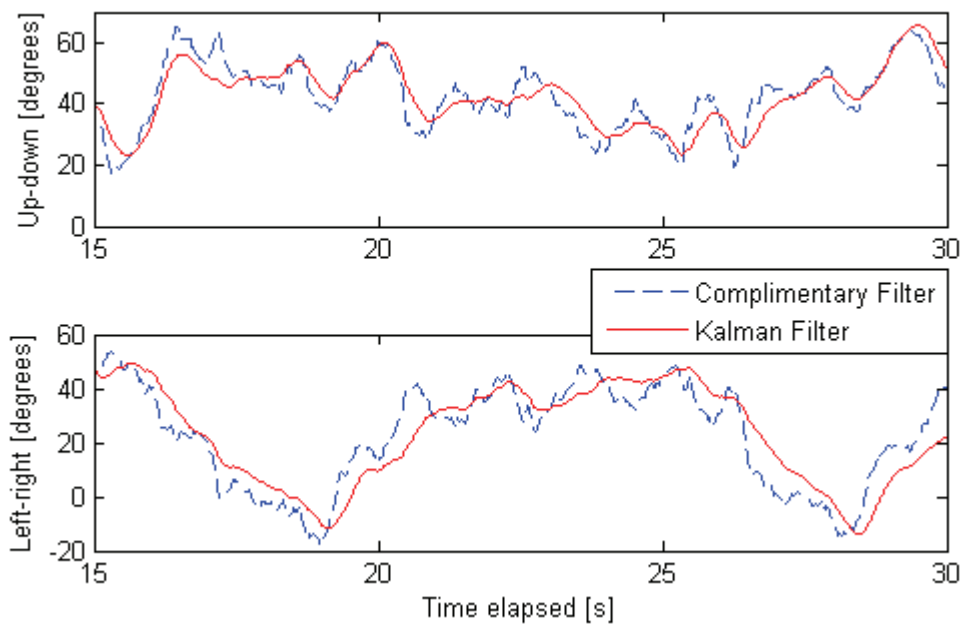


Figure 5.9: Head orientation estimation (magnified)

The up-down angles (pitch) plot in Figure 5.8a shows that the skaters start the run with their heads straight, but once they gain the cruising speed, they tend to lean forward and tilt their heads down. The left-right (roll) angles plot shows remarkable periodicity similar to what is seen in Figure 5.6. The figure suggests that, as is the case in reality, the skater leans towards the centre of the curves when traveling around them. The local

minima points again refer to events where the skaters push forward while straightening their heads as they pass the lines marked with red in Figure 5.4.

5.3.3 Speed Estimation

Estimating translational speeds was a significantly more challenging task. In general, IMU speed predictions, obtained using integration methods, are fused with the speeds obtained from GPS data in applications such as automobiles. However, GPS data is not available for indoor use, which is the case with speed skaters. The pedometer approach can be used to estimate running speed, where the time difference between acceleration minima observed in the vertical direction, as seen in Figure 5.2, along with average stride lengths are used to compute the speed [4]. However, such periodic events cannot be observed during speed skating since the very nature of the sport dictates that athletes slide forward.

Furthermore, simply integrating the forward component of the acceleration vector is prone to multiple sources of error. The orientation of the IMU needs to be determined with extreme precision to ensure the accuracy of the acceleration. Even a small error such as $25mg$ in acceleration can result in an error more than $3.6m/s$ within $15s$. Given that the average speed of speed skating is less than $10m/s$, the error is prohibitively large.

However, the average speed per half lap could be computed using the time difference between the acceleration minima shown in Figure 5.6 along with the knowledge of the length of a speed skating track, which is 111.1m [53]. Figure 5.10 shows the comparison of the speed of three different skaters as they complete four laps on the speed skating track.

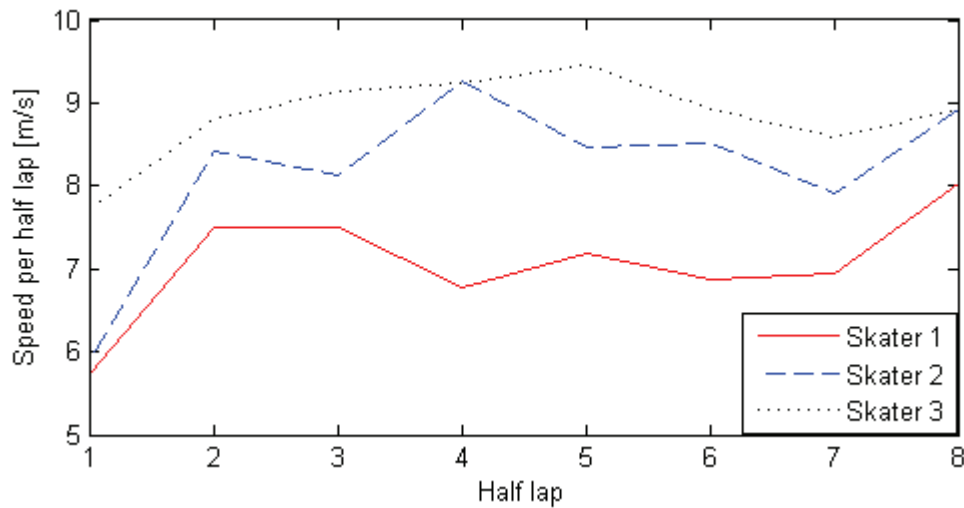


Figure 5.10: Speed in each half lap

The skaters-in-action were captured on video in order to validate the speeds computed using the iNEMO. Table 5.3 summarizes the speeds and the errors of skaters 4 and 5. The percent errors in speed estimation are illustrated in Figure 5.11 for easier visualization. It is evident that the maximum error is less than 12% and that the mean of the absolute error in all cases is 4%.

Table 5.3: Comparison of speeds (in m/s)

Half lap	Skater 4				Skater 5			
	Video	iNEMO	Error	%Error	Video	iNEMO	Error	%Error
1	6.194	5.955	-0.239	-3.860	5.917	6.605	0.688	11.622

2	6.187	5.922	-0.265	-4.284	6.568	7.149	0.581	8.850
3	6.126	6.078	-0.048	-0.786	6.599	6.653	0.053	0.809
4	6.553	6.799	0.247	3.766	6.999	6.850	-0.149	-2.131
5	6.560	6.551	-0.010	-0.146	6.793	7.032	0.238	3.510
6	6.446	6.207	-0.239	-3.712	6.912	7.131	0.219	3.173
7	6.687	7.348	0.661	9.888	6.727	6.944	0.216	3.217
8	6.491	6.417	-0.075	-1.148	6.655	6.392	-0.262	-3.941

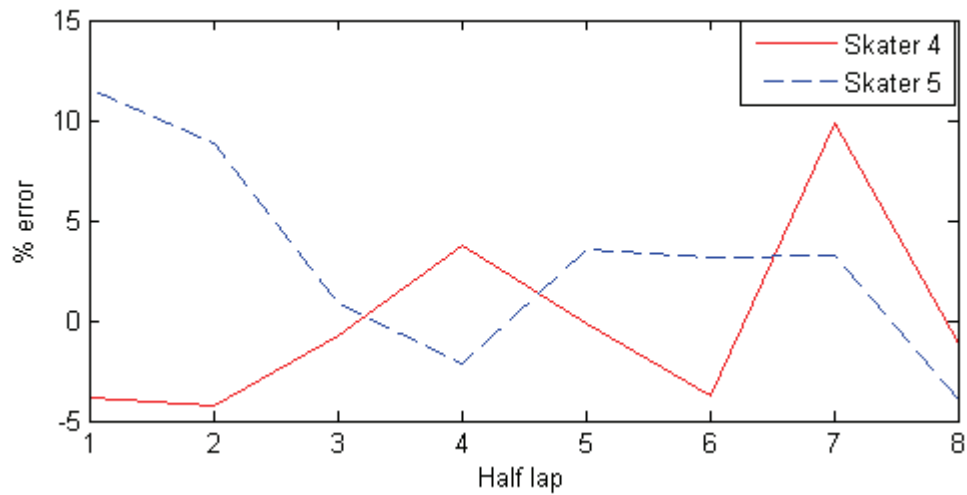


Figure 5.11: Percent error in speed estimation

The Kalman filter algorithm described in 2.3.3 can be used to fuse the average speed shown in Figure 5.10 and the instantaneous speeds obtained by integrating the acceleration. The resulting speed is shown in Figure 5.12.

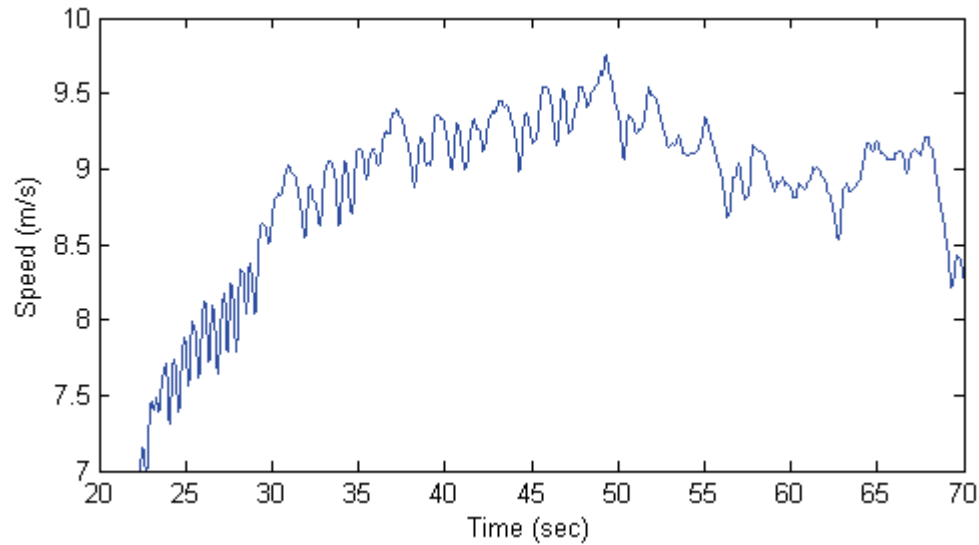


Figure 5.12: Speed as a function of time – improved using the Kalman filter

Kalman filtering allows the contribution from less reliable inputs to be minimized, and in this case, the less reliable input was the speed computed by integrating the acceleration. Since the speed estimated using the periodic behavior is more reliable, more weight was given to it.

One of the challenges associated with this technique is that the average speed can only be estimated once skaters travel at least half a lap, which is 55.55m. Also, the speed obtained via integrating the forward component of acceleration accumulates very quickly. Considering how skaters take approximately 8-9 seconds to travel half a lap, the rate at which this data is available is less than 0.2Hz. Therefore, the real-time operation of this technique needs further improvement.

5.3.4 Performance Comparison

Once the traveling speeds are known, the athletes find it useful to compare the head motion associated with slow and fast laps. Knowing which head motions are associated with best performance can help athletes to improve their techniques. Figure 5.13 illustrates a comparison of the lateral acceleration the head experiences during half-lap 5 and 7 in the case of skater 3. The respective average speeds are 9.44m/s and 8.58m/s. The magnitude of acceleration in half-lap 7 is slightly smaller than that of half-lap 5, and this observation is the result of skaters tilting their heads less in the latter. Less head-tilting occurs when skaters travel slower during the curved segment of the track. Instantaneous speeds during each half lap are also readily compared using this tool.

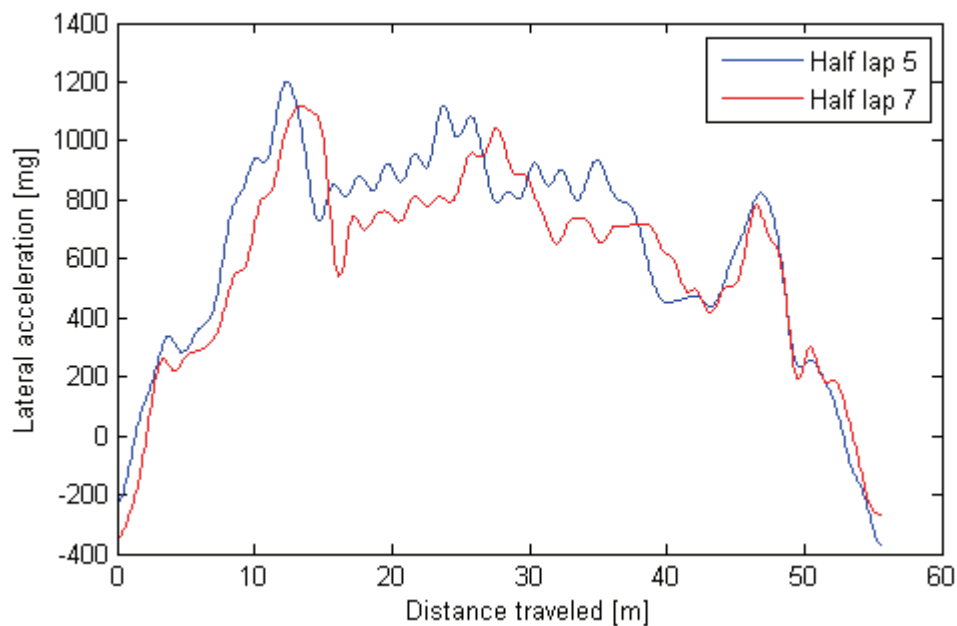


Figure 5.13: Comparing the head motion in two half laps

5.3.5 Graphical User Interface

A graphical user interface (GUI) was developed to make the system user-friendly. The results presented in sections 5.3.1 to 5.3.3 are made available to skaters and their coaches the moment they complete their runs, so they can examine the profile of head motion to identify the causes for slowing down on a particular lap. Figure 5.14 shows the acceleration at each position on the track using the GUI.

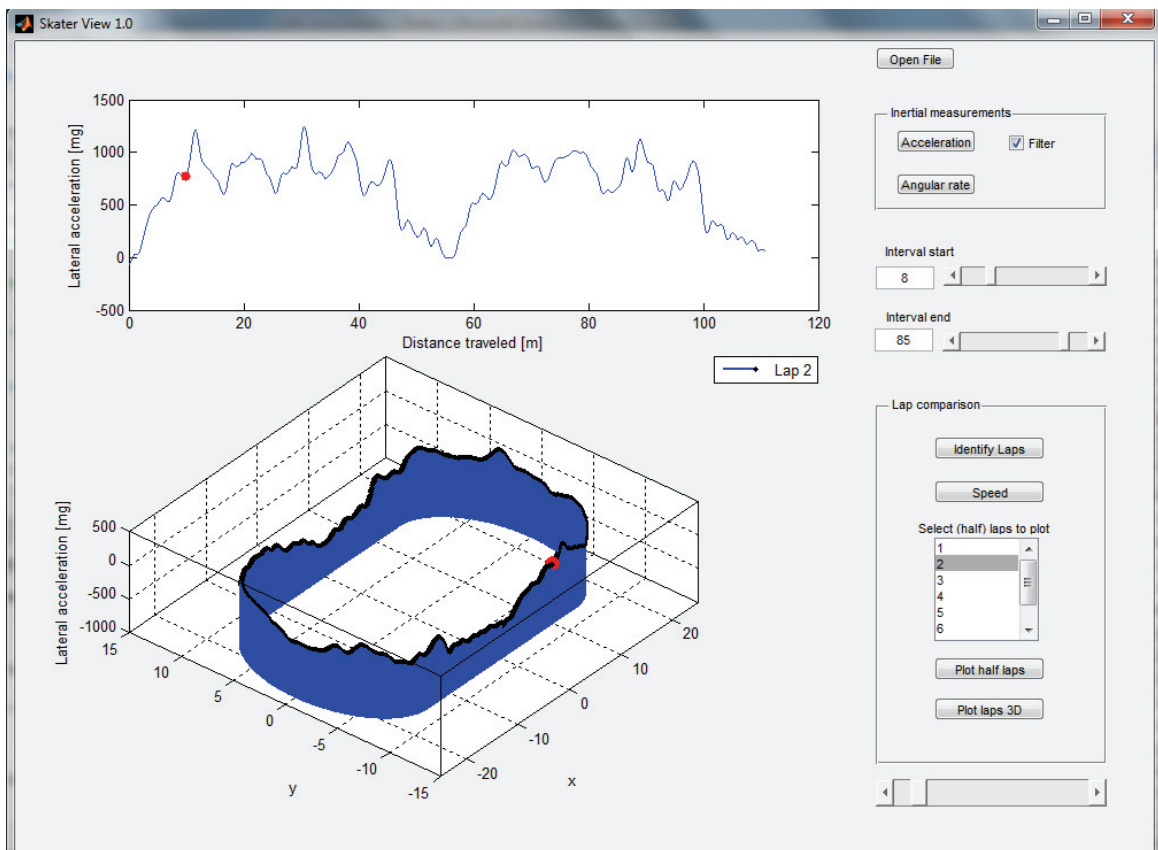


Figure 5.14: GUI showing the acceleration as a function of position on the skating track

The GUI usage is very simple. Once the data file is loaded, accelerometer and gyroscope measurements can be plotted. The sliders labeled “Interval start” and “Interval stop” can be used to adjust the start and the end points of lap data. Once the laps are identified, the speeds can be plotted, and the list box lists the available half laps that can be superimposed to compare head motion. The buttons “Plot half laps” and “Plot laps 3D” can be used to plot the acceleration data of multiple laps in 2D and 3D, and the slider at the bottom right corner can be used to move the skater’s position, indicated by a red dot in Figure 5.14, around the track.

6 Conclusions and Future Work

6.1 Conclusion

A human head motion monitoring system based on the iNEMO inertial measurement unit and the STM32W wireless module was implemented in this thesis. Firmware implementation details on sensor integration, data storage, and wireless transmission were presented. A sensor calibration scheme to ensure sensor reliability, which is essential in medical applications, was presented.

The iNEMO-based head motion monitoring system was used to develop an activity recognition scheme that can help physicians characterize head motion of their patients when they engage in day-to-day activities both prior to and following vestibular surgery. In doing so, the head-motion characteristics of several day-to-day activities such as running, walking, jumping, and biking were studied. The activity classification scheme that was developed was based on the Bayesian classification model.

A graphics user interface (GUI) was developed to provide Speed Skaters and their coaches with head motion data after their training is complete. Kalman filter based sensor fusion techniques were used to estimate their traveling speeds. Also, design challenges associated with monitoring such sports were identified.

6.2 Future work

In future, a custom IMU will be designed to have multiple accelerometers and gyroscopes to obtain a fault-tolerant design. Multiple IMUs placed on the core body can supplement the readouts from the head worn IMU for closer tracking of the limb movement, which can contribute to further athletic performance improvements. Furthermore, it is beneficial to have the wireless radio on the same PCB to make the overall device smaller.

The activity recognition scheme can be improved by analyzing head motion data of patients with compromised vestibular systems and using more sophisticated classifiers than naïve Bayes. Although the objective of this thesis was to use human head motion measurements for activity recognition, additional sensor modules placed on the core body and on legs can be used to improve the pattern recognition process. Ultra-sound transmitters and receivers can be used to estimate the position of speed skaters, and these results can be fused with IMU results to improve speed skater tracking.

This platform is also used to be used to collect data associated with athletes colliding and falling. Collecting such data in an un-simulated setting is an extremely difficult task. However, the availability of such data can be used to develop a system that can notify athletes the levels of trauma and concussions they experience when they collide with other athletes or the fence of the skating rink.

Furthermore, this platform will be expanded to multiprocessor implementations of embedded systems [54], [55]. Special attention will be paid to arithmetic optimization

[56], [57], [58] and verification [59], including the spectral methods for incompletely specified functions [60]. Moreover, the debug effort [61], dependable implementation [62], [63], and designs containing efficient serial interfaces for improve communication [64] will also be objectives of future implementations. Improvements to sensor fusion, with the help of techniques such as the ones presented in [65], will be a major extension of this work in order to enhance activity classification.

Appendix A

1. Data Storage

The most important configurations used to communicate with the SDIO interface:

```

NVIC_InitStructure.NVIC_IRQChannel = SDIO_IRQn;
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 11;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
NVIC_Init(&NVIC_InitStructure);

SDIO_InStruct.SDIO_ClockDiv = SDIO_TRANSFER_CLK_DIV;
SDIO_InStruct.SDIO_ClockEdge = SDIO_ClockEdge_Rising;
SDIO_InStruct.SDIO_ClockBypass = SDIO_ClockBypass_Disable;
SDIO_InStruct.SDIO_ClockPowerSave = SDIO_ClockPowerSave_Disable;
SDIO_InStruct.SDIO_BusWide = SDIO_BusWide_1b;
SDIO_InStruct.SDIO_HardwareFlowControl =
                                SDIO_HardwareFlowControl_Disable;
SDIO_Init(&SDIO_InStruct);

```

Listing 1: SDIO interface configuration [24]

Function calls used to perform read and write operations on the SD card:

```

// Single block read/write

SD_WriteBlock(u32Address, s_Buffer_Block_Tx, BlockSize);
SD_ReadBlock(u32Address, s_Buffer_Block_Rx, BlockSize);

// Multiple block read/write

SD_WriteMultiBlocks(u32Address, s_Buffer_MultiBlock_Tx,
                    BlockSize, NumberOfBlocks);
SD_ReadMultiBlocks(u32Address, s_Buffer_MultiBlock_Rx,
                  BlockSize, NumberOfBlocks);

```

Listing 2: Read and write operations [24]

The preceding function definitions are included with Keil IDE installations.

2. Wireless Interface

Key statements used to configure the DMA controller to communicate with the USART peripheral to transmit data:

```
RCC_APB1PeriphClockCmd( RCC_APB1Periph_USART2, ENABLE );

DMA_InitStructure.DMA_PeripheralBaseAddr = (u32)&USART2->DR;
DMA_InitStructure.DMA_MemoryBaseAddr = ((u32)&pkt_tx_buffer);

DMA_InitStructure.DMA_Priority = DMA_Priority_High;

DMA_DeInit(DMA1_Channel7);
DMA_Init(DMA1_Channel7, &DMA_InitStructure);
```

Listing 3: DMA configuration for wireless transmission

`pkt_tx_buffer` is an instance of the packet structure described in Figure 4.7, and it contains data that needs to be transmitted.

Wireless transmission task using DMA:

```
pkt_tx_buffer.sync[0] = 0x55;    // Synchronization bytes
pkt_tx_buffer.sync[1] = 0x0E;
DMA1_Channel7->CMAR = (uint32_t)& pkt_tx_buffer;

DMA_Cmd(DMA1_Channel7, ENABLE);
```

Listing 4: Wireless transmission task

Appendix B

1 Kalman Filter

This appendix summarizes the state and noise definitions used for orientation estimation using the Kalman filter. These definitions were modified when the travelling speed of speed skaters were computed using the Kalman filter. The state transition and the input matrices, based on eq. 26 (pg. 57), are as follows.

$$A = \begin{bmatrix} 1 & -\Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -\Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -\Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} \Delta t & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Delta t \\ 0 & 0 & 0 \end{bmatrix}$$

The output matrix:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The covariance matrix of process noise is defined using the noise variance observed for the accelerometer and the gyroscope during calibration. The process noise model presented below emphasizes the noise only on individual axes because the

accelerometer and gyroscope calibration scheme introduced in section 4.1.1 and 4.1.2 minimizes cross axes effects.

$$Q = \begin{bmatrix} aX & gX * \Delta t^2 & 0 & 0 & 0 & 0 \\ 0 & gX & 0 & 0 & 0 & 0 \\ 0 & 0 & aY & gY * \Delta t^2 & 0 & 0 \\ 0 & 0 & 0 & gY & 0 & 0 \\ 0 & 0 & 0 & 0 & aZ & gZ * \Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & gZ \end{bmatrix}$$

aX , aY , and aZ refer to noise variance of each of the three axes of the accelerometer, and gX , gY , and gZ refer to noise variance of each of the three axes of the gyroscope.

The covariance matrix of measurement noise is constructed using the variance observed for the angles computed using the accelerometer and magnetometer during calibration.

$$R = \begin{bmatrix} accAngleNoise & 0 & 0 \\ 0 & accAngleNoise & 1 \\ 0 & 0 & magAngleNoise \end{bmatrix}$$

Appendix C

1. Python script to interface with Panda3D

The following script was developed based on the example code provided with the Panda3D engine [47]. The script reads serial data and unpacks them into roll, pitch, and yaw values.

```
from struct import unpack_from
import sys

#serial port
import serial
import io

from direct.showbase.ShowBase import ShowBase
from direct.task import Task
from direct.actor.Actor import Actor
from direct.interval.IntervalGlobal import Sequence
from panda3d.core import Point3

class RotateApp(ShowBase):
    def __init__(self):
        ShowBase.__init__(self)

        # Load the environment model.
        self.enviro = self.loader.loadModel("models/environment")
        # Reparent the model to render.
        self.enviro.reparentTo(self.render)
        # Apply scale and position transforms on the model.
        self.enviro.setScale(0.25, 0.25, 0.25)
        self.enviro.setPos(-8, 42, 0)

        # Add CameraTask and Spin procedures to the task manager.
        self.taskMgr.add(self.CameraTask, "CameraTask")
        self.taskMgr.add(self.DizzyFix, "DizzyFix")

        # load and transform the panda actor
        self.spinActor = Actor("models/DizzyFixActor", {"DizzyFix":
"models/DizzyFixActor1"})

        self.spinActor.setScale(0.3, 0.3, 0.3)
        self.spinActor.reparentTo(self.render)
```

```

# Define a procedure to move the camera.
def CameraTask(self, task):
    self.camera.setPos(0, -20, 3)
    return Task.cont

# Define a procedure to read serial data and set roll, pitch, yaw
def DizzyFixTask(self, task):
    s = serial.Serial('com10', 9600, timeout=0.03)
    buff = s.read(20000000)

    rf_packets = []
    for i in range(len(buff)/4):
        rf_packets.append(unpack_from("f", buff, i*4)[0])

    # each rf_packet is 16 bytes:
    # first 4 bytes -- preamble and commands
    # next 12 bytes -- (float) roll, pitch, yaw

    i = 0
    collect = 0
    rpy = [0, 0, 0] # vector to hold roll, pitch, yaw

    for item in rf_packets:

        # first item is 1.1062e-39 as it contains bits for
        # synchronization and commands
        if item > 1.1062186389011544e-40 and item <
        1.1062186389011544e-38:
            collect = 1
        else:
            if collect == 1:
                rpy[i] = item
                i = i+1

            if i == 3:
                i = 0
                collect = 0

        # set pitch, roll, and yaw
        self.spinActor.setHpr(rpy[2], rpy[1], rpy[0])

    return Task.cont

app = DizzyFixApp()
app.run()

```

References

- [1] D. E. Angelaki and K. E. Cullen, "Vestibular System: The Many Facets of a Multimodal Sense," *Annual Review of Neuroscience*, vol. 31, pp. 125-150, 2008.
- [2] K. E. Cullen, "Introduction to the Vestibular System," Internet: <http://www.medicine.mcgill.ca/physio/cullenlab/introtovest1.html> [accessed January 2012].
- [3] S. Sadeghi, K. Cullen, "Vestibular System," *Scholarpedia*, vol. 3, no. 1, pp. 3013, 2008. Internet: http://www.scholarpedia.org/article/Vestibular_system [accessed March 2012]
- [4] B.-r. Chen, "LEGSys: wireless gait evaluation system using wearable sensors," *Proceedings of the 2nd Conference on Wireless Health*, San Diego, California, 2011.
- [5] G. Liu, X. Tang, H. D. Cheng, J. Huang, and J. Liu, "A novel approach for tracking high speed skaters in sports using a panning camera," *Pattern Recognition*, vol. 42, pp. 2922-2935, 2009.
- [6] A. T. Barth, M. A. Hanson, H. C. Powell, and J. Lach, "TEMPO 3.1: A Body Area Sensor Network Platform for Continuous Movement Assessment," *Proceedings of the 6th International Workshop on Wearable and Implantable Body Sensor Networks*, 2009, pp. 71-76.
- [7] B. Najafi, T. Khan, and J. Wrobel, "Laboratory in a box: Wearable sensors and its advantages for gait analysis," *Proceedings of the IEEE Engineering in Medicine and Biology Society Annual Conference*, pp. 6407-6510, 2011.
- [8] H. Vathsangam, A. Emken, D. Spruijt-Metz, and G. S. Sukhatme, "Toward free-living walking speed estimation using Gaussian Process-based Regression with on-body accelerometers and gyroscopes," *Proceedings of the 4th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, 2010, pp. 1-8.
- [9] S. Yoonseon, S. Seungchul, K. Seunghwan, L. Doheon, and K. H. Lee, "Speed Estimation From a Tri-axial Accelerometer Using Neural Networks," *Proceedings of the IEEE Engineering in Medicine and Biology Society Annual Conference*, 2007, pp. 3224-3227.
- [10] S. Chen, C. L. Cunningham, B. C. Bennett, and J. Lach, "Enabling longitudinal assessment of ankle-foot orthosis efficacy for children with cerebral palsy," *Proceedings of the 2nd Conference on Wireless Health*, San Diego, California, 2011, pp. 37-46.

- [11] A. T. Barth, B. Boudaoud, J. S. Brantley, S. Chen, C. L. Cunningham, T. Kim, J. Harry C. Powell, S. A. Ridenour, J. Lach, and B. C. Bennett, "Longitudinal high-fidelity gait analysis with wireless inertial body sensors," *Proceedings of the 1st Conference on Wireless Health*, San Diego, California, 2010, pp. 192-193.
- [12] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," *Proceedings of the 17th conference on Innovative applications of artificial intelligence*, Pittsburgh, Pennsylvania, 2005.
- [13] J. Suutala, S. Pirttikangas, and J. Rönning, "Discriminative Temporal Smoothing for Activity Recognition from Wearable Sensors," *Ubiquitous Computing Systems*, vol. 4836, 2007, pp. 182-195.
- [14] D. A. James, N. Davey, and T. Rice, "An accelerometer based sensor platform for in situ elite athlete performance analysis," in *Proceedings of IEEE Sensors*, 2004, pp. 1373-1376 vol.3.
- [15] N. Langholz, "Pulling Punches: A Nonparametric Regression Approach to Punch Force Estimation," *CENS (Center for Embedded Networks Sensing) Technical Seminar Series*, University of California, Los Angeles, January 2012.
<http://research.cens.ucla.edu/events/2012/01/27/Nathan%20Langholz%20Powerpoint1.pdf>
- [16] P. J. Troped, M. S. Oliveira, C. E. Matthews, E.K. Cromley, S. J. Melly, and B. Craig, "Prediction of activity mode with global positioning system and accelerometer data," *Medicine and Science in Sports and Exercise*, vol. 40, pp. 972-978, May 2008.
- [17] APDM, "Opal," Internet: <http://apdm.com/products/movement-monitors/opal/> [accessed January 2012].
- [18] Xsens, "MTw," Internet: <http://www.xsens.com/en/mtw> [accessed January 2012].
- [19] MotionNode, "MotionNode Bus," Internet: http://www.motionnode.com/MotionNode_Bus_Specification.pdf [accessed January 2012].
- [20] Memsense, "Nano IMU," Internet:
<http://www.memsense.com/index.php/Product-Pages/nimu-miniature-light-weight-3d-digital-output-sensor/menu-id-162.html> [accessed January 2012].
- [21] Cleveland Medical Devices Inc., "Kinetisense," Internet: http://www.clevemed.com/products/kinetisense_overview.shtml [accessed January 2012].
- [22] V. Loseu, J. Mannil, and R. Jafari, "Lightweight power aware and scalable movement monitoring for wearable computers: a mining and recognition technique at the fingertip of sensors," *Proceedings of the 2nd Conference on Wireless Health*, San Diego, 2011, pp. 67-76

- [23] ARM, "ARM Processors overview," Internet: <http://www.arm.com/products/processors/index.php> [accessed February 2012].
- [24] ST Microelectronics, "STEVAL-MKI062V2: iNertial Module (iNEMO)," Internet: <http://www.st.com/internet/evalboard/product/250367.jsp> [accessed January 2012].
- [25] N. Abbate, I. Aleo, A. Basile, C. Brigante, and A. Faulisi, "Design of an Inertial Motion Module," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1-4.
- [26] Sparkfun Electronics, "IMU 6DOF v4 Sensor Board," Internet: <http://www.sparkfun.com/products/8726> [accessed January 2012].
- [27] Analog Devices, "Triaxial Inertial Sensor with Magnetometer," Internet: http://www.analog.com/static/imported-files/data_sheets/ADIS16400_16405.pdf [accessed January 2012].
- [28] ST Microelectronics, "STM32W-EXT," Internet: <http://www.st.com/internet/evalboard/product/247100.jsp> [accessed January 2012].
- [29] Digi International Inc., "XBee ZB," Internet: <http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/zigbee-mesh-module/xbee-zb-module> [accessed January 2012].
- [30] Embedded Platform Lab, National Tsing Hua University , "EcoMote," Internet: <http://www.ecomote.net/> [accessed January 2012].
- [31] Digi International Inc. "Xtreme OEM," Internet: http://www.digi.com/pdf/ds_xstreammodule.pdf [accessed January 2012].
- [32] Texas Instruments, "MSP430 Wireless Development Tool – EZ430-RF2500," Internet: <http://www.ti.com/tool/ez430-rf2500> [accessed January 2012].
- [33] Z. Zilic and K. Radecka, "Fault Tolerant Glucose Sensor Readout and Recalibration," *Proceedings of the 2nd Conference on Wireless Health*, San Diego, California, 2011. pp. 169 – 170.
- [34] A. Giordano and F. M. Hsu. *Least Square Estimation with Applications to Digital Signal Processing*, New York, NY, USA: John Wiley & Sons, Inc.,1985.
- [35] ST Microelectronics, "AN3192: Application Note - Using LSM303DLH for a tilt compensated electronic compass," August 2010.
- [36] Z. Zilic and B. Karajica, "High-level design of integrated microsystems - arithmetic perspective," in *Proceedings of the IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, 2011, pp. 77-82.
- [37] L. B. Jackson, *Digital Filters and Signal Processing*, Norwell MA: Kluwer Academic Publishers, 1996, pp. 290-313.

- [38] Y. Fan and Z. Zilic, "Bit Error Rate Testing of Communication Interfaces," *IEEE Transactions on Instrumentation and Measurements*, Vol. 57, No. 5, May 2008, pp. 897-906.
- [39] S. P. Tseng, L. Wen-Lung, S. Chih-Yang, H. Jia-Wei, and C. Chin-Sheng, "Motion and attitude estimation using inertial measurements with complementary filter," in *Proceedings of the 8th Asian Control Conference (ASCC)*, 2011, pp. 863-868.
- [40] N. Abbate, A. Basile, C. Brigante, A. Faulisi, and F. La Rosa, "Modern Breakthrough Technologies Enable New Applications Based on IMU Systems," *Journal of Sensors*, vol. 2011, Article ID 707498, 2011.
- [41] Keil, "RTOS and Middleware," Internet: <http://www.keil.com/rtos/> [accessed November 2011].
- [42] FreeRTOS, "The FreeRTOS Project," Internet: <http://www.freertos.org/> [accessed November 2011].
- [43] ST Microelectronics, "RM008: Reference manual for STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced ARM-based 32-bit MCUs," Internet: <http://www.st.com/internet/mcu/subclass/1169.jsp> [accessed February 2012].
- [44] Keil, "Flash File System," Internet: <http://www.keil.com/rl-arm/rl-flash.asp> [accessed November 2011].
- [45] ChaN - The Electronic Lives Manufacturing, "FatFs Generic FAT File System Module," Internet: http://elm-chan.org/fsw/ff/00index_e.html [accessed November 2011].
- [46] Clearwater Clinical Ltd., "DizzyFIX: A natural treatment for vertigo and dizziness," Internet: <http://www.dizzyfix.com/dizzyfix>, [accessed March 2012].
- [47] Carnegie Mellon University, "Panda3D," Internet: <http://www.panda3d.org/>, [accessed February 2012].
- [48] H. Zhang, "Exploring conditions for the optimality of naive Bayes," *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 19, No. 2, 2005.
- [49] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-Supervised Self-Training of Object Detection Models," in *Proceedings of the Seventh IEEE Workshops on Application of Computer Vision*, 2005, pp. 29-36.
- [50] X. Guorong, Z. Wei, and C. Peiqi, "EM algorithms of Gaussian mixture model and hidden Markov model," in *Proceedings of the International Conference on Image Processing*, 2001, pp. 145-148.
- [51] D. G. Kleinbaum and M. Klein, "Logistic Regression," *Maximum Likelihood Techniques: An Overview*, ed: Springer New York, 2010, pp. 103-127.

- [52] Z. Wang, G. Q. Liu, and J. C. Guo, "An Improved K-Means Algorithm Based on Multiple Feature Points," in *Proceedings of the International Workshop on Intelligent Systems and Applications*, 2009, pp. 1-5.
- [53] International Skating Union (ISU), "Special Regulations and Technical Rules – Speed skating and short track speed skating," *53rd Ordinary Congress*, pp. 115, June 2010. Internet: <http://www.isu.org/vsite/vfile/page/fileurl/0,11040,4844-203082-220305-167852-0-file,00.pdf>
- [54] Grbic, S. Brown, S. Caranci, R. Grindley, M. Gusat, G. Lemieux, K. Loveless, N. Manjikian, S. Srbljic, M. Stumm, Z. Vranesic and Z. Zilic, "Design and Implementation of the NUMachine Multiprocessor", *Proceedings of 35th ACM/IEEE Design Automation Conference DAC '98*, pp.66-69, San Francisco, June 1998.
- [55] S. Brown, N. Manjikian, Z. Vranesic, S. Caranci, A. Grbic, R. Grindley, M. Gusat, K. Loveless, Z. Zilic and S. Srbljic, "Experience in Designing a Large-Scale Multiprocessor using Field-Programmable Devices and Advanced CAD Tools", *Proceedings of 33rd ACM/IEEE Design Automation Conference DAC '96*, pp. 24-29, Las Vegas, June 1996.
- [56] K. Radecka and Z. Zilic, "Arithmetic Transforms for Compositions of Sequential and Imprecise Datapaths", *IEEE Transactions on Computer Aided Design (CAD) of Integrated Circuits and Systems*, Vol. 25, No. 7, pp. 1382-1391, July 2006.
- [57] Y. Pang, K. Radecka and Z. Zilic, "Optimization of Imprecise Circuits Represented by Taylor Series and Real-Valued Polynomials", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 29, No. 8, Aug. 2010, pp. 1177-1190.
- [58] Y. Pang, K. Radecka and Z. Zilic, "An Efficient Hybrid Engine to Perform Range Analysis and Allocate Integer Bit-widths for Arithmetic Circuits", *Proceedings of ACM/IEEE Asia and South Pacific Design Automation Conference, ASP-DAC 2011*, Jan. 2011, pp. 455-460.
- [59] K. Radecka and Z. Zilic, "Design Verification by Test Vectors and Arithmetic Transform Universal Test Set", *IEEE Transactions on Computers*, Vol. 53, No. 5, pp. 628-640, May. 2004.
- [60] Z. Zilic and Z. Vranesic, "A Multiple-Valued Reed-Muller Transform for Incompletely Specified Functions", *IEEE Transactions on Computers*, vol. 44, No. 8, pp. 1012-1020, August 1995.
- [61] M. Boulé, J-S. Chenard and Z. Zilic, "Debug Enhancements in Assertion-Checker Generation", *IET Computers and Digital Techniques*, Vol. 1, No. 6, pp. 669-677, Nov. 2007.

- [62] M.H. Neishaburi and Z. Zilic, "Enhanced Reliability Aware NoC Router", *Proc. Intl. Symposium on Quality Electronic Design*, ISQED 2011, Mar. 2011
- [63] M.H. Neishabouri and Z. Zilic, "Reliability Aware NoC Router Architecture Using Input Buffer Sharing", *Proceedings of Great Lakes Symposium on VLSI*, pp. 511-516, May 2009.
- [64] Y. Fan and Z. Zilic, "*Accelerating Test, Validation and Debug of High Speed Serial Interfaces*", Springer Verlag. 2011.
- [65] M. Janidarmian, Z. Zilic and K. Radecka, "Issues in Multi-Valued Multi-Modal Sensor Fusion", *Proceedings of IEEE International Symposium on Multiple-Valued Logic*, ISMVL, 2012.