Wireless Realtime Motion Tracking System using Localised Orientation Estimation

Alexander D Young

Doctor of Philosophy
Institute of Computing Systems Architecture
School of Informatics
University of Edinburgh
2010

Abstract

A realtime wireless motion tracking system is developed. The system is capable of tracking the orientations of multiple wireless sensors, using a semi-distributed implementation to reduce network bandwidth and latency, to produce real-time animation of rigid body models, such as the human skeleton. The system has been demonstrated to be capable of full-body posture tracking of a human subject using fifteen devices communicating with a basestation over a single, low bandwidth, radio channel.

The thesis covers the theory, design, and implementation of the tracking platform, the evaluation of the platform's performance, and presents a summary of possible future applications.

Acknowledgements

My sincere thanks to the many people who helped me throughout the duration of my studies.

Thanks to the members of the SMART centre gait analysis laboratory, Red Kite animation, and Strange Company for their feedback and interest.

Thanks to the members of the Speckled Computing Consortium. In particular: my supervisor, D.K. Arvind; Martin Ling, for his involvement in firmware implementation and hardware design; and Mat Barnes, for many useful discussions and comments.

Finally, thanks to my parents, Viola and Stephen; my flatmates, Joe, Erin, and Chus; my friends, Claire, Cristina, Despina, Graham, Ross, Russell, and Ryan; and to all my other friends and family for their help and support.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Alexander D Young*)

Table of Contents

List of Figures

Part I

Introduction

Chapter 1

Introduction

1.1 The Study of Motion

The study of motion has fascinated people throughout history. Advances in technology have made it possible to capture motion in ever easier and more accurate ways.

1.1.1 History of Biomechanics

As with many scientific subjects the study of motion, and in particular human motion, can be traced back to the ancient Greeks [2]. The fundamental concepts of deductive and mathematical reasoning can be traced back to Socrates, Plato and Aristotle. Indeed, Aristotle (-384 – -322) is regarded to have written the first ever book on biomechanics, "On the Movement of Animals", in which he sees animal bodies as mechanical systems.

From the time of the ancient Greeks until the dawn of the Renaissance little progress was made on the advancement of biomechanical science. Leonardo da Vinci (1452 – 1519) produced many detailed studies of human anatomy claiming that:

> "it is indispensable for a painter, to become totally familiar with the anatomy of nerves, bones, muscles, and sinews, such that he understands for their various motions and stresses, which sinews or which muscle causes a particular motion" [3].

However, much of this work remained unpublished for centuries and so cannot be considered to have had great scientific impact [2].

The father of modern biomechanics, Giovanni Alfonso Borelli (1608 – 1679), heavily influenced by the work of Galileo Galilee (1564 – 1642), was the first to understand the musculoskeletal system as a set of levers that magnified motion rather than force. He worked out the forces of equilibrium of the joints of a standing body, many years before Newton was to produce his laws of motion, calculated the centre of gravity of the human body and performed experiments to measure volumes of inspired and expired air.

From the time of Borelli, little work was produced in the area of biomechanics. However, the Age of Enlightenment brought many important advances in mathematics, most importantly the work of Isaac Newton (1642 – 1727) on the laws of motion and René Descartes (1596 – 1650) on geometrical algebra. It was not until the latter half of the 19th century and the development of motion capture that biomechanics took off as a major research area.

1.1.2 Motion Capture

Early motion capture was based on the techniques of chronophotography developed by Etienne-Jules Marey (1830 – 1904). Marey's technique made it possible to capture the phases of motion on a single photographic plate, as illustrated in Figure 1.1. Suddenly, for the first time, it was

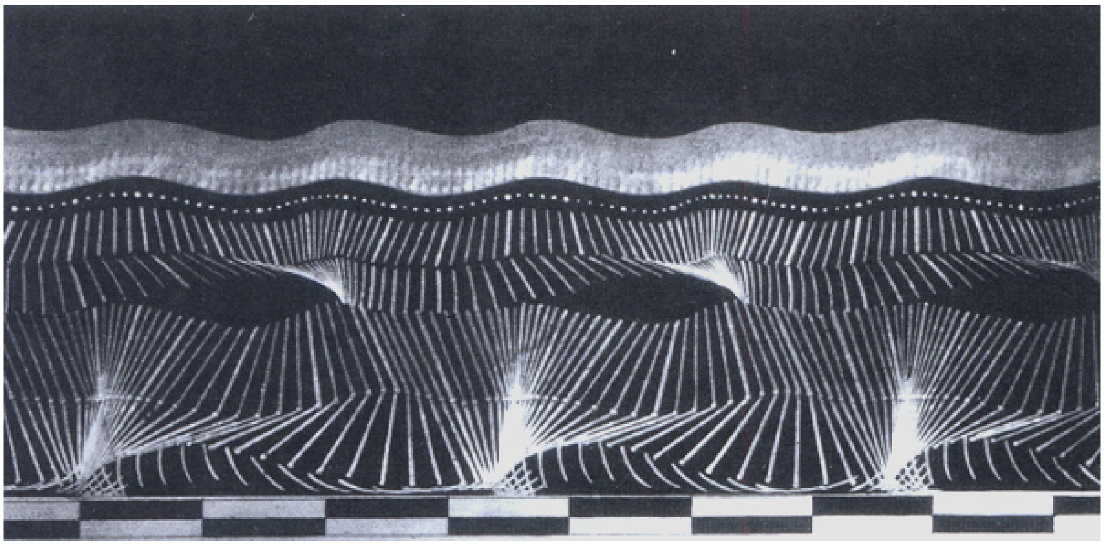


Figure 1.1: Human Motion – Etienne-Jules Marey 1883

possible to accurately record the progression of the limbs. Marey was responsible for many advances in the study of human motion, including being the first person to correlate motion with ground reaction forces [4].

Chronophotography was further developed by Eadweard Muybridge (1830 – 1904) who used multiple still cameras to capture rapid motion. Muybridge produced many photographic studies of motion, including subjects descending staircases, boxing, children walking and most famously demonstrated that a horse at full gallop lifts all four hooves off the ground, illustrated in Figure 1.2. This result, while long conjectured, had never previously been demonstrated.

In addition to creating the field of optical motion capture the early work of Marey on chronophotography was to later evolve into the cinematography that we know today. Modern films including computer generated effects and 3D animation have reunited the fields of motion capture and cinematography to produce amazing computer-generated characters capable of interacting with real actors in a highly realistic manner.

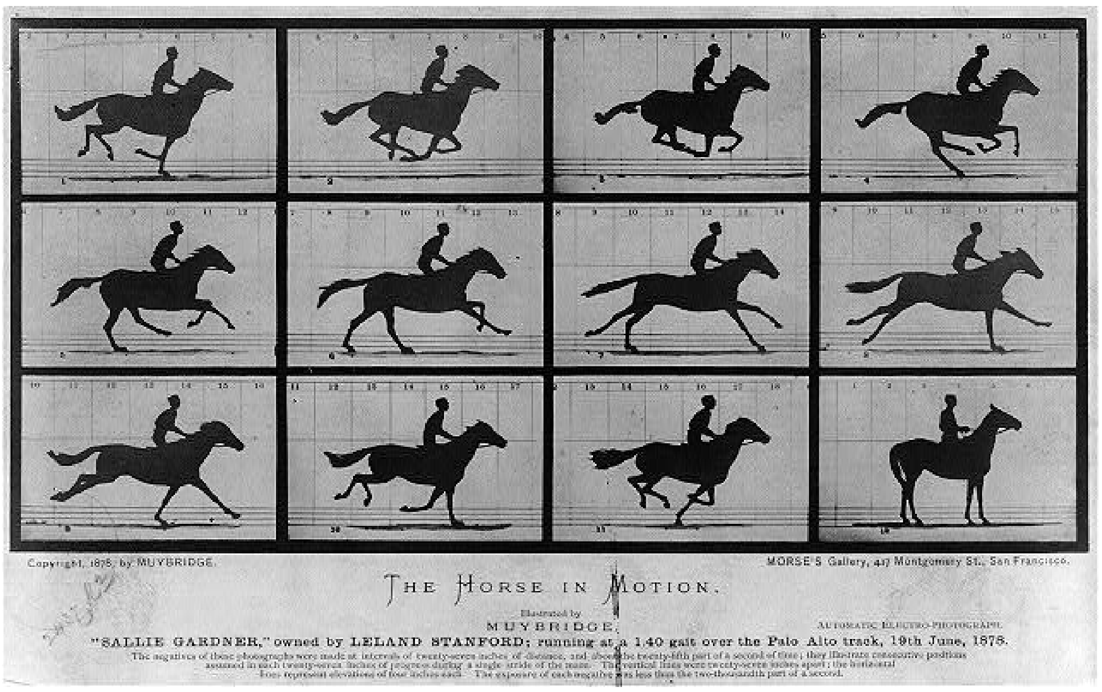

Copyright, 1878, by MUYBRIDGE.
MORSE'S Gallery, 417 Montgomery St., San Francisco.
THE HORSE IN MOTION.
Illustrated by
MUYBRIDGE.
AUTOMATIC ELECTRO-PHOTOGRAPH.
"SALLIE GARDNER," owned by LELAND STANFORD; running at a 1.40 gait over the Palo Alto track, 19th June, 1878.

Figure 1.2: “The Horse in Motion” – Eadweard Muybridge 1878

1.1.3 State-of-the-Art Motion Capture

Optical motion capture systems have advanced far beyond the early designs used by Marey and Muybridge. Such systems are capable of tracking many points at very high speed, with great accuracy, in three dimensions. However, some fundamental limitations of optical methods remain. Primarily, optical systems will always be limited in tracking volume by the coverage of the available cameras. Additionally the complexities of modern optical tracking solutions make them very expensive, far beyond the reach of casual users or even small companies.

Several alternatives to optical tracking have been developed. These are:

Mechanical Tracking Joint angles are tracked using an instrumented exoskeleton containing potentiometers or other rotational encoders at the joints. Such systems are typically cumbersome to wear and present serious difficulties in aligning external instrumented joints with the internal joint being tracked. This is a particular challenge for joints with multiple degrees of freedom, such as the shoulder [5]. However, mechanical systems have the advantage that they are self-contained, allowing tracking over large areas without external infrastructure.

Acoustic Tracking The position of markers is tracked using multi-lateration of ultrasonic time of flight measurements. Such systems share the disadvantages of optical systems in that they require line-of-sight links and have limited coverage areas. Accuracy is further limited by multipath effects.

Magnetic Tracking The positions and orientations of sensors are tracked based on observations of a generated magnetic field. Such systems have the advantage that the body is mostly transparent to magnetic fields and so there is no limitation to line-of-sight links. However, magnetic field strength decreases rapidly with distance from the source, limiting tracking area, and the field is easily distorted by ferrous objects in the surrounding area.

Inertial Tracking Inertial motion capture is based on measurements of rotational and linear accelerations or velocities to estimate rotation and position. Inertial tracking has the advantage that, like mechanical tracking, it is a self-contained system. However, use of low cost sensors limit the accuracy of such systems as integration of noisy sensed data causes cumulative errors to build up. Recent advances in data fusion and sensor miniaturisation have allowed the development of small drift-corrected inertial rotation sensors [6, 7, 5].

Of the current motion tracking technologies available today, inertial tracking presents the most interest as it has the potential to provide truly limitless tracking. Unlike the source-based tracking solutions, such as optical, acoustic and magnetic tracking, there is no need to provide extensive infrastructure. In comparison to mechanical tracking, inertial tracking is preferable as it does not require awkward positioning of sensors over the subject's joints. However, today's inertial tracking systems require a wired network of motion sensors, mounted in a fitted body suit, and a significant amount of processing power.

1.2 Wireless Sensor Networks

Advances in the design of miniature sensors, low power processors, radios and supporting electronics, along with advances in battery chemistry, have allowed the development of small, low cost, Wireless Sensor Networks(WSNs) [8]. By distributing processors along with sensors throughout the network, WSNs present interesting opportunities to exploit localised data processing to reduce the requirements for network bandwidth and centralised processing power [9] encountered in earlier wireless sensor systems.

Work within the wireless sensor network community has concentrated on the development of protocols and applications utilising ad-hoc networks consisting of large numbers of devices [10]. For this reason, work within the community has mainly focussed on areas such as location inference, low power routing and medium access control protocols and data aggregation [11].

The advent of Micro-Electro-Mechanical Systems (MEMS) has allowed the building of minute sensors such as accelerometers and gyroscopes, necessary for development of inertial motion tracking. These sensors have found application in Body Sensor Networks (BSNs) used for activity detection and gesture recognition [12, 13, 14, 15, 16]. However, within the wireless sensor community as a whole, with its focus on large scale ad-hoc deployments, little attention has been paid to the possibility of using similar hardware and design concepts to develop a full body wireless motion tracking systems.

It is proposed that, by utilising localised processing of sensed data to reduce network bandwidth and centralised processing requirements, a wireless sensor network, composed of multiple inertial sensing devices, could operate as a low cost, flexible, non-restrictive, motion tracking system.

1.3 Motivating Applications

Motion capture has many potential applications. In the context of this thesis two applications will be considered as motivations: gait analysis and animation. These two applications will provide the driving force behind design decisions as well as a context in which to evaluate performance.

1.3.1 Gait Analysis

Gait analysis is often used both for diagnosis, where conditions such as Parkinson's disease can be first detected by alterations in gait [17], and rehabilitation following injury directly to limbs or to the brain's motor control faculties following a stroke.

Traditionally gait analysis has used visual assessment to assess the range of motion of the limbs. Such a technique is limited in accuracy with even trained professionals unlikely to be able to differentiate changes of less than 5° [18]. To overcome this limit, mechanical goniometers have been used, but these are not much more accurate due to problems of mounting in relation to the joints of interest.

More recently dedicated gait analysis laboratories have been equipped with state-of-the-art optical motion trackers and force pads. These allow more accurate capture under laboratory conditions. However, the requirement to bring subjects into specialised laboratories and to apply numerous markers present their own problems. Subjects, especially young children, when faced with such surroundings will affect unnatural gait and tire quickly.

In the gait analysis application the following desirable features can be identified:

1. Low cost

   Lower costs are important to move gait analysis away from specialist facilities and into mainstream use. Low cost systems may well be sufficient to provide basic analysis of gait problems which can be referred to specialist units for further investigation if necessary.

2. Minimal intrusion to subject

Reducing the disturbance to the subject should allow for more natural capture. Lightweight sensors, with no limits on tracking area, could be used to run extended tests over the course of a subject's daily routine.

3. Easy to use

   To allow data to be gathered easily, minimal specialised training should be required to use the system. Ideally, the system should be easy enough to use so that it could be used unsupervised.

4. Accurate

   The system should have sufficient accuracy to be reliably used for clinical diagnosis. In particular results should be repeatable.

1.3.2 Animation

In traditional animation each frame is carefully hand drawn. This painstaking process means that it can take many months to produce even relatively short animated films. The advent of computers allows automation of some aspects of this work but it still requires great skill to animate a character in a realistic way. This problem becomes even harder in modern 3D animation.

Motion capture allows rapid input of complex motion using professional performers. This can greatly reduce the time required to produce an animated feature and increase realism. The ability to produce live previews of capture data allows directors to quickly assess the performance and re-shoot as necessary. Sophisticated tools exist to allow the retargeting of motion data to character models of varying dimensions.

Motion capture for animation has traditionally used optical capture methods. These can produce very good results but are limited in key areas. Cost is a primary issue, with only large animation studios able to purchase and maintain their own equipment. Marker occlusion also presents problems when dealing with complex sets or multiple interacting characters.

In the animation application the following desirable features can be identified:

1. Low cost

   Low system cost lowers the barrier to entry of the motion capture market encouraging independent studios and amateurs to produce exciting and innovative work.

2. Easy to use

   The system should not require extensive training to setup and maintain. This again helps to lower barriers to widespread use.

3. Live feedback on capture

   Instant feedback on performance allows for more interaction between director and performer. This allows directors to capture the exact motion they need, rather than have to capture blindly and cut later to achieve the desired result.

4. Moderate accuracy

   Accuracy is judged purely on aesthetic qualities rather than scientific precision. As such it is important that movements be fluid, rather than showing rapid corrections, even if this causes increased overall error.

1.3.3 Summary of Application Goals

Both motivating applications show common desirable characteristics. Reducing system cost allows motion capture technology to reach new users. As inertial tracking devices, requiring no investment in infrastructure, can be made for a fraction of the cost of optical systems, and often require far less post processing, they can potentially lower the cost of entry into the motion capture market.

Ease of use is important both to reduce training and maintenance costs and also to encourage users from non-technological backgrounds to experiment with the system and find new and novel applications. While mechanical tracking may present the cheapest option, the difficulty of accurately aligning mechanical joints, and the impediment of wearing the required exoskeleton, is likely to frustrate users.

Live feedback on capture allows faster decisions to be made about the quality and suitability of the capture performance. Inertial tracking again presents an advantage here, as it does not suffer from problems of occlusion and reflection that can require significant post-processing to rectify.

1.4 Summary

Motion tracking has many applications in areas such as animation, health care, and human computer interaction. Traditional motion capture has mainly concentrated on optical tracking methods that have major limitations in terms of cost and the requirement of large dedicated capture areas. Such limitations have prevented motion capture technology from being accessible to the general public.

The development of wireless sensor network technologies incorporating low power, autonomous, sensing, processing and communication in small packages, along with developments in inertial tracking allow the design of low cost, easy to use motion capture systems.

Such systems, with their ability to be used outside of existing motion capture facilities, present an exciting opportunity to increase access to motion tracking technology.

While inertial tracking promises a vision of unencumbered motion tracking, today's systems, requiring wired body suits, cannot achieve this full potential.

1.5 Research Direction and Contributions

The goal of this thesis is to examine the design, implementation and analysis of a completely wireless realtime motion capture system. The system should aim to be low cost, easy to use, and be reasonably accurate for tracking everyday human motion while maintaining realtime feedback.

A low-power body sensor network for motion tracking will be developed, each node of the network capable of estimating its own orientation in real time. In order to achieve this a simple orientation algorithm will be developed. The algorithm will be shown to achieve comparable accuracy for reduced implementation cost compared to existing algorithms.

The use of local processing will be demonstrated as the key development in achieving full-body realtime posture tracking, significantly reducing required network bandwidth and increasing tolerance of intermittent packet loss.

The behaviour of the orientation filter will be analysed and dynamic linear accelerations due to subject motion shown to be the primary source of errors. Preliminary work on a novel collaborative orientation algorithm, incorporating estimation of linear acceleration, is presented as the basis for future work on improving system accuracy.

1.5.1 Contributions

The key contributions of this thesis are:

- The development of a low-complexity orientation estimation algorithm suitable for use on low-power body sensor network devices.

- The development of the first integrated system comprising hardware, firmware, and software for full-body wireless realtime posture tracking.

- The analysis of orientation estimation algorithm errors indicating that dynamic linear accelerations are the primary source of errors.

- The development of a novel distributed orientation estimation algorithm utilising collaborative linear acceleration estimation.

1.5.2 Publications

The work presented in the thesis has been the basis for the following publications by the author:

Young et al. [19] discusses the implementation of the *Orient* hardware and software. Details of the orientation processing, device power requirements, and network performance are presented. This paper is the first publication of a wireless body sensor network capable of realtime full-body posture tracking.

Young [20] provides a comparison of orientation filter algorithms, considering implementation costs and accuracy for a walking subject. The algorithm used by the *Orient* system is shown to have the lowest computational requirements, while maintaining comparable accuracy to existing alternative algorithms.

Young and Ling [21] discusses the effects of dependencies between data packets when working with an unreliable network link. Multi-device wireless posture tracking is used as an example to illustrate the questions to consider during network design. Processing of data locally on sensing devices is shown to provide the greatest resilience to intermittent packet loss.

Young et al. [22] discusses the orientation errors caused by linear acceleration and presents an algorithm for estimating linear acceleration in a distributed manner. Preliminary results based on combined optical and inertial capture are presented.

Young [23] provides further simulated validation of the linear acceleration estimation algorithm proposed in [22], and demonstrates the use of an objective function for automated optimisation of body model proportions.

Lympouridis et al. [24] presents the integration of data from an *Orient* network into the MAX-MSP sound synthesis environment for the development of interactive virtual musical instruments.

Additional publications utilising the work presented are:

Lympourides et al. [25] discusses the requirements for improvisational performances, and the suitability of the *Orient* system for such applications.

Arvind and Bates [26] discusses the use of the *Orient* sensors in the real-time analysis of golf strokes. The paper also demonstrates the use of a PDA to allow unrestricted use of the system.

Arvind and Valtazanos [27] utilises *Orient* sensors to measure the similarity between the members of a dance couple performing the tango.

Arvind and Bartosik [28] discusses the analysis of *Orient* data for training bipedal robots to walk.

Bates et al. [29] discusses the use of accelerometer data for monitoring breathing rate of hospital patients. Uses Orient devices for data capture.

1.6 Thesis Overview

The thesis is divided in to nine chapters in four sections. Chapter 2 presents a review of existing work in related areas of inertial motion capture and wireless sensor networks.

Part II of the thesis is concerned with the development of the theoretical aspects of orientation estimation. In Chapter 3 the required theory of body modelling, orientation estimation and the development of an optimised filter design are presented. Simulated results to demonstrate basic operation of the filter are given in Chapter 4.

Part III outlines the design and evaluation of the *Orient* sensor platform. The implementation details of the hardware, firmware and software are given in Chapter 5. Chapter 6 presents results of orientation estimation accuracy while Chapter 7 presents an evaluation of the entire system.

Part IV concludes by presenting ongoing work and suggestions for future improvements and directions for future research in Chapter 8. The conclusions of the thesis are presented in Chapter 9.

Chapter 2

Review of Existing Inertial Trackers

This chapter presents an overview of existing work in related areas. Commercial inertial tracking solutions and their limitations are discussed. This is followed by a brief discussion of academic research into the problem of orientation estimation. Finally, a review of relevant research within the wireless sensor network community is presented.

2.1 Overview of Inertial Motion Tracking

Inertial motion tracking captures the posture of a subject by gathering the orientation of the subject's limbs and applying these rotations to a model of the subject. Limb orientation is gathered using miniature orientation sensors attached to each limb segment. Full details of the mapping process are presented in Chapter 3.

In order to capture the full posture of the subject, fifteen sensors are required [30]. These sensors must be capable of producing an orientation estimate relative to a common co-ordinate frame. The ability of a device to estimate its orientation depends on the available sensors. Simple devices may only be capable of estimating the rotation from a known initial rotation. As such devices estimate the cumulative rotation over time, they are liable to drift away from the true orientation as errors are accumulated. More complex devices can additionally estimate the initial orientation and provide estimate drift correction.

Fully drift corrected orientation sensors use a combination of three sensor types [7]:

Magnetometers provide a measurement of the Earth's magnetic field. The horizontal component of the Earth's magnetic field allows the heading of a device to be estimated, as with a normal magnetic compass.

Linear accelerometers provide a measurement of the device acceleration, including the acceleration due to gravity. By filtering out the high frequency accelerations due to movement, the direction of the local gravity vector can be estimated. Observation of the gravity vector allows the pitch and roll angles to be estimated.

Rate gyroscopes give a measurement of the angular velocity of the device. Integration of the angular velocity results in an estimate of the orientation relative to a known initial orientation.

Data from the magnetometers and accelerometers combine to provide an estimate of orientation relative to an Earth-fixed co-ordinate frame. Rate gyroscope data is used to estimate relative rotation during rapid movements when linear accelerations dominate the accelerometer readings.

2.2 Commercial Inertial Trackers

There are many commercial companies that produce inertial tracking products. However, the majority of products are designed for navigation systems and are considerably too large for use in human motion tracking. There are three inertial motion capture devices worth considering in terms of full body capture:

InterSense Wireless InertiaCube3 The InertiaCube3 [31] is an integrated inertial/magnetic 3-Degree of Freedom (3-DoF) orientation tracker, employing a Kalman filter for orientation estimation. The device is available in both wired and wireless versions. A maximum of four wireless InertiaCubes can be tracked using a single receiver although multiple receivers may be used to support up to thirty-two devices.

An external processor unit is available to reduce processing load on the host PC by offloading the processing of the orientation filter.

Xsens MTx The MTx [32] is a wired inertial/magnetic unit capable of 3-DoF tracking. Multiple MTx devices may be combined, using the proprietary Xbus interface, to build complete motion tracking systems. Each Xbus master unit can support up to ten devices. No details are available about the maximum number of supported devices.

MicroStrain Inertia-Link The Inertia-Link [33] is a purely inertial 3-DoF tracker and is worth mentioning as it supports onboard orientation processing and wireless uplink using a low-bandwidth IEEE 802.15.4 compatible radio. Multiple devices are supported although not using a scheduled delivery system.

2.2.1 Full Body Inertial Motion Capture

Two full body inertial motion capture systems are available: the Xsens Moven [34] and the Animazoo GypsyGyro-18. Both systems use a wired network of inertial/magnetic sensors, with wireless uplink to a PC for processing. Sensors are mounted on a Lycra body-suit that acts as both a mounting solution and cable guide. The Moven uses 16 MTx units while the Gypsy uses 18 modified InertiaCubes.

The systems are similar in specification, both weighing approximately 2kg and having runtimes of up to three hours. Processing requirements are significant with both systems recommending at least a 2.6GHz Pentium 4 processor for the host system.

2.2.2 Summary of Commercial Devices

Table 2.1 shows a comparison of the commercial orientation tracking platforms.

Property	Device		
	InterSense Wireless InertiaCube3	Xsens MTx	MicroStrain Inertia-Link
Wireless	Yes	No	Yes
Local Processing	No	No	Yes
Multiple Device Support	4 per receiver, 32 total	10 per Xbus	Yes, no details of maximum.
Maximum Update Rate (Hz)	180 (2 devices) 120 (4 devices)	120	100 (wireless) 250 (wired)
Maximum Rotational Rate (°/s)	1200		
Static Accuracy Heading Pitch & Roll	 1° RMS 0.25° RMS	 1° 0.5°	 0.5° 0.5°
Dynamic Accuracy	As Static	2° RMS	2°
Power (mW)	240	360	450
Dimensions (mm)	$31.3 \times 43.2 \times 14.8$	$38 \times 53 \times 21$	$41 \times 63 \times 24$
OS Support	Windows Linux MacOS X	Windows Linux	Windows

Table 2.1: Comparison of Commercial Motion Trackers

Of the available systems only the InterSense InertiaCube3 supports fully wireless tracking of multiple devices. However, wireless tracking is limited to four devices per receiver as raw data is transmitted for processing by the host PC or dedicated processor. Full body tracking is only available using wired body suits to network multiple sensors.

2.3 Inertial Sensing in Research

Research into inertial sensing is undertaken in two main areas: robotic navigation, and virtual and augmented reality.

2.3.1 Robotic Navigation

Work in robotics deals with the problem of tracking the movement of mobile robots as they travel through an environment. As with motion capture, inertial tracking is interesting as it provides a tracking solution without the requirement of external infrastructure.

Inertial robotic tracking is a difficult problem as there are many sources of noise in the environment. Magnetic field measurements are disrupted by interference from motors and moving ferrous materials, there may be substantial vibration affecting accelerometer measurements and there is also great potential for electrical noise. To combat these problems complex filters are used such as those developed by Harada *et al.* [35, 36, 37]. These filters, modifications of the standard Kalman filter, provide good estimation performance but are computationally intensive.

One advantage in the robotic tracking problem is that the dynamics and control inputs of the system are well known allowing good system modelling and prediction to be performed.

2.3.2 Virtual Reality

Much work has been performed on using inertial sensing to insert humans into virtual environments by Bachmann *et al*. [38, 7]. This work has included the design of wired inertial/magnetic sensors for use in human posture tracking. This work will be discussed in greater detail in Section 3.3. To summarise, work has been directed towards producing a reduced cost complementary Kalman filter suitable for realtime orientation tracking. Although this produces lower cost filters than traditional Kalman filters, it still appears too computationally expensive for use on an embedded platform. Current results indicate a processing time of 1.6ms per update [38] for a Java implementation, although no details are given as to the processor used. It is safe to assume that a modern desktop processor would be at least an order of magnitude more powerful than a low power embedded processor putting this filter out of the capabilities of an embedded implementation.

2.3.3 Summary

Existing research into inertial tracking has concentrated on the design of orientation estimation filters that are optimal in terms of Mean Square Error (MSE). For this reason it is common to choose variants on the Kalman filter structure. Although filters with comparable MSE performance are chosen to reduce processing requirements this is not seen as the primary goal.

The complexity of existing filters precludes their use in low power embedded systems, instead requiring external host processing.

2.4 Inertial Sensing in Wireless Sensor Networks

Wireless sensor network nodes increasingly incorporate inertial sensors such as accelerometers and gyroscopes as advances in MEMS technology reduce cost, power consumption and size. Many projects have utilised these sensors for activity detection and gestural recognition. However, few have attempted to provide full motion tracking abilities.

A brief summary of general purpose WSNs is presented, followed by further discussion of the most relevant projects.

2.4.1 General Purpose Wireless Sensor Networks

2.4.1.1 Hardware

Many hardware platforms have been designed for use as WSN nodes. Early platforms, such as the Berkeley Mica [39] mote, were based on 8-bit micro-controllers, such as the Atmel AVR ATMega series, and used low bandwidth, narrowband, radios for communication. These early devices were limited by the use of 8-bit processors with little inbuilt RAM or program FLASH. The radios in early devices were typically low frequency devices, such as the CC1000 from ChipCon, operating in the 433, 868 and 915MHz bands. These radios were selected due to their good propagation properties and low power consumption. However, these lower frequencies required large antennas.

The drive to miniaturise devices has led to the use of higher frequency radios, such as the ChipCon CC2420, that operate in the 2.45GHz Industrial Scientific and Medical (ISM) band. The use of such radios has allowed the use of smaller antennas and higher data-rates, but come at a cost in terms of power consumption and propagation. The use of the 2.45GHz band means that devices have to contend with interference from many other systems such as WiFi networks, cordless phones and microwave ovens. Devices in the 2.45GHz band tend to use spread-spectrum radios, rather than narrowband systems, in order to reduce interference from other systems. However, this comes at a cost as spread-spectrum radios typically have higher power consumption due the cost of the increased signal processing required.

Current generation WSN devices, such as the T-Mote [40] and TinyNode [41], have moved to using more powerful 16-bit processors. The use of larger processors, along with increased amounts of RAM and program memory have allowed more complex applications and frameworks to be created. Developments in micro-controller design, particularly the design of the Texas Instruments MSP430 series, mean that the move to these larger processors has in fact reduced power consumption, in both low power sleep and active run modes, when compared to the previous generations of 8-bit micro-controllers [40].

Larger devices, such as the Sun Microsystems SPOT platform, utilising 32-bit processors have also been developed. The aim of these devices is to act as a micro-server, overseeing the work of several smaller nodes, within a hierarchical network architecture.

2.4.1.2 Protocols

In addition to hardware development, a great amount of work within the WSN community is focussed on the development of protocols, such as Medium Access Control (MAC) and routing protocols, to support applications. Typically, WSNs are designed to be deployed in an ad-hoc fashion, and so it is important that the protocols used support self-organisation. Furthermore, sensor networks are often characterised by dense deployments that emphasise the importance of multi-hop communication to reduce individual power consumption [10].

MAC protocols for sensor networks can be broken down into two broad categories: synchronised and unsynchronised. Unsynchronised protocols, such as B-MAC [42] and Speck-MAC [43, 44], use the concept of low power listening. In these systems, sensor nodes spend the majority of their time in a low power mode, periodically waking up to sample the radio channel to detect activity. On detecting activity the receiving node stays awake to receive the packet. Communication is achieved by transmitting, either special wake-up packets or multiple identical data packets, for a duration slightly greater than the receiving period. This ensures that all neighbouring nodes will receive the packet regardless of phase differences in local clocks.

While unsynchronised algorithms are attractive for devices that communicate rarely, and without fixed frequency, they are unsuitable for use when greater throughput is required. The requirement to use redundant retransmission, to ensure that all devices receive the transmitted packets regardless of clock drift, wastes much of the available network bandwidth. Furthermore, as access to the radio channel is contention-based, such algorithms cannot guarantee fairness.

In an attempt to optimise channel use and power consumption, alternative MAC algorithms have been developed based on time synchronisation [45]. In time-synchronised protocols, access to the radio channel is divided into time slots which are in turn allocated to individual devices. The process of allocating slots in large networks can be exceedingly complex [46].

Time synchronisation allows for better utilisation of the available network bandwidth in high contention cases, as devices do not have to contend for access, which guarantees fairness. However, in order to maintain synchronisation devices must communicate at regular intervals. The desire to have decentralised control, and to be adaptive to sudden changes in network environment, has led to the development of hybrid algorithms, such as Z-MAC [47], that combine contention-based control slots within time-multiplexed channel access schemes.

2.4.2 Tyndall National Institute

Two wireless Inertial Measurement Units (IMUs) have been constructed at the Tyndall National Institute in Cork, Ireland [48]. The first platform is comprised of stackable 25×25mm boards. Current devices include a processor board, communications board, Field Programmable Gate Array (FPGA) board, and inertial measurement board. The inertial measurement board contains a standard set of accelerometers, magnetometers and rate gyroscopes. The 25mm platform is designed to operate as a 6 Degree of Freedom (6-DoF) tracker, in addition to operating as a micro-server for additional smaller sensor units. The smaller platform consists of a pair of $10 \times 10 \times 10$mm cubes wired together; one cube containing an 8-bit Atmel ATMega128L micro-controller and 2.4GHz radio transceiver and the other cube the inertial sensors.

Both platforms use the same set of sensors: Honeywell HMC1052L magnetometers, Analog Devices ADXL202JE accelerometers and ADXRS150 rate gyroscopes. No indication is given if the gyroscopes support an extended range,[1] suggesting the device is capable of measuring a maximum rotation rate of $150^{\circ}/s$.

A Matlab based Kalman filter has been developed for performing 6-DoF orientation and position estimation based on data from the sensor devices [49]. The system has been demonstrated to track a single device in realtime although no details of latency or processing requirements have been published.

Although the system is designed for networked applications, at the time of writing, no published information was available about the performance of networking multiple devices.

2.4.3 University of Tokyo

An inertial measurement platform developed at the University of Tokyo [35] provides an example of an attempt to use a locally implemented Unscented Kalman filter (UKF) for orientation estimation in robotic tracking. The device consists of the usual sensor set of accelerometers, magnetometers and rate gyroscopes. Processing was provided by a 32-bit Hitachi H8S2633F microprocessor clocked at 16MHz. The complexity of the UKF implementation prevented update rates greater than 12Hz due to the computational cost of updating the filter state.

More recent work [37], has included the design of an IMU measuring just $24 \times 24 \times 21$mm, but this device does not include a battery nor does it support wireless networking. Additionally the device uses a 32-bit processor clocked at 50MHz for filter implementation that, while no power budget is available, likely puts it well beyond the capabilities of a standard WSN node.

[1] See Section 5.2.4.3 for further details

2.4.4 MIT Media Laboratory

A wireless IMU has been constructed at the MIT Media Laboratory for use in interactive dance performances [14]. The unit consists of 16-bit Texas Instruments MSP430 processor and Nordic Semiconductor nRF2401 2.45GHz radio transceiver in addition to Analog Devices ADXL203 accelerometers and ADXRS300 rate gyroscopes.

The lack of magnetic sensing ability prevents the MIT device from achieving fully drift-corrected orientation tracking. Instead the system is used to investigate correlation in movement between multiple interacting dancers. The system uses a Time Division Multiple Access (TMDA) network to connect multiple devices transmitting raw data to a basestation for processing by a host PC. With the 1Mbps data rate provided by the nRF2401 it is claimed that the system could support up to thirty devices updating at 100Hz each; this has not been demonstrated to date.

2.4.5 ECO

The ECO device [50] from University of California, Irvine, is currently the smallest wireless inertial measurement unit. The device measures just $12 \times 12 \times 9.5$mm including a 40mAh lithium polymer battery. However, to achieve this small size the device is extremely limited including only a Nordic Semiconductor NRF24E1 integrated 8-bit micro-controller and 2.45GHz radio transceiver and a Hitachi H48C tri-axial accelerometer. This system is clearly not capable of orientation or position tracking although it has found use in interactive performance systems.

The platform has been further developed to include multiple different modules based on the ECO design [15]. New designs include an acceleration, temperature and light sensing unit, an image sensing unit and a single axis gyroscope unit.

2.4.6 Summary of Inertial Sensing in Wireless Sensor Networks

Wireless sensor network hardware covers a large range from tiny 8-bit devices to powerful 32-bit micro-servers. However, the most common hardware is based around low-power 16-bit processors as these currently offer the greatest performance with acceptable power consumption. Many protocols have been developed to support WSN applications. However, these protocols are generally targeted at large scale ad-hoc deployments and are not ideal for the highly specific task of wireless motion capture.

Many wireless sensor network nodes include acceleration sensing as, due to the development of MEMS sensor technology, tri-axial accelerometers are now available at low cost with small dimensions. As with ECO these devices are not capable of full motion tracking on their own, instead they are used for activity and gestural recognition applications.

More fully featured inertial measurement units have been designed but in general these have not been applied to the problem of full body wireless motion capture. The processing requirements of existing orientation estimation filters require intensive processing capabilities not commonly found in low power nodes. Additionally no existing system has been demonstrated to be capable of tracking multiple nodes in a low latency networked environment as required for full body motion capture.

2.5 Summary

Commercial inertial motion tracking systems are not designed for fully wireless full body capture. Devices are often targeted at applications involving a single device, such as head tracking or for use in inertial navigation problems. As such little emphasis is given to multiple device networks. The systems that do target full body capture use wired body networks and body suits to mount devices and act as cable guides. No fully wireless system is currently available.

Wireless sensor networks provide the missing interaction between multiple sensing devices required for fully wireless motion tracking but no system has been demonstrated to perform full body drift corrected tracking. The most capable devices, the Tyndall and Tokyo IMUs, have not been demonstrated in a networked setting and other devices lack the required sensing capabilities.

Existing research into orientation filtering has concentrated on the design of filters that are optimal in terms of MSE. Although some effort has been put into the reduction of processing requirements such filters are still too computationally complex to be used in a low power device such as a wireless sensor network node.

Part II

Theory & Simulation

Chapter 3

Theory

This chapter presents the theoretical foundation for implementing postural tracking using wireless inertial/magnetic orientation sensors. First the concepts of co-ordinate frames and rotations are introduced. This is followed by a discussion of three-dimensional body modelling. Finally the problem of orientation estimation, based on inertial and magnetic data, is discussed leading to the development of a highly efficient orientation estimation filter suitable for use on a low power sensor node.

3.1 Rotations in Three Dimensional Space

3.1.1 Co-ordinate Frames and Rotations

Co-ordinate frames allow the mathematical description of points in three-dimensional space. A co-ordinate frame consists of a unique fixed point, the *origin*, and three mutually perpendicular lines passing through this point, the **X**, **Y**, and **Z** axes. The **X**, **Y**, and **Z** axes are described by unit length basis vectors. Every point in three-dimensional space can be given an unique co-ordinate tuple consisting of a linear combination of the basis vectors describing the distance from the *origin* along the **X**, **Y**, and **Z** axes. Figure 3.1 illustrates a sample co-ordinate frame. The frame is said to be right-handed as the direction of the positive **Y**-axis lies in the direction of the curled middle finger of the right hand when the index finger lies in the positive **X** direction and the thumb in the positive **Z**. Left-handed frames are also possible.

To implement orientation-only posture tracking we will require four co-ordinate frames: the world frame, the sensor frame, the joint frame, and the screen frame.

The world frame provides a reference co-ordinate frame shared between all devices in the system. For the purposes of this thesis, the world frame is a standard aerospace co-ordinate frame with positive **Z**-axis pointing down towards the centre of the Earth, positive **X**-axis pointing in the direction of magnetic North projected into the XY plane, as defined by the **Z**-axis, and positive **Y**-axis pointing East to complete a right-handed frame. The definition

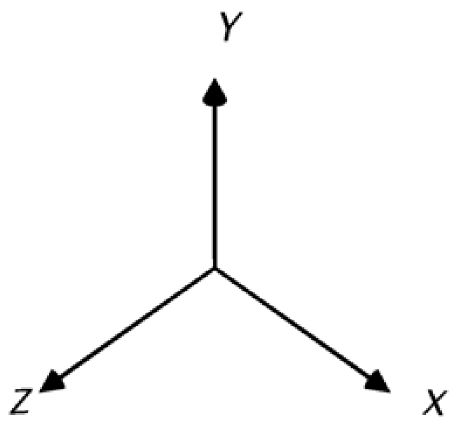


Figure 3.1: Right-handed Co-ordinate Frame

of the world frame, using magnetic North as a basis vector, causes problems in the face of magnetic field distortions. When faced with severe local distortions in the magnetic field, different devices will not share a common co-ordinate reference frame. This problem will be further discussed in Section 3.3.3.3 and Chapter 7.

The sensor frame provides a local co-ordinate frame in which the orientation sensing device measures its reference vectors. The exact choice of basis vectors is arbitrary provided that the resulting co-ordinate frame remains right-handed and the same choice is used for all devices in the system.

The joint co-ordinate frame represents the local co-ordinate frame of a particular joint in the body model. Such frames may be referenced absolutely, relative to the global world co-ordinate frame, or relatively, to the co-ordinate frame of the parent joint in the body model hierarchy.

The final co-ordinate frame, the screen frame, represents a standard computer graphics co-ordinate frame with positive **X**-axis points left, positive **Y** pointing up and positive **Z** pointing into the screen. This co-ordinate frame is used to maintain consistency with common computer graphics applications.

In order to easily represent and manipulate co-ordinate frames, and the rotations between them, it is important to have a good mathematical representation. There are three mathematical rotational representations in common usage: Euler angles, rotation matrices and quaternions. These will be discussed in turn and their advantages and disadvantages compared.

3.1.2 Euler Angles

Euler angles, named after Leonard Euler (1707-1783), provide a relatively intuitive description of rotations. Euler's theorem states [51]:

> "Any two independent orthonormal co-ordinate frames can be related by a sequence of rotations (not more than three) about co-ordinate axes, where no two successive rotations may be about the same axis."

The condition that subsequent rotations be about distinct axes yields the following twelve possible Euler angles sequences:

XYZ	**YZX**	**ZXY**
XZY	**YXZ**	**ZYX**
XYX	**YZY**	**ZXZ**
XZX	**YXY**	**ZYZ**

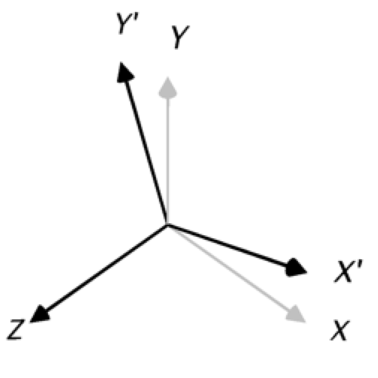
Euler angle sequences are read from left to right. A typical example, the **ZYX** or aerospace sequence, is illustrated in Figure 3.2, where the co-ordinate frame is rotated 20° about the **Z** axis, 30° about the **Y** axis and 40° about the **X** axis. First the co-ordinate frame is rotated about the **Z** axis, (Figure 3.2(a)), moving the **X** and **Y** axes to $\mathbf{X}'$ and $\mathbf{Y}'$, respectively. The second rotation is performed about the resulting $\mathbf{Y}'$ axis, (Figure 3.2(b)), and the final rotation about the resulting $\mathbf{X}''$ axis, (Figure 3.2(c)). The complete composite rotation is shown in Figure 3.2(d) with the original co-ordinate frame as a reference.

When dealing with Euler angles using the aerospace sequence, the angles are given special names: Yaw, Pitch, and Roll, corresponding to rotation about the **Z**-, **Y**- and **X**-axis respectively.

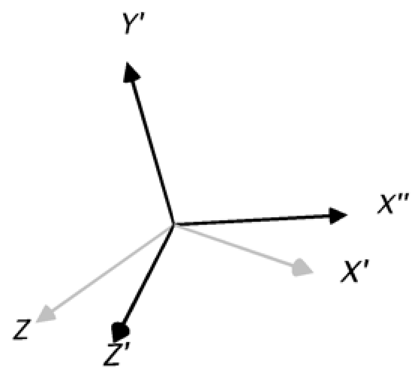
It is important when using Euler angles to always use the same rotation order, as different rotation orders result in different solutions. Figure 3.3 illustrates how using a different rotation order yields a different result. The rotation angles about each axis have the same magnitude as the previous example; only the order of rotations was altered.

Although simple to understand, Euler angles have significant disadvantages in tracking applications. Primarily there are discontinuities in the tracking of subjects as the angles wrap from 360° back to 0°. Another substantial problem occurs as the second angle of the sequence approaches $\pm 90^\circ$. In this case it is possible to lose a degree of freedom as the first and third axes of rotation become aligned. As an example, using the aerospace sequence, consider attempting to track a plane as it flies from East to West. As the plane passes through a point directly overhead, the heading angle becomes undefined switching from due East to West. Such discontinuities present problems when trying to apply filtering to Euler angles.

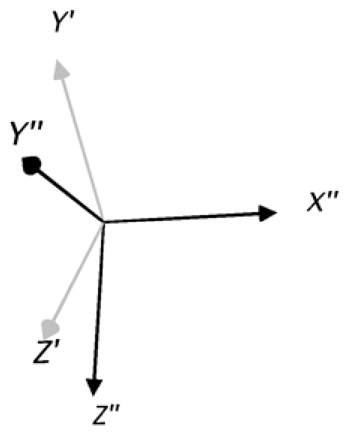
In order to manipulate rotations mathematically, Euler angles must be converted to another form such as a rotation matrix or quaternion.



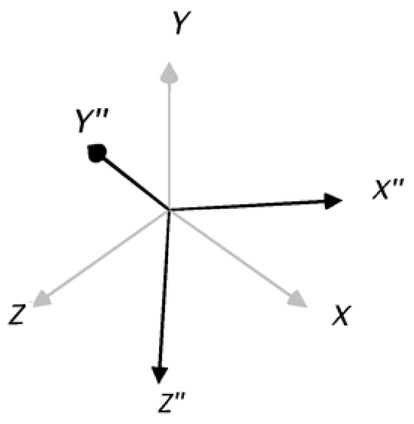
(a) Rotation about **Z**



(b) Rotation about $\mathbf{Y}'$

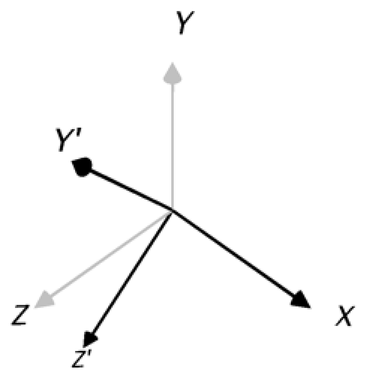


(c) Rotation about $\mathbf{X}''$

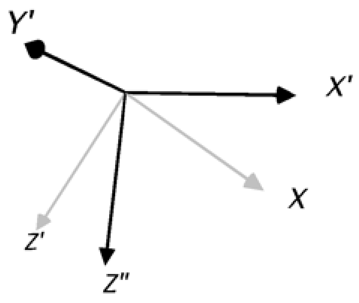


(d) Composite rotation

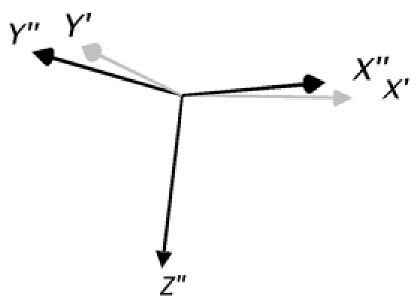
Figure 3.2: Aerospace (ZYX) Euler angle sequence



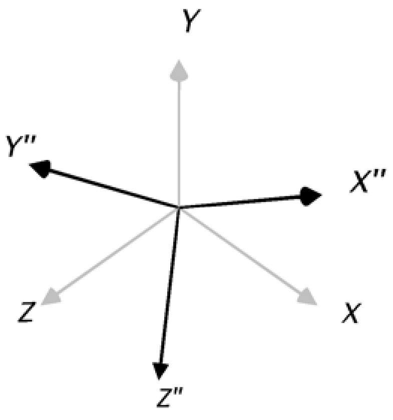
(a) Rotation about **X**



(b) Rotation about $\mathbf{Y}'$



(c) Rotation about $\mathbf{Z}''$



(d) Composite rotation

Figure 3.3: XYZ Euler angle sequence

3.1.3 Rotation Matrices

Rotation matrices are the subset of 3×3 orthogonal matrices with determinant 1. Rotation matrices are often used in computer graphics, as they can directly manipulate vectors through multiplication.

3.1.3.1 Matrix Operations

Equality Two matrices are equal if every element of matrix M_1 is equal to the corresponding element of M_2. Testing the equality of two rotation matrices therefore involves nine equality comparisons.

$$M_1 = M_2 \iff M_{1_{i,j}} = M_{2_{i,j}} \tag{3.1}$$

Addition Two matrices are added by adding each corresponding element. Adding two matrices thus involves nine scalar additions.

$$(M_1 + M_2)_{i,j} = M_{1_{i,j}} + M_{2_{i,j}} \tag{3.2}$$

The additive inverse $-M$ is given by scalar multiplication by -1.

$$-M = -1M \tag{3.3}$$

Matrix addition is commutative and associative.

Scalar multiplication Scalar multiplication of a matrix, M, by s is given by multiplying each element of M by s. Scalar multiplication thus involves nine scalar multiplies.

$$(sM)_{i,j} = sM_{i,j} \tag{3.4}$$

Scalar multiplication is commutative, distributive and associative.

Matrix multiplication Matrix multiplication is only possible if the number of columns in the first matrix is equal to the number of rows in the second matrix. Thus matrix multiplication is possible if M_1 is an $m \times n$ matrix and M_2 is an $n \times p$ matrix. Matrix multiplication of M_1 by M_2 is then given by:

$$(M_1 M_2)_{i,j} = \sum_{k=0}^{n} M_{1_{i,k}} M_{2_{k,j}} \tag{3.5}$$

Matrix multiplication is associative and distributive but not in general commutative.

$$M_1 M_2 \neq M_2 M_1 \tag{3.6}$$

The complexity of matrix multiplication, using the simple row by column algorithm, is $O\left(n^3\right)$ for $n \times n$ matrices and $O(mnp)$ in the general case of an $m \times n$ matrix multiplied by an $n \times p$ matrix. For example, matrix multiplication of two 3×3 rotation matrices requires three multiplies and two additions per element resulting in a total of twenty-one scalar multiplies and eighteen additions.

Matrix Inversion An $n \times n$ matrix is invertible if there exists a matrix M^{-1} such that:

$$M^{-1}M = MM^{-1} = I \tag{3.7}$$

The complexity of an $n \times n$ matrix inversion, using Gaussian elimination, is $O\left(n^3\right)$.

In addition to the standard algebraic operations the following operations apply specifically to rotation matrices.

Combined rotation The combined rotation of two rotation matrices is equivalent to matrix multiplication. A rotation M_1 followed by M_2 is given by:

$$M_{1\to2} = M_2M_1 \tag{3.8}$$

Inverse rotation The inverse of a rotation matrix is a special case of matrix inversion where M^{-1} is given by the matrix transpose.

$$M^{-1} = M^T \tag{3.9}$$

In this special case matrix inversion requires six swaps each using three move operations to swap the six non-diagonal elements.

Vector rotation A rotation matrix can be used to directly rotate a three dimensional vector through matrix multiplication.

$$\vec{v'} = M\vec{v} \tag{3.10}$$

Vector rotation therefore requires fifteen scalar operations.

Angular Velocity Given an angular velocity vector, $\vec{\omega} = (\theta, \phi, \psi)$, we can calculate a rotational rate matrix $\dot{M}$ [52].

$$\dot{M} = \begin{bmatrix} 0 & -\psi & \phi \\ \psi & 0 & -\theta \\ -\phi & \theta & 0 \end{bmatrix} M \tag{3.11}$$

3.1.3.2 Relationship Between Rotation Matrices and Co-ordinate Frame Basis Vectors

Rotation matrices have a special relationship to the observed position of basis vectors in rotated co-ordinate frames. First we construct our initial co-ordinate frame to have basis vectors.

$$\mathbf{X} = (1,0,0)^T \tag{3.12}$$

$$\mathbf{Y} = (0,1,0)^T \tag{3.13}$$

$$\mathbf{Z} = (0,0,1)^T \tag{3.14}$$

Now consider the effect of applying a 90° rotation around the **Z**-axis. The rotation matrix, R, for such a rotation is:

$$R = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.15}$$

Applying this matrix to each of the original basis vectors gives

$$x = R\mathbf{X} = (0,-1,0)^T \tag{3.16}$$

$$y = R\mathbf{Y} = (1,0,0)^T \tag{3.17}$$

$$z = R\mathbf{Z} = (0,0,1)^T \tag{3.18}$$

where x, y and z are the basis vectors of the original co-ordinate frame as seen in the rotated frame. It can be seen that the observed positions of the original basis vectors are exactly the columns of the rotation matrix that rotates the original co-ordinate frame into the rotated frame. From this we can see that it is possible to estimate the rotation between two co-ordinate frames if the basis vectors of one frame are observable in the other.

3.1.4 Rotation Quaternions

Quaternions are an extension to complex numbers describing a four dimensional space using a single scalar real component and three dimensional imaginary part. There are three common representations of quaternions:

1. Linear combination

$$q = w + x\vec{i} + y\vec{j} + z\vec{k} \tag{3.19}$$

   where $\vec{i}, \vec{j}, \vec{k}$ represent the standard orthonormal basis of three space.

2. Four dimensional vector

$$q = (w, x, y, z) \tag{3.20}$$

3. Tuple of real scalar and imaginary vector

$$q = (w, \vec{v}) \tag{3.21}$$

$$\vec{v} = (x, y, z) \tag{3.22}$$

Quaternions representing rotations are given by the subset of quaternions with unit length.

3.1.4.1 Quaternion Operations

Equality Two quaternions are equal if and only if their components are equal. Thus testing quaternion equality requires four equality operations.

$$q_1 = q_2 \iff w_1 = w_2, x_1 = x_2, y_1 = y_2, z_1 = z_2 \tag{3.23}$$

Addition Defined as for vector addition. Requires four scalar additions.

$$q_1 + q_2 = (w_1 + w_2, x_1 + x_2, y_1 + y_2, z_1 + z_2) \tag{3.24}$$

The additive inverse $-q$ is given by scalar multiplication by -1.

$$-q = -1q \tag{3.25}$$

Scalar multiplication Defined as for vector multiplication by a scalar. Requires four scalar multiplies.

$$sq = (sw, sx, sy, sz) \tag{3.26}$$

Quaternion multiplication Standard quaternion multiplication can be expressed in terms of vector maths as:

$$q_1 q_2 = w_1 + w_2 - \vec{v_1} \cdot \vec{v_2} + w_1 \cdot \vec{v_2} + w_2 \cdot \vec{v_1} + \vec{v_1} \times \vec{v_2} \tag{3.27}$$

Simplifying, the components can be calculated as:

$$w = w_1 w_2 - x_1 x_2 - y_1 y_2 - z_1 z_2 \tag{3.28}$$

$$x = w_1 x_2 + x_1 w_2 + y_1 z_2 - z_1 y_2 \tag{3.29}$$

$$y = w_1 y_2 - x_1 z_2 + y_1 w_2 + z_1 x_2 \tag{3.30}$$

$$z = w_1 z_2 + x_1 y_2 - y_1 x_2 + z_1 w_2 \tag{3.31}$$

Quaternion multiplication thus requires four multiplies and three additions/subtractions per element resulting in a total of sixteen scalar multiplies and twelve additions.

Quaternion multiplication is associative and distributive but not in general commutative.

$$q_1 q_2 \neq q_2 q_1 \tag{3.32}$$

The following further operations can be defined:

Combined rotation Combining rotations specified by quaternions is equivalent to quaternion multiplication.

$$q_{1 \to 2} = q_2 q_1 \tag{3.33}$$

Inverse Rotation The inverse rotation is given by the quaternion conjugate.

$$q^{-1} = q^* = (w, -\vec{v}) \tag{3.34}$$

Thus calculating an inverse rotation quaternion only requires three operations.

Vector rotation To rotate a vector, $\vec{v}$, by a quaternion q it must first be converted to a pure quaternion, v.

$$v = (0, \vec{v}) \tag{3.35}$$

The rotated vector, $\vec{v'}$, may then be found in the following manner:

$$v' = q \cdot v \cdot q^* = \left(0, \vec{v'}\right) \tag{3.36}$$

Vector rotation therefore requires two quaternion multiplies. However, by taking advantage of the knowledge that a pure quaternion has a zero real part, the number of operations can be reduced to 41.

Angular Velocity Given the angular velocity, ω, expressed as a pure quaternion, we can calculate a rate quaternion $\dot{q}$ [7].

$$\omega = (0, \theta, \phi, \psi) \tag{3.37}$$

$$\dot{q} = \frac{1}{2}\omega q \tag{3.38}$$

Re-normalise Quaternions in the rotation group must satisfy the constraint that they have unit length. Thus, re-normalising becomes:

$$norm(q) = \frac{q}{|q|} \tag{3.39}$$

3.1.5 Comparison of Rotation Representations

It can be seen from Table 3.1 that Euler angles have the lowest storage and transmission costs. However, as they cannot be easily combined mathematically and suffer from discontinuities preventing smooth tracking through all orientations they are not suitable for use in this application.

Euler Angle	Rotation Matrix	Quaternion
3	9	4

Table 3.1: Storage/Transmission Cost Comparison in Bytes

It can be seen from the previous sections that rotation matrices and quaternions provide equivalent operations. However, as shown in Table 3.2 quaternions are significantly more computationally efficient, in terms of mathematical operations such as addition, multiplication and comparison, than rotation matrices in all cases except vector rotation. Additionally they are simple to re-normalise. This property becomes especially important when performing cumulative operations in fixed-point implementations where rounding errors can quickly degrade the orthogonality of matrices.

Operation	Rotation Matrix	Quaternion
Equality	9	4
Addition	9	4
Scalar Multiplication	9	4
Multiplication	45	28
Inverse	18	3
Vector Rotation	15	41

Table 3.2: Processing Cost Comparison

3.2 Body Model

In order to describe the posture and motion of a complex body, such as a human, it is necessary to have a robust body model. A generic skeleton can be modelled as a set of rotational joints connected by bones arranged in a tree structure. Each bone is modelled as a rigid body, an idealisation of a generic solid body which allows for no deformation. Such simple rigid bodies can be described mathematically by three-dimensional vectors. Complex bodies can be built by combining multiple simple bodies with fixed rotation joints.

A basic skeleton, suitable for modelling a biped or quadruped, requires only fifteen bones as illustrated in Figure 3.4.

The fifteen bones are:

Left hand	Head	Right hand
Left upper arm	Upper spine / Shoulders	Right upper arm
Left forearm	Lower spine / Pelvis	Right forearm
Left upper leg		Right upper leg
Left lower leg		Right lower leg
Left foot		Right foot

Such a model is suitable for capturing gross movements of the limbs. Enhanced models could be easily produced by increasing the number of bones in key areas such as the spine, feet and hands.

The movement of the body model can be determined through the use of kinematics. Kinematics, unlike dynamics, ignores the forces acting on the bones and instead uses only geometrical and temporal properties such as position, velocity and acceleration. There are two possible kinematic methods that can be used for modelling body posture: forward and reverse kinematics.

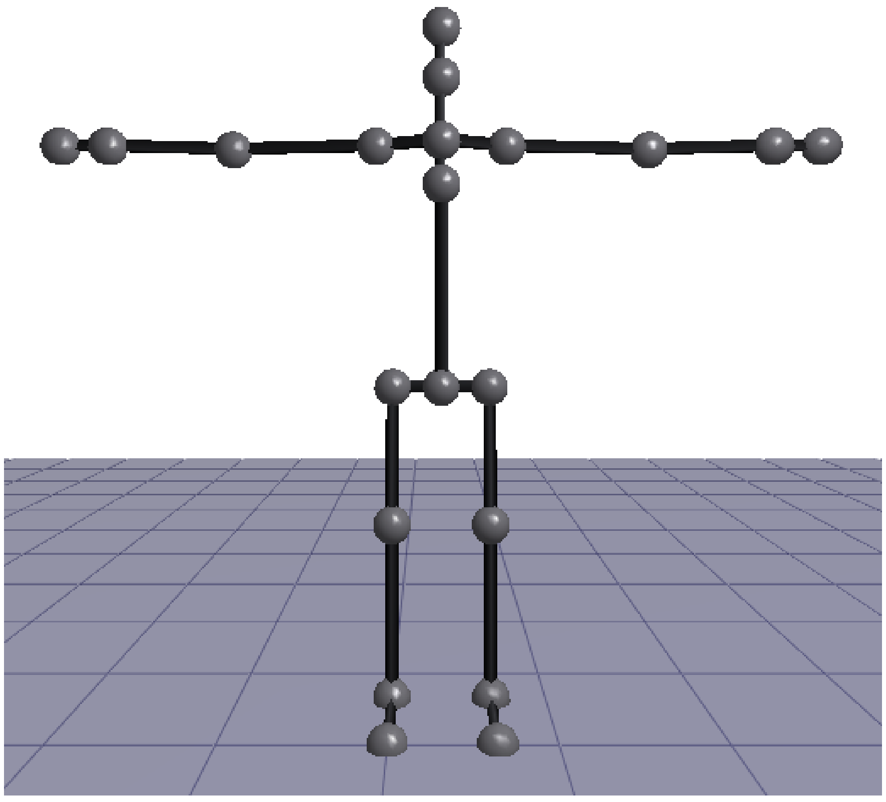
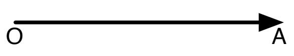


Figure 3.4: Basic Skeletal Model

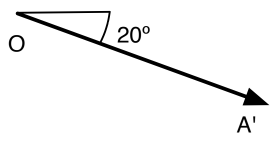
3.2.1 Forward Kinematics

In forward kinematics the position of any joint of the body can be calculated from knowledge of the bone connectivity, bone length and joint rotation.

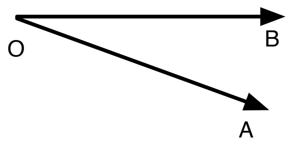
The model is represented by a tree of joints, each with a local co-ordinate frame. The position of any point in the body may be found by traversing the tree structure accumulating the orientation and position transforms as each joint is traversed. In animation the root of the joint tree is usually considered to represent the pelvis. In order to calculate the relative positions of all the bones, the tree is traversed in a pre-order depth-first manner starting from the root.



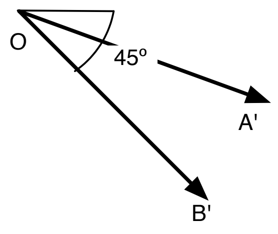
(a) First joint segment before rotation



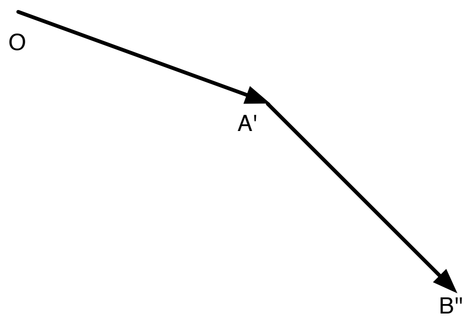
(b) First joint segment after rotation



(c) Second joint segment before rotation and translation



(d) Second joint segment after rotation



(e) Second joint after rotation and translationl

Figure 3.5: Applying Forward Kinematics

The process of updating a forward kinematic model is shown in Figure 3.5. The example shows the process of computing the end position of a simple two-joint system in two dimensions. In this example the two joint segments, A and B, are of equal length with A having a rotation of 20° from the horizontal and B having a rotation of 45°. The process of calculating the end-point begins by placing the un-rotated vector A at the origin, O (Figure 3.5(a)). The vector A is then rotated to calculate its end point, A' (Figure 3.5(b)). The same process is repeated with vector B (Figures 3.5(c) & 3.5(d)). Finally, the rotated vector B' is added to A' to yield the final position, B'' (Figure 3.5(e)). As can be seen the rotation of each joint can be specified as a rotation from a single world co-ordinate frame.

As the bones are assumed to be of fixed geometry the entire body posture can be defined by the set of joint angles. In order to drive the model it is therefore necessary to provide the rotations of all joints in the system. In the case of the simple model of Figure 3.4, this requires fifteen devices capable of tracking three degree of freedom orientation. Provided that the orientation data available to the model has sufficiently low noise, no additional constraints are required.

The static alignment error between the sensor and body segment co-ordinate frames can be calibrated by having the subject stand in a known stance. This alignment correction will also correct any static error in the sensor's orientation estimate.

3.2.2 Reverse Kinematics

An alternative approach to obtaining the body posture is to estimate the joint angles given the positions of a set of joints including, at the very least, the set of all end effectors. In the case where the set of measured joint positions is a subset of the full set of joint positions, the missing intermediate joints must be estimated. As the number of intermediate joints increases, so does the number of possible solutions. In such cases it may be necessary to employ joint angle constraints and other heuristics in order to determine the most likely stance.

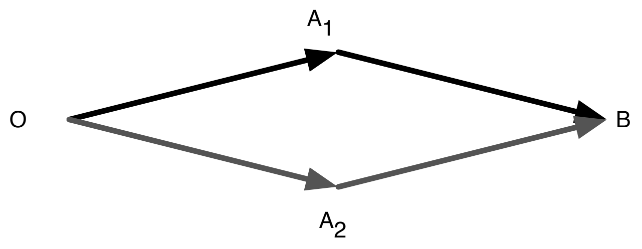


Figure 3.6: Example of Multiple Solutions in Reverse Kinematics

The problem of multiple solutions is illustrated in Figure 3.6 for a two-joint system in two dimensions. The points O and B are known, as are the lengths of the intermediate joint segments. This leaves two possible solutions for the centre point position, A_1 and A_2. Without further constraints on the model each of these solutions is equally likely, although only one will correctly reconstruct the original stance. In three dimension this problem becomes even harder as A may lie anywhere on the edge of a circle on the plain perpendicular to the line OB.

Using a reverse kinematic model presents an advantage over forward kinematics as it allows for the use of fewer sensors. Basic tracking of a human might be achieved using only six sensors. However, the sensors are now required to provide full six degree of freedom tracking. Additionally, as the model now requires joint constraints and heuristics to fill in missing information it is now limited in application to those subjects for whom the constraints are valid.

3.2.3 Summary

Reverse kinematics requires the minimum number of tracking devices. However, each device must be highly accurate in tracking its position and orientation. To estimate the positions and rotations of intermediate joints requires a complex body model, including many heuristics unique to the subject, to obtain the most likely posture estimate. This may require significant processing depending on the number of intermediate joints in the model.

Forward kinematics requires significantly less processing as the body posture is completely determined by the joint rotations. This allows the use of simpler, orientation-only, tracking devices. However, more devices are now necessary.

For this project a forward kinematic model was chosen as this allows the resulting motion capture system to be used for any articulated structure without special knowledge of its constraints. The simple model chosen may later be enhanced to provide more realistic estimation of key areas, such as the spine, through the use of interpolation or reverse kinematics. Such an approach has successfully been applied in commercial models [34].

3.3 Orientation Estimation

3.3.1 Estimation Problem

The purpose of the orientation estimation filter is to combine sensed data to produce an estimate of the rotation between the world co-ordinate frame and the sensor co-ordinate frame. It is assumed that the subject being tracked is generally in direct contact with the surface of the Earth.

The data available to the filter comprises three measured vectors from the three sensor sets:

Accelerometers measure the static and dynamic acceleration in the body frame. The measured acceleration vector, $\vec{A}$, is composed of: $\vec{a}$, the dynamic acceleration, $\vec{g}$, the static acceleration due to gravity, the accelerometer offset, $\vec{O}_A$, and $\vec{n}_A$ a measurement noise component. The matrix S_A provides for normalisation of the differing scaling factors associated with each sensed axis.

$$\vec{A} = S_A \left(\vec{g} + \vec{a} + \vec{O}_A + \vec{n}_A\right) \tag{3.40}$$

Magnetometers measure the magnetic field in the body frame. The measured magnetic vector, $\vec{M}$, is composed of the magnetic field, $\vec{m}$, the magnetometer offset, $\vec{O}_M$, and a measurement noise component $\vec{n}_M$. As with the accelerometers the S_M matrix provides for scaling factor normalisation.

$$\vec{M} = S_M \left(\vec{m} + \vec{O}_M + \vec{n}_M\right) \tag{3.41}$$

Rate Gyroscopes measure the rotational rate in the body frame. The measured gyroscope vector, $\vec{G}$ can be considered to be composed of three components: $\vec{\omega}$, the rotational rate,

$\vec{O}_G$, the gyroscope bias offset and $\vec{n}_G$ a measurement noise component. The S_G matrix is again for scaling factor normalisation.

$$\vec{G} = S_G\left(\vec{\omega} + \vec{O}_G + \vec{n}_G\right) \tag{3.42}$$

In each of the above measurement functions the scaling and offset factors are subject to variation with time and temperature.

3.3.2 Requirements for Orientation Estimation Filter

To implement an orientation filter suitable for use in a BSN it must not only be capable of estimating orientation with low error and latency, but must also be implemented with the limited resources available to a BSN node. Typically such nodes will be limited to low power 16-bit processors with small amounts of RAM and, in order to build compact and unobtrusive devices, have limited energy storage capacity so every effort should be made to limit the processing requirements. Additionally, as radio communication typically is much more expensive than processing [10], the filter should reduce the requirement for data transmission. For these reasons this thesis will take the approach of finding the simplest filter capable of meeting the following key requirements:

Low Error The orientation error should be minimised where possible.

Low Latency The orientation update should be produced without significant processing or group delays. Processing delay is related to the complexity of the filter implementation. Group delay is the delay between a sample being acquired and its maximum effect on the output of the filter. This property is related to the transfer function of the filter.

Low Noise The filter should minimise high frequency noise that would produce jitter in the orientation estimate.

Reduced Bandwidth The filter output should have a fixed bandwidth and allow subsampling without introducing aliasing effects. Human motion capture, for everyday activities such as walking and gesturing, does not require high frequency information. Studies of acceleration power distribution show that the majority of typical acceleration lies within the range of 0-18Hz [53].

Compact Output The filter should produce an output in a compact representation for transmission.

3.3.3 Naive Estimation Strategies

Two simple methods exist to perform orientation estimation given the available sensor data of rotation rate, linear acceleration and magnetic field vector.

3.3.3.1 Rate Gyroscope Integration

The simplest approach to orientation estimation would be to integrate directly the rate gyroscope data to produce an estimate of accumulated rotation from a known initial orientation. This may be easily expressed in quaternion form given the initial orientation, q, a pure quaternion giving the angular velocity in radians per second in the sensor co-ordinate frame, ω, and the time step between updates, Δt. Recall that from Equation 3.38 we can calculate a rate quaternion, $\dot{q}$, as:

$$\dot{q} = \frac{1}{2}\omega q \tag{3.43}$$

The rate quaternion for one time step is therefore:

$$\dot{q}_t = \frac{1}{2}\omega q_t \Delta t \tag{3.44}$$

The estimated orientation quaternion after one integration time step, q_{t+1}, can now be calculated as:

$$q_{t+1} = \dot{q}_t + q_t \tag{3.45}$$

Finally, in order to reduce the effect of rounding errors in fixed precision implementations, the resulting quaternion should be normalised to produce the base estimate for the subsequent time step:

$$q_{t+1} = \frac{\dot{q}_t + q_t}{|\dot{q}_t + q_t|} \tag{3.46}$$

This estimation method has the advantage that there is very little latency in producing the estimated orientation. The latency is simply the processing time to integrate the latest sample. However, this method is naive, as it ignores rate gyroscope bias and scaling errors that causes the estimation error to change with time. Recall from Section 3.3.1 that the rate gyroscope measurement function was:

$$\vec{G} = S_G\left(\vec{\omega} + \vec{O}_G + \vec{n}_G\right) \tag{3.47}$$

We can consider the effects of each term individually.

An error in the scaling factor, S_G, will result in the gyroscope appearing to under- or over-estimate the rotational rate. This will in turn cause the orientation estimate to under- or over-estimate rotation. As an example, consider the case when the scaling value is increased by a factor of two. In this case the angular rate will be double the true value resulting in the device appearing to rotate at twice its true speed. This will cause the orientation estimate to alter by two degrees for every degree of rotation.

A non-zero offset bias error, $\vec{O}_G$, in the rate gyroscope reading will cause the device to appear to rotate at a constant angular velocity even while stationary.

The effect of the Earth's own rotational rate can be discounted as it so low that it will effectively undetectable by the gyroscopes. The Earth's rotational rate is approximately:

$$\omega_{Earth} \approx \frac{360}{24 \times 60 \times 60} = 0.004^{\circ}/\mathrm{s}, \tag{3.48}$$

equivalent to less than 0.002% of the gyroscope range at maximum sensitivity.

The presence of noise, $\vec{n}_G$, in the gyroscope measurement has a less troublesome effect on the accuracy of the estimation process. Assuming that the noise has a mean value of zero, it will not cause any persistent error in the estimated orientation. The input noise will, however, produce noise in the estimated output. The magnitude and frequency characteristics of the output noise will be directly related to those of the input noise.

3.3.3.2 Vector Observation

An alternative approach to rate gyroscope integration would be to use the position of the observed acceleration and magnetic vectors. Recall that the world frame was defined by a positive **Z** vector pointing towards the centre of the Earth, and an **X** vector pointing in the direction of magnetic north projected into the XY plane. It is therefore possible for the device to directly measure the basis vectors of the world co-ordinate frame and thus calculate its orientation.

From Section 3.3.1, the accelerometer measurement function was:

$$\vec{A} = S_A \left(\vec{g} + \vec{a} + \vec{O}_A + \vec{n}_A \right) \tag{3.49}$$

Thus, while at rest, or travelling at constant velocity, the accelerometers will measure only the acceleration due to gravity, $\vec{g}$, plus the inevitable measurement noise, $\vec{n}_A$. Due to the design of the accelerometer,[1] $\vec{g}$ points in the exact opposite direction of the true gravity vector. Thus by inverting $\vec{g}$, and normalising the resultant vector, the device can directly measure the world co-ordinate frame's **Z** basis vector in the device's local co-ordinate frame.

$$\mathbf{Z} = -\frac{\vec{g}}{|\vec{g}|} \tag{3.50}$$

Having found the world frame's **Z** basis vector it is possible to estimate the position of the world's **X** basis vector by projecting the measured magnetic field vector into the **XY** plane defined by the **Z** vector. This is achieved by decomposing the measured magnetic vector into its vertical and horizontal components. The vertical component of the magnetic field, $\vec{m}_V$, can be found using the vector dot product:

$$\vec{m}_V = \vec{m} \cdot \vec{g} \tag{3.51}$$

[1] An accelerometer held in a fixed position will measure static acceleration as the difference from free-fall thus appearing to accelerate upwards rather than downwards.

The horizontal component, $\vec{m}_H$, is then found by subtraction:

$$\vec{m}_H = \vec{m} - \vec{m}_V \tag{3.52}$$

Finally the horizontal component is normalised to yield the world co-ordinate frame's **X** vector:

$$\mathbf{X} = \frac{\vec{m}_H}{|\vec{m}_H|} \tag{3.53}$$

The final basis vector of the world co-ordinate frame can be computed by a vector cross product:

$$\mathbf{Y} = \mathbf{Z} \times \mathbf{X} \tag{3.54}$$

Recall from Section 3.1.3.2 that, by measuring the basis vectors of the world reference frame, we can directly reconstruct the rotation matrix between the world and sensor co-ordinate.

The resulting rotation matrix can be converted to a positive real quaternion by equating the terms of the calculated rotation matrix with the matrix given in Equation 3.55 [51].

$$\begin{bmatrix} 2q_w^2 - 1 + 2q_x^2 & 2q_xq_y + 2q_wq_z & 2q_xq_z - 2q_wq_y \\ 2q_xq_y - 2q_wq_z & 2q_w^2 - 1 + 2q_y^2 & 2q_yq_z + 2q_wq_x \\ 2q_xq_z + 2q_wq_y & 2q_yq_z - 2q_wq_x & 2q_w^2 - 1 + 2q_z^2 \end{bmatrix} \tag{3.55}$$

The above Orient vector observation algorithm presented is stable provided that the measured acceleration and magnetic field vectors are non-zero and do not become aligned. For example, if alignment occurs, the **X** and **Y** basis vectors become all zeros resulting in a non-orthogonal matrix. Such a matrix cannot be considered as a rotation.

An alternative algorithm that bypasses the generation of an intermediate rotation matrix is presented by Xiaoping et al. [54]. The resulting Factored Quaternion Algorithm (FQA) works in a similar manner to the stated vector algorithm, by discarding the vertical component of the magnetic field.

The previous approaches to estimating the rotation between two co-ordinate frames based on vector observation do not yield an optimal solution. As the magnetic field vector is projected into the horizontal plane, defined by the acceleration vector, the information in its vertical component is discarded.

Optimal solutions to the vector observation problem seek to find the rotation matrix, R, that minimises Wahba's loss function [55]:

$$L(R) = \frac{1}{2}\sum_{i=1}^{n} a_i \left|\vec{b}_i - R\vec{w}_i\right|^2 \tag{3.56}$$

where $\vec{b}_i$ are the observed vectors in the local co-ordinate frame, $\vec{w}_i$ are the reference vectors in the world frame, and a_i are a set of non-negative weights applied to each vector to account for their individual accuracy. In general with noise present in the measurement of the vectors it is impossible to minimise the loss function to zero. Numerous solutions to minimising (3.56) exist [56] including the commonly cited QUEST algorithm [57].

The vector observation methods suffer from errors in the measured acceleration vector due to dynamic acceleration. From Equation 3.49 we can see that in addition to the static acceleration due to gravity the accelerometers also measure the dynamic acceleration caused by device movement. This additional acceleration will cause the apparent position of the world's **Z** vector to move. This in turn corrupts the positions of the other observed basis vectors leading to an incorrect orientation estimate.

Due to the influence of dynamic accelerations on the estimated orientation, the results of a vector observation are only valid when the device is in a static orientation or moving at constant linear velocity. For very low rotational rates the measured acceleration vector may be low-pass filtered to remove the majority of the dynamic acceleration. By choosing a filter cut-off such that the frequencies up to the highest frequency of rotation are passed, but all higher frequencies are attenuated, a rough approximation can be made.

An early prototype was developed using this method. It was discovered, unsurprisingly, that in order to filter out dynamic linear accelerations it was necessary to use aggressive low-pass filtering of the accelerometer data. This low-pass filtering introduced significant delay and loss of high frequency information in the estimation process.

3.3.3.3 Effect of Field Variations on Vector Observations

The use of vector observation methods for orientation estimation assumes that both the magnetic and gravitational fields form parallel vector fields within the capture volume.

The use of optimal algorithms that minimise Equation 3.56 present a problem in practical use. The inclination angle, and therefore the vertical component, of the Earth's magnetic field varies considerably with location as shown in Figure 3.7. Unless variation in the inclination angle is accounted for, in the definition of the reference magnetic field vector, an optimal vector observation algorithm will generate an inaccurate orientation. The resulting pitch and roll errors occur as the algorithm must compromise between the vertical information contained in each vector. As non-optimal algorithms discard this vertical information from the magnetic field they can be used without alteration almost anywhere on the Earth's surface. The only exception is in the vicinity of the Earth's magnetic poles where the inclination angle approaches $^{\circ}90$, the magnetic field and gravity vectors become aligned, and the resulting orientation becomes undefined.

The large scale effect of magnetic field declination, the angle between magnetic North and grid North, is less important. While the large scale variation in declination angle is at least as great as in inclination angle, as shown in Figure 3.8, the resulting error can be easily be corrected or ignored. As the horizontal component of the magnetic field is by definition

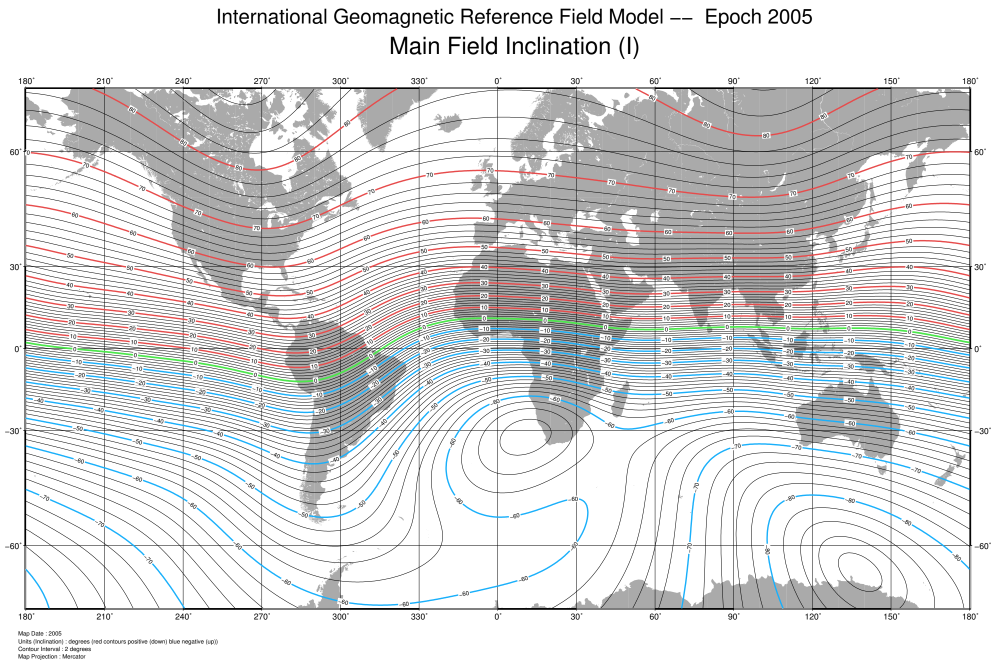

International Geomagnetic Reference Field Model –– Epoch 2005
Main Field Inclination (I)
Map Date : 2005
Units (Inclination) : degrees (red contours positive (down) blue negative (up))
Contour Interval : 2 degrees
Map Projection : Mercator

Figure 3.7: Map of Magnetic Field Inclination Angle Variation [1]

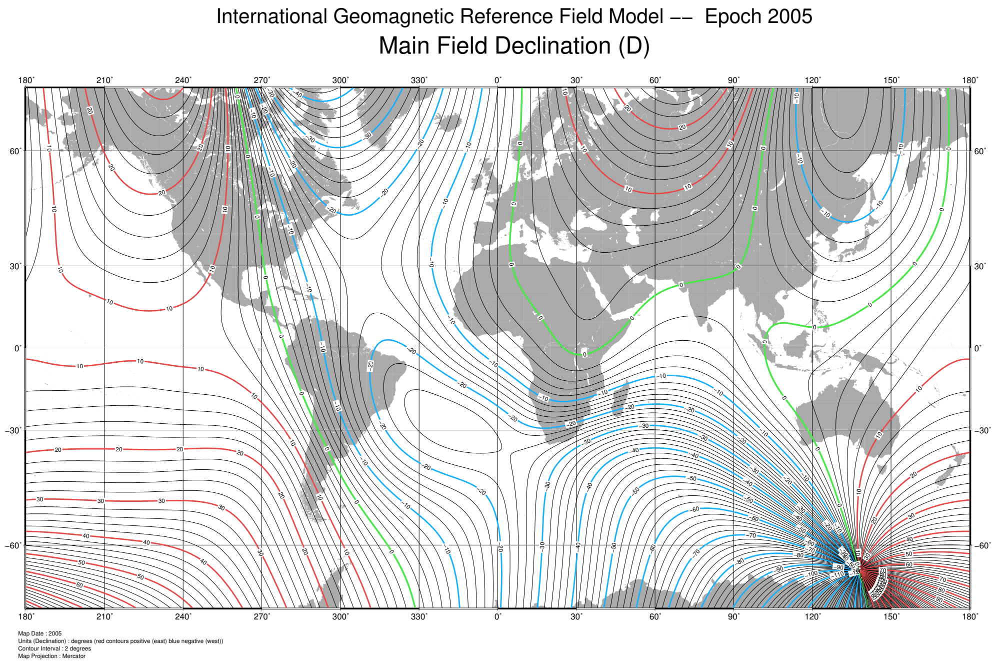

International Geomagnetic Reference Field Model –– Epoch 2005
Main Field Declination (D)
Map Date : 2005
Units (Declination) : degrees (red contours positive (east) blue negative (west))
Contour Interval : 2 degrees
Map Projection : Mercator

Figure 3.8: Map of Magnetic Field Declination Angle Variation [1]

orthogonal to the local gravity vector it only affects the resultant heading angle. Provided that the local variation in declination within the capture volume is small, the local co-ordinate frame, defined by the magnetic North and down vector, can be easily converted to a global frame with a single heading rotation to re-align magnetic North and grid North. It is small scale distortions in the magnetic field, caused by ferrous objects in the capture environment, that will cause the greatest error.

The gravitational field of the earth varies depending on the latitude, height above sea-level, the tidal pull of the moon and the local terrain [58].

Gravitational variation with latitudes account for the greatest variation in field strength, varying from $9.780m/s^2$ at the equator to $9.832m/s^2$ at the poles, a change of approximately 0.5%. This variation is due to the oblate nature of the Earth's surface and the centrifugal force required to keep an object on the surface of the Earth as it rotates. The magnitude of the local gravitational field can by calculated using the International Gravity Formula, Equation 3.57 [59], where g is the gravitational field strength in m/s^2, and λ is the latitude in degrees.

$$g = 9.780495(1 + 0.0053024\sin\lambda^2 - 0.0000073\sin 2\lambda^2) \tag{3.57}$$

The effect of altitude variations can be determined from application of Newton's law of gravitation, Equation 3.58, and the second law of motion, Equation 3.59, where F is the gravitational force on an object of mass m caused by another object of mass m_2 at a distance r, and G is the universal gravitational constant, a is acceleration and g is the acceleration due to gravity.

$$F = G\frac{mm_2}{r^2} \tag{3.58}$$

$$F = ma \tag{3.59}$$

$$\Rightarrow g \equiv a = G\frac{m_2}{r^2} \tag{3.60}$$

The ratio of the gravitational force is the ratio between the gravitational force at sea-level, g_0, and the gravitational force at a height, g_h.

$$g_0 = G\frac{m_2}{r^2} \tag{3.61}$$

$$g_h = G\frac{m_2}{(r+h)^2} \tag{3.62}$$

$$\frac{g_0}{g_h} = \frac{r^2 + 2rh + h^2}{r^2} \tag{3.63}$$

$$= 1 + 2\frac{h}{r} + \frac{h^2}{r^2} \tag{3.64}$$

$$\Rightarrow g_h = \frac{g_0}{1 + 2\frac{h}{r} + \frac{h^2}{r^2}} \tag{3.65}$$

From Equation 3.61 we can estimate that the gravitational field strength at the top of the highest

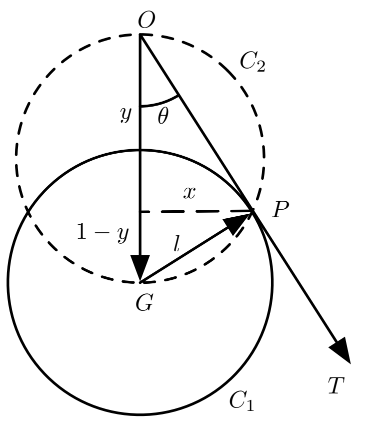


Figure 3.9: Geometry for Calculating Worst Case Acceleration Error

mountain on Earth is

$$\frac{g_0}{1+2\frac{8850}{6.371\times10^6}+\frac{8850^2}{(6.371\times10^6)^2}} \approx 0.997g_0 \tag{3.66}$$

taking the height of Everest at 8850m and the mean radius of the Earth as 6.371×10^6m. This represents a change of approximately 0.3%. A further reduction in experienced field strength is due to the increase in required centrifugal force, however this change is negligible and can be ignored [58].

The variations in gravitational field strength due to latitude and altitude effect only the magnitude of the local gravity vector, not its direction which always points towards to the centre of mass of the Earth. The tidal effect of the moon and sun and the effect of the local terrain cause variations both in the magnitude and the direction of the experienced gravity vector.

Any component of acceleration perpendicular to the gravitational vector pointing towards the centre of the Earth will result in an error between the measured gravity vector and the true Earth normal. The magnitude of this error can be calculated based on the geometry of Figure 3.9. The figure illustrates the effect of a small acceleration, $\vec{l}$ with magnitude l, combined with the main gravity vector, $\vec{g}$. The worst case error angle θ_{max} is the angle between the line OG, representing the normal direction of gravity, and OT, the tangent to the circle, C_1, representing the locus of $\vec{g}+\vec{l}$. From the diagram:

$$\theta_{max} = \tan^{-1}\left(\frac{x}{y}\right). \tag{3.67}$$

The point of intersection of C_1 and OT is the intersection of the two circles C_1 and C_2, given by Equations 3.68a and 3.68b respectively.

$$x^2+(y-1)^2-l^2=0 \tag{3.68a}$$

$$x^2+\left(y-\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2=0 \tag{3.68b}$$

Combining (3.68a) and (3.68b) and solving for y:

$$y=1-l^2. \tag{3.69}$$

By Pythagoras:

$$x=\sqrt{l^2-(1-y)^2}=\sqrt{l^2-l^4} \tag{3.70}$$

$$\therefore \theta_{max}=\begin{cases} \tan^{-1}\left(\frac{\sqrt{l^2-l^4}}{1-l^2}\right), & 0\leq l<1 \\ \lim_{l\to 1}\left(\tan^{-1}\left(\frac{\sqrt{l^2-l^4}}{1-l^2}\right)\right)=\frac{\pi}{2}, & l=1 \\ \pi, & l>1 \end{cases} \tag{3.71}$$

The average gravitational effect of the moon on an object can be again calculated from Newton's laws, taken the mass of the moon to be 7.36×10^{22}kg and the average distance from the Earth to the moon as 385×10^6m.

$$a=G\frac{m_2}{r^2} \tag{3.72}$$

$$=6.673\times10^{-11}\frac{7.36\times10^{22}}{(385\times10^6-6.371\times10^6)^2} \tag{3.73}$$

$$=3.43\times10^{-5}\text{m/s}^2\approx3.5\times10^{-6}g \tag{3.74}$$

The gravitational effect of the moon is therefore approximately 0.0003% of the Earth's gravitational effect. The error due to the gravitational effect of the moon can be calculated using Equation 3.71 as 0.0002°.

Equation 3.71 cannot be simply solved for the magnitude of l required to cause an error θ. However, for small values of θ, less than 1°, (3.71) can be simplified to:

$$\theta\approx\tan^{-1}\left(\frac{l}{1}\right), \tag{3.75}$$

with virtually no loss of accuracy. For example, to introduce an error of 0.1° would require an acceleration of

$$l=\tan(0.1^\circ)=0.0017g\approx0.017\text{m/s}^2. \tag{3.76}$$

To create this acceleration at a one metre radius would require a spherical mass of:

$$m=ar^2/G=0.017\times1^2/6.673\times10^{-11}=255\times10^6\text{kg}. \tag{3.77}$$

To give this some perspective, the required mass is equivalent to a sphere of lead with a diameter of approximately 35m. From this we can see that the mass required to influence the measurement of the local Earth normal vector are of such a large scale that they can be safely ignored. Indeed, measurements of the effect of the Scottish mountain Schiehallion on the local gravity vector to the North and South of the mountain, conducted by the Rev. Nevil Maskelyne in 1774, resulted in a deviation of only 11.6 arc-seconds or 0.0032° between the two measurement stations [60]. Errors due to variations in gravity due to nearby masses are henceforth ignored.

3.3.3.4 Effect of a Non-Stationary Subject Reference Frame

The system is not designed to be able to track the motion of a subject in a co-ordinate frame moving relative to the surface of the Earth. However, it is worth briefly addressing this issue as it is a common occurrence. For example, it may be interesting the examine the motion of a subject onboard some form of transport.

In the case of reference frame undergoing a linear translational motion, such as a car travelling along a straight road, every point in the frame will experience common linear accelerations relative to the Earth-fixed frame. The effect of this acceleration will therefore be the same for all sensors resulting in a common error in orientation estimation as the gravitational and linear accelerations combine. The relative posture of the subject will therefore be accurate, although the absolute orientation will be in error.

A more complicated scenario is that of a car following a winding road. In this case the accelerations experienced by each point in the moving reference frame vary depending on the radius from the current centre of curvature. The linear acceleration experienced by each point in the moving reference can be calculated based on the angular velocity and acceleration of the reference frame relative to the Earth-fixed frame, and the offset from the centre of curvature. The variation in linear acceleration experienced by different sensors on the subject's body will result in a variation in orientation error. The formula for the calculation of linear acceleration will be given in Chapter 4 as Equation 4.17.

In summary, given perfect knowledge of the motion of a moving subject reference frame, it would be theoretically possible to compensate for the resulting linear accelerations. However, this thesis will not attempt to do so. Finally, it should be noted that the distortion of the magnetic field by most forms of transport would prevent use of the system as it stands.

3.3.4 Kalman Filter

The Kalman filter is a common filter design used in control theory. It can be shown to be the optimal recursive filter in the sense that it minimises the estimated error covariance [61]. The Kalman filter was originally developed for use in inertial guidance of rockets.

The standard Kalman Filter is defined as follows. Consider a linear system, with an $n \times 1$ state vector, $\vec{x}$, that obeys the following discrete time evolution:

$$\vec{x}_{k+1} = A\vec{x}_k + B\vec{u}_k + \vec{w}_k \tag{3.78}$$

where, A is an $n \times n$ state transition matrix representing the system dynamics, u is a $p \times 1$ vector of known system control inputs, B is an $n \times p$ matrix relating control inputs to system state, and $\vec{w}$ is an $n \times 1$ process noise vector, with covariance matrix Q_k, representing uncertainty in the system dynamics. Now suppose that there are m indirect measurements of the state vector:

$$\vec{y}_k = C\vec{x}_k + \vec{v}_k \tag{3.79}$$

where, C is an $m \times n$ observation matrix relating system state to observed measurements and $\vec{v}$ is a measurement noise vector with covariance R_k.

The Kalman filter can now be defined [62] as:

$$K_k = AP_kC^T \left(CP_kC^T + R_k\right)^{-1} \tag{3.80}$$

$$\hat{\vec{x}}_{k+1} = \left(A\hat{\vec{x}}_k + B\vec{u}_k\right) + K_k \left(\vec{y}_{k+1} - C\hat{\vec{x}}_k\right) \tag{3.81}$$

$$P_{k+1} = AP_kA^T + Q_k + AP_kC^T R_k^{-1} CP_kA^T \tag{3.82}$$

where, K is the Kalman gain matrix and P is the estimation error covariance matrix.

The Kalman filter is a predictor-corrector filter, where the first term in Equation 3.81 is the prediction, and the second term is the correction. An intuitive understanding of the filter behaviour can be formed by examining the limits of the Kalman gain given by Equation 3.80.

$$\lim_{R_k \to 0} K_k = AC^{-1} \tag{3.83}$$

$$\lim_{P_k \to 0} K_k = 0 \tag{3.84}$$

It follows that as the measurement noise covariance, R_k, is reduced, greater weight is given to the correction. Similarly, as the estimated error covariance, P_k, is reduced more weight is given to the prediction. The estimated error covariance is calculated at each update according to Equation 3.82.

3.3.4.1 Criticism of the Kalman Filter

Direct Kalman filters, as illustrated in Figure 3.10, which include estimates of the orientation and angular velocities in the state vector, are common in Attitude and Heading Reference System (AHRS) designs. However, these filters are computationally complex due to the large number of interrelated components in the state vector. Attempts to implement such filters on embedded platforms have suffered from very poor update rates due to the cost of filter implementation [35].

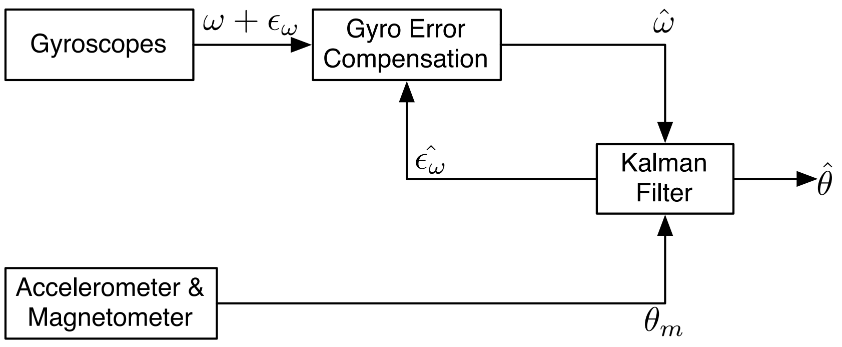


Figure 3.10: Direct Kalman Filter

An attractive property of the Kalman filter for AHRS is that the predictive nature of the filter allows the apparent system latency to be reduced by predicting the state into the future. In realtime wireless motion tracking, the ability to predict the orientation state into the future would allow for significant reduction in latency between subject movement and corresponding body model update by covering delays related to low bandwidth network latency. However, accurate prediction is only possible with a highly accurate dynamics model for the subject being tracked.

Typical Kalman filter implementations show their heritage in aerospace applications by assuming a dynamic model where angular velocity is preserved. This model is clearly inaccurate for human movement. Consider the motion of a walking human. The motion of the legs has distinct phases during which different motion characteristics are displayed [18]. Attempting to accurately model these motions would add further complexity to the already complex filter structure. Additionally, as different parts of the body have different characteristic dynamics the filter would need to be optimised for each body part making the system less general.

The standard Kalman filter only works with linear systems with white process and measurement noise. The extended Kalman filter relaxes the condition on linearity by linearising around the current estimate [61]. Several further modifications have been suggested that may reduce estimation error by improved handling of non-linear systems. However, these extensions significantly increase processing requirements [37].

3.3.4.2 Complementary Kalman Filter

An alternative to the direct Kalman filter, proposed for use in head tracking applications by Foxlin [63], is the complementary Kalman filter illustrated in Figure 3.11. In this version the Kalman filter is not used to directly track the orientation. Instead it is used to estimate the gyroscope bias and orientation error which are fed into the orientation calculation. As such it preserves the fast response of pure gyroscope tracking discussed in Section 3.3.3.1 while providing bias estimation and orientation correction. As the Kalman filter in this design does

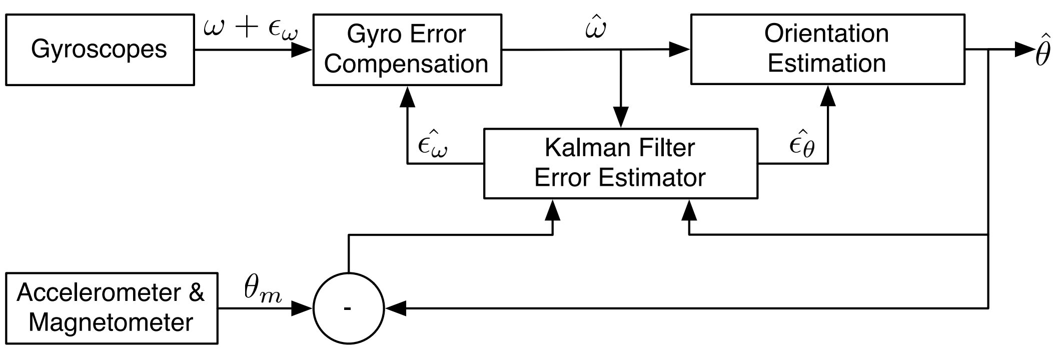


Figure 3.11: Complementary Kalman Filter

not directly track orientation, it reduces the dimensions of the filter state vector, and significantly reduces the processing requirements. However, this implementation does not allow the predictive nature of the Kalman filter to cover system latency.

The complementary Kalman filter proposed by Foxlin is based on an Euler angle representation of the device orientation. As such it suffers from discontinuities when the pitch angle of the device approaches $\pm 90^{\circ}$, or when the heading or roll angles approach $\pm 180^{\circ}$. While this was not considered a major problem for head-tracking applications it is liable to cause significant problems when applied to full body motion capture as the potential range of motion is much greater.

3.3.5 Complementary Filter

Kalman filtering relies upon knowledge of the characteristics of the process dynamics, process noise and measurement noise to produce optimal results. The Kalman gain is continually updated to optimally weight the prediction and correction terms of the filter. This requires significant processing resources even in the case of the complementary Kalman filter. An alternative technique is to forgo the bias error estimation and rely solely on error minimisation through complementary filtering.

The complementary approach is taken by Bachmann [7] resulting in the filter illustrated in Figure 3.12. This filter operates by direct integration of rate gyroscope data combined with an error correction term derived from comparing the estimated gravity and magnetic vectors to the observed vectors.

The error estimation is performed by comparing the measured gravity and magnetic field vectors with the predicted vectors based on the estimated orientation. This requires that the position of the reference vectors in the world frame be well known. While this is obviously true for the gravity vector, it is not true of the Earth's magnetic field vector, the vertical component of which can vary considerably with location [64].

The comparison process produces a six-component error vector. This is then converted

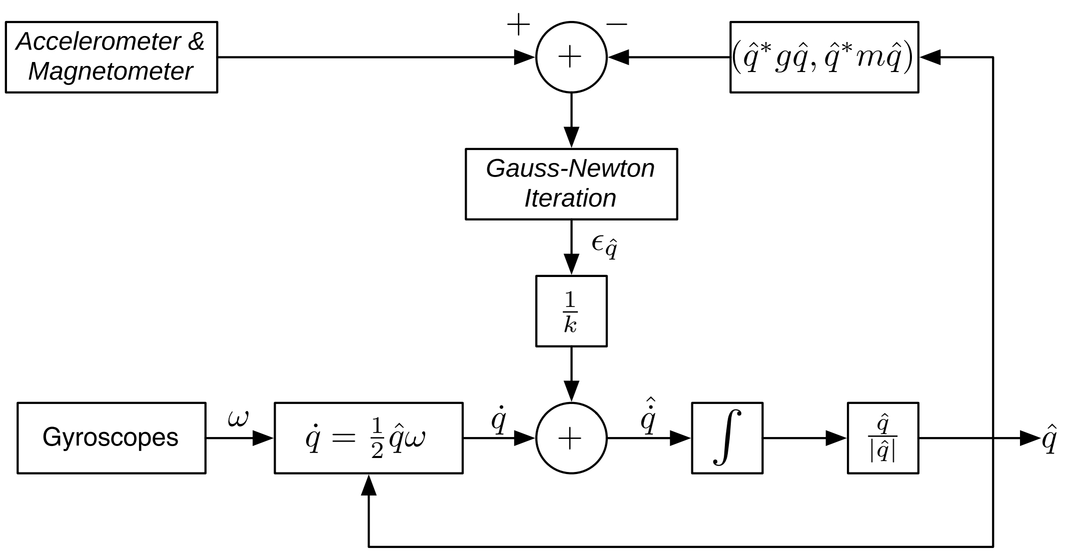

Accelerometer & Magnetometer
+
−
$(\hat{q}^* g\hat{q}, \hat{q}^* m\hat{q})$
Gauss-Newton Iteration
$\epsilon_{\hat{q}}$
$\frac{1}{k}$
Gyroscopes
ω
$\dot{q} = \frac{1}{2}\hat{q}\omega$
$\dot{q}$
$\hat{\dot{q}}$
$\int$
$\frac{\hat{q}}{|\hat{q}|}$
$\hat{q}$

Figure 3.12: Bachmann Complementary Quaternion Filter

to an error quaternion through the use of Gauss-Newton iteration to find the quaternion that best minimises the error vector. This process allows different weightings to be applied to the accelerometer and magnetometer data based on empirical knowledge of accuracy.

The error quaternion produced by the Gauss-Newton process is scaled by a filter co-efficient, k, to introduce a low-pass filtering effect on the drift correction minimising high frequency noise from the accelerometers.

The use of Gauss-Newton iteration involves multiple iterations per sample in order to converge on the optimal solution. In early work this involved calculating the inverse of a 4×4 matrix at each iteration. Subsequently, this was simplified to calculating the inverse of a 3×3 matrix. In order to reduce processing requirements, the drift correction was not calculated for every input sample.

In later work Bachmann returns to the concept of the complementary Kalman filter similar to that of Foxlin. The Gauss-Newton iteration used to calculate the quaternion error term is replaced by an implementation of the QUEST algorithm [38]. The QUEST implementation requires the calculation of the inverse of a 4×4 matrix. However, unlike the Gauss-Newton method, this solution is non-iterative, only needing to be calculated once per sample.

3.3.5.1 Criticism of the Bachmann Filter

As seen in Section 3.1.3.1, calculating the inverse of a general matrix is an expensive operation. The computational cost is further increased as the elements of the inverse matrix cannot be guaranteed to lie within the range of standard 16-bit fixed-point formats. This requires that matrix inverses are computed in floating-point maths, greatly increasing the processing time and cost on embedded processors.

A second flaw has been identified in the Bachmann filter structure related to gyroscope bias passthrough. Any small offset in the gyroscope input will cause an offset error in the output. The correction term derived from the accelerometers and magnetometers will limit this error but cannot entirely eliminate it. This effect limits the ability of the Bachmann filter to estimate static orientations accurately. The details of this flaw will be further discussed in Chapter 4.

Finally, the Bachmann filter performs drift correction regardless of dynamic linear accelerations. As the accelerometers measure the combination of dynamic and static acceleration the resulting orientation estimate may be significantly corrupted. No attempt is made to minimise these errors.

3.3.6 Direct Conversion Complementary Quaternion Filter

It is proposed in this thesis that the Bachmann complementary filter design can be improved by replacing the vector comparison and Gauss-Newton iteration with a direct vector to quaternion conversion and by gating the two orientation estimates, as illustrated in Figure 3.13.

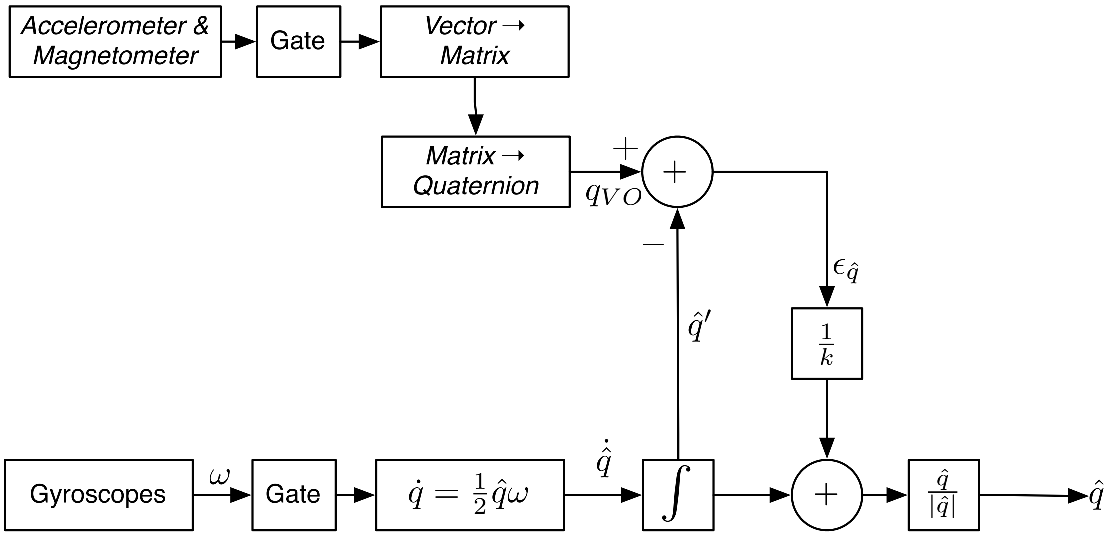


Figure 3.13: Direct Conversion Complementary Quaternion Filter

The replacement of the vector comparison stage by a direct vector to quaternion conversion allows the filter to estimate the true orientation without resort to iterative methods. Furthermore, this conversion can be efficiently implemented in fixed point maths unlike the QUEST algorithm adopted in later work by Bachmann.

The vector-to-quaternion conversion process is implemented as a two-stage process. First, an estimated rotation matrix is calculated using the method outlined in Section 3.3.3.2. This matrix is then converted to quaternion form to allow calculation of a correction factor.

There is an ambiguity when converting from a rotation matrix to a quaternion representation of rotation. 3×3 rotation matrices have a unique mapping to orientations while quaternions have a double mapping.

$$R \equiv q \equiv -q \tag{3.85}$$

The quaternion generated when converting from matrix form may be either positive or negative real depending on the exact value of R.

Integration of rate gyroscope data will produce both positive and negative real quaternions giving a smooth path through quaternion space. In order to generate corrections it is necessary that the quaternion generated by the vector observation process be the closer of the $q, -q$ pair. The decision to invert the vector observation quaternion is based on the dot product of the vector observation and rate gyroscope estimates.

$$\varepsilon_{\hat{q}} = \begin{cases} q_{VO} - \hat{q}', & q_{VO} \cdot \hat{q}' \geq 0 \\ -q_{VO} - \hat{q}', & q_{VO} \cdot \hat{q}' < 0 \end{cases} \tag{3.86}$$

Gating of the orientation estimates allows incorrect updates to be suppressed. Rather than implement the complex Kalman filter structure to weight the data, the proposed filter uses simple heuristics to determine in which sensors to place most trust.

The absolute orientation estimate based on the accelerometer data is only valid when the device is not experiencing any linear accelerations other than gravity. Therefore, the filter correction is only performed when the magnitude of the acceleration vector is within a programmable bound of $1g$. This helps reduce the effect of acceleration errors in the output and also reduces filter implementation cost by suppressing unnecessary calculations. The programmable bound, combined with the low-pass effect of the filter, allows corrections to be made in the presence of high-frequency, zero mean, accelerometer noise.

In a similar manner to the accelerometer gating, the gyroscope data is ignored when the individual gyroscope readings are below a programmable threshold. This allows suppression of bias integration errors by ignoring the gyroscopes when it is known that they have a poor signal-to-noise ratio.

The vector observation algorithm is given by Algorithm 1 and the complete filter update by Algorithm 2.

3.3.7 Summary

In a system with well understood dynamics the Kalman filter structure allows the design of an optimal orientation estimation filter. Additionally, the predictive nature of the filter allows apparent system latency to be reduced. However, for systems involving large state vectors and complex dynamics, the Kalman filter is very computationally expensive. Additions to the Kalman filter to handle non-linear systems further increase the processing costs.

The complementary Kalman filter structure reduces the complexity of the required state vector to a more manageable number of components. However, use of the Kalman error esti-

Algorithm 1: Vector observation

```
Input: $\vec{a}$ the measured acceleration vector, $\vec{m}$ the measured magnetic field vector
Output: $q_{vo}$ the estimated orientation in quaternion format
begin
    $\vec{e}_3 = \frac{\vec{a}}{\|\vec{a}\|}$;
    $\vec{e}_1 = \frac{\vec{m} - \vec{e}_3(\vec{e}_3 \cdot \vec{m})}{\|\vec{m}_t - \vec{e}_3(\vec{e}_3 \cdot \vec{m}_t)\|}$;
    $\vec{e}_2 = \vec{e}_3 \times \vec{e}_1$;
    $\hat{R} = [\vec{e}_1 : \vec{e}_2 : \vec{e}_3]^T$ ;
    $q_{vo} = \texttt{MatrixToQuaternion}(\hat{R})$;
    return $q_{vo}$
end
```

mator is still more complex than more direct methods of error estimation such as the complementary quaternion filter developed by Bachmann.

Further simplification to Bachmann's filter is provided by replacing the iterative error generation process by a direct vector-to-quaternion conversion. This substitution removes the requirement for any matrix inversions, allowing the filter to be implemented entirely in fixed point maths and greatly reduces processing requirements. Using simple heuristics to assess the reliability of sensor data allows the proposed filter to reduce errors present in the Bachmann filter.

Algorithm 2: Direct conversion complementary quaternion filter

```
begin Initialise
    $\vec{a}_0, \vec{m}_0, \vec{\omega}_0 = \texttt{GetNextSample}();$
    $\hat{q}_0 = \texttt{VectorObservtion}(\vec{a}_0, \vec{m}_0);$
end
while running do
    $\vec{a}, \vec{m}, \vec{\omega} = \texttt{GetNextSample}();$
    $\vec{a}, \vec{m}, \vec{\omega} = \texttt{ApplyScaleAndOffset}(\vec{a}, \vec{m}, \vec{\omega});$
    begin Inertial orientation update
        // Apply gate to gyroscope signal
        foreach component, c, in $\vec{\omega}$ do
            $c = \texttt{abs}(c) < g_T \,?\, 0 : c;$
        end
        // Calculate rotational rate derivative in quaternion form.
        // Assumes scaling by $1/2\Delta t$ during scale and offset correction.
        $\dot{q} = \omega \hat{q}_{t-1};$
        // Integrate rotational rate for current time step
        $\hat{q}_t = \hat{q}_{t-1} + \dot{q};$
    end
    begin Gated drift correction
        if $1 - a_T < \|\vec{a}\| < 1 + a_T$ then
            $q_{vo} = \texttt{VectorObservtion}(\vec{a}, \vec{m});$
            // Resolve ambiguity between equivalent quaternions
            if $q_{vo} \cdot \hat{q} < 0$ then
                $q_{vo} = -q_{vo};$
            end
            // Integrate error correction
            $\hat{q}_t = \hat{q}_t + \frac{1}{k}(q_{vo} - \hat{q}_t);$
        end
    end
    // Normalise quaternion to reduce effect of rounding errors
    $\hat{q}_t = \hat{q}_t / \|\hat{q}_t\|;$
end
```

Chapter 4

Simulation and Analysis of Orientation Estimation Filter

In order to more clearly illustrate the behaviour of the proposed filter (Figure 3.13), it was implemented in a Python simulation. To produce clear graphs of filter behaviour, all rotations were considered to be about the **Y** (pitch) axis. Graphs of filter output were produced by converting the quaternion output of the filter to the corresponding aerospace Euler angles.

The performance of the filter can be considered in terms of the requirements outlined in Section 3.3.2. In each example the values chosen for constants are not meant to necessarily reflect realistic conditions, but are instead chosen for illustrative purposes.

4.1 Bounded Error

4.1.1 Static Error

The simple gyroscope integration technique of Section 3.3.3 produces a cumulative output error due to gyroscope offset bias. In the complementary filter this error is reduced by the error correction derived from the additional sensors.

First, consider the case when the gyroscope offset bias is zero but there is an error in the orientation output. This scenario is analogous to filter initialisation, where there may be an error between the initial orientation estimate and the true initial orientation. In this case the measured rate quaternion, $\dot{q}$, is zero but the error, $\varepsilon_{\hat{q}}$, is large. Therefore, the estimated rate quaternion, $\hat{\dot{q}}$, is non-zero. When integrated, $\hat{\dot{q}}$ will cause the estimated orientation to converge on the true orientation at a rate determined by the filter coefficient k. This process is illustrated in Figure 4.1 where the device is simulated to be at a pitch angle of 45°. It can clearly be seen that for all values of k the filter converges on the true pitch angle from its initial estimate of 0°. The effect of k is also clearly visible with larger values causing the settling time of the filter to increase.

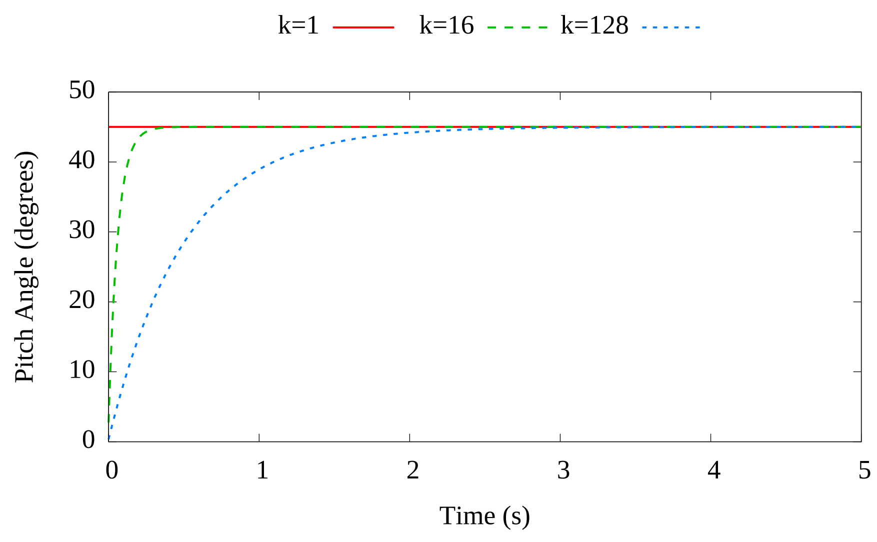


Figure 4.1: Simulated Orientation Estimation Convergence

Now consider the case when there is an offset bias error present in the gyroscope measurements. In this case the filter converges to a steady state when

$$\dot{\hat{q}} = \dot{q} - \frac{1}{k}\varepsilon\hat{q} = 0 \tag{4.1}$$

At this point no further correction can be accomplished. This therefore introduces a steady state error related to the gyroscope bias error and the filter coefficient k. This error is illustrated in Figure 4.2 for a gyroscope offset bias error of $1^{\circ}/s$ in each axis. From the figure we can

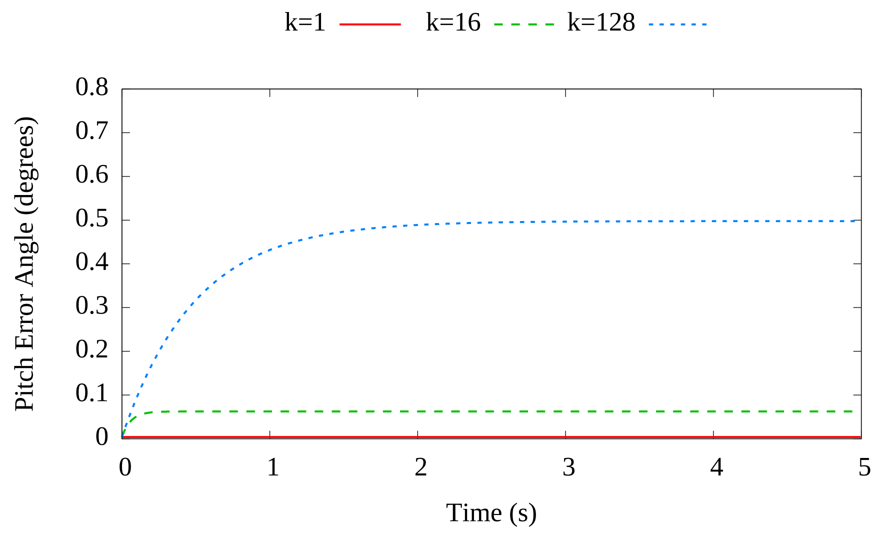


Figure 4.2: Simulated Orientation Bias Error without Gyroscope Gating

see that as k is increased there is a corresponding increase in the output error of the filter. The output bias error of the filter introduces a rotation offset between the true orientation of the device and the estimated orientation. From the graph we can see that for a static orientation the error is limited. The orientation offset introduced by gyroscope offset bias error is not constant but depends on the current orientation of the device.

As the limit of the output bias error is controlled by k so is the stability of the output error. This is illustrated in Figure 4.3 for gyroscope bias drift modelled as a random walk process with a maximum rate of change of $0.01^{\circ}/s$. From the graph we can see that as k increases

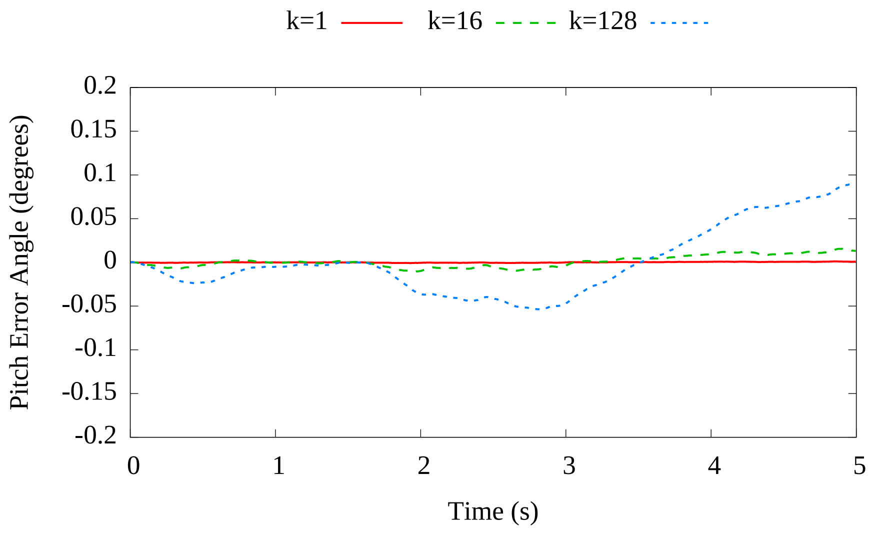


Figure 4.3: Simulated Orientation Bias Drift

the susceptibility of the filter to small changes in gyroscope offset bias becomes greater. This makes sense when we recall that increasing k increases the effect of gyroscope offset bias.

The addition of gyroscope gating allows this error to be minimised when the device is nearly stationary, increasing the static accuracy of the resulting filter. The individual components of the rotational rate vector from the gyroscopes are gated to only allow rotational rates above a programmable threshold. This allows the gyroscope data to be ignored when the device is almost stationary, when the bias errors dominate the signal. The limit on the static accuracy is therefore determined by the accuracy of the accelerometer and magnetometer based estimate. The gyroscope gating approach taken in the direct conversion filter is less powerful than the bias estimation process used in the more advanced Kalman filter implementation proposed by Foxlin but is significantly less expensive to implement.

4.1.2 Dynamic Error

4.1.2.1 Gyroscope Integration

Gyroscope bias and scale errors will result in accumulated error in the estimated orientation. Bias errors will cause the filter to consistently over-estimate rotation in one direction while under-estimating in the opposite direction. Scale factor errors will cause symmetrical errors with all rotations being over- or under-estimated depending on whether the scale factor is greater or less than unity.

Analysing the error growth for a fixed axis of rotation is relatively simple. The estimated angle, θ_t, is calculated from the previous value, θ_{t-1}, the rotational rate, ω_t, and a time step of Δt as:

$$\theta_t = \sum_{t=0}^{T} \omega_t \Delta t = \theta_{t-1} + \omega \Delta t \tag{4.2}$$

Replacing ω by a scale and offset corrupted version:

$$\theta_t = \sum_{t=0}^{T} (s\omega_t + o)\Delta t = \theta_{t-1} + (s\omega_t + o)\Delta t \tag{4.3}$$

where $s = 1 \pm \delta s$ and $o = 0 \pm \delta o$ are the scale and offset error respectively.

The error in the estimated rotation, $\delta\theta_t$, after one time step is therefore:

$$\begin{aligned} \delta\theta_t &\leq \left|\frac{\partial\theta_t}{\partial s}\right| \delta s + \left|\frac{\partial\theta_t}{\partial o}\right| \delta o \\ &\leq (|\omega_t|\, \delta s + \delta o)\, \Delta t. \end{aligned} \tag{4.4}$$

The worst case error after N integration steps can therefore be calculated as:

$$\delta\theta_N \leq N\Delta t\, (\omega_{max}\delta s + \delta o)\,, \tag{4.5}$$

where ω_{max} is the maximum rotational rate during the integration.

The partial derivatives in Equation 4.4 provide estimates of the error sensitivity. The sensitivity to bias errors is constant, however, the sensitivity to scale errors increases as the rotational rate increases. When the rotational rate is low, $|\omega|\, \delta s << \delta o$, the per time step integration error is dominated by the offset.

Attempting to estimate the maximum error after N integration steps when both the rate and axis of rotation are time varying functions is exceedingly complex due to the non-linear nature of the problem. However, the understanding gained from the simpler fixed axis case can still be applied. The effect of scale factor errors will dominate during rapid rotations while slow rotations will suffer more from bias. The error in the estimated orientation will steadily increase. The resulting error is bounded to within $\pm 180^{\circ}$ only by the modulo arithmetic properties of rotations.

Gyroscope integration is also subject to error in the event that the rotational rate of the device exceeds the maximum rate of the gyroscopes. The resulting error is similar to that of a scale factor less than unity as the gyroscope signal saturates at its maximum value.

It it clear that, without drift correction, the errors introduced by gyroscope bias and scaling factor errors will lead to the estimated orientation becoming increasingly unreliable. It is therefore important to investigate the limits of drift corrections in the face of linear accelerations.

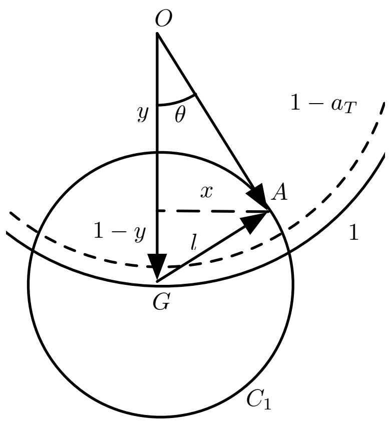


Figure 4.4: Geometry of θ_{max} including Gating

4.1.2.2 Vector Observation

The maximum error due to a given magnitude of linear acceleration was developed in Section 3.3.3.3 as Equation 3.71. The effect of applying a gating process to vector observation, as introduced in Section 3.3.6, is now considered.

The accelerometer gating process states that vector observation and drift correction is only performed if the magnitude of the measured acceleration is within a programmable bound of $1g$. That is:

$$1 - a_T < \|\vec{g} + \vec{a}\| < 1 + a_T, \tag{4.6}$$

where a_T is the gating threshold. This condition is necessary but not sufficient to indicate that the measured acceleration corresponds to gravity [65]. There are infinitely many solutions to Equation 4.6 other than the trivial solution $\vec{a} = \vec{0}$.

To understand the effects of gating it is necessary to consider the geometry shown in Figure 4.4. In the case where the point (x, y), previously calculated by Equations 3.70 and 3.69, lies within a_T of the circle:

$$x^2 + y^2 = 1, \tag{4.7}$$

which represents the locus of possible $1g$ accelerations, the maximum error is the same as without gating. If this condition is not satisfied then the maximum error occurs at the intersects of Equation 3.68a and the minimum acceleration gate threshold:

$$x^2 + y^2 = 1 - a_T. \tag{4.8}$$

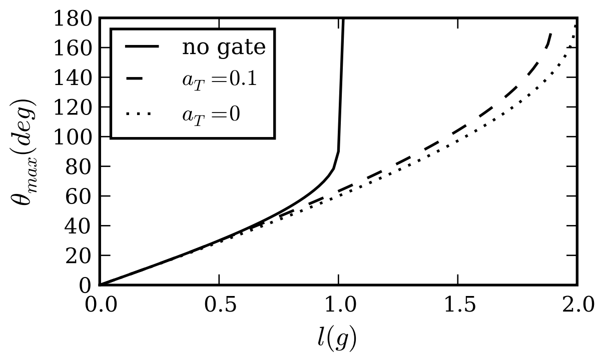


Figure 4.5: Effect of gating threshold a_T on θ_{max}.

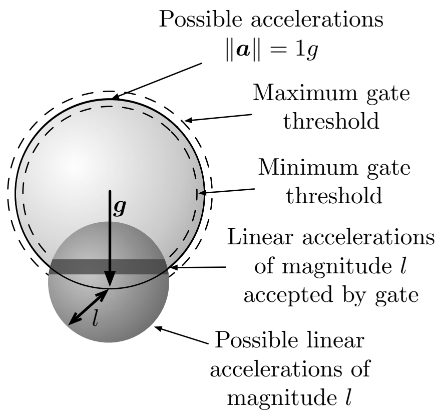

Possible accelerations
$\|\boldsymbol{a}\| = 1g$
Maximum gate threshold
Minimum gate threshold
$\boldsymbol{g}$
Linear accelerations of magnitude l accepted by gate
l
Possible linear accelerations of magnitude l

Figure 4.6: Visualising effect of accelerometer drift correction

The resulting error can then be calculated as:

$$y = \frac{(1-a_T)^2 - l^2 + 1}{2} \tag{4.9a}$$

$$x = \sqrt{l^2 - (1-y)^2} \tag{4.9b}$$

$$\theta_{max} = \tan^{-1}\left(\frac{x}{y}\right). \tag{4.9c}$$

The effect of applying gating to accelerometer data is shown in Figure 4.5. By applying a gating threshold the maximum error is reduced for linear accelerations with magnitude less than $2g$. The reduction in worst case provides little obvious practical benefit. The differences are essentially indistinguishable until the error reaches 40°. However, gating provides a secondary advantage, as the probability of incorrect drift corrections is decreased.

To understand the potential reduction in error probability it is useful to consider the simple case of a linear acceleration vector, of fixed magnitude, that is equally likely to point in any direction. The solutions to the vector sum $\vec{a} = \vec{g} + \vec{l}$ can be visualised as the surface of a sphere, as illustrated in Figure 4.6. Without gating all of these possible solutions would be accepted by the filter with the majority resulting in error. When gating is applied only solutions within

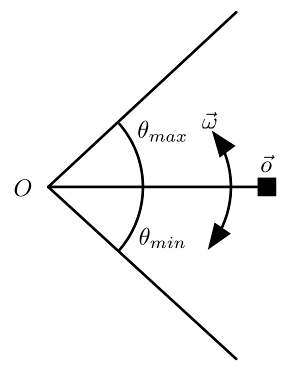


Figure 4.7: Oscillating Beam Scenario

the shaded zone, bounded by intersects of the minimum and maximum gate thresholds, are accepted. The probability of error is therefore reduced to the ratio of the surface area of the zone to the surface area of the sphere.

The reduction in error probability is determined by the three-dimensional distribution of linear accelerations. For some motions gating will successfully reject all erroneous vector observations, whereas for other motions gating may be guaranteed to still accept error. This is best illustrated through the use of simulation.

4.1.2.3 Simulation of the Effects of Linear Accelerations

The simplest motion guaranteed to introduce error in vector observations is that of a device undergoing constant linear acceleration. From the analysis shown in Section 3.3.3.3 it is simple to demonstrate that this results in a constant error with maximum value according to Equation 3.71. This type of error does not occur in natural motions as constant non-zero acceleration would result in an ever increasing velocity.

A more interesting scenario is that of a device mounted on the end of an oscillating beam, as illustrated in Figure 4.7. This type of motion is common in human movements such as walking, waving and shaking hands.

The sensor is mounted at an offset, $\vec{o}$, from the origin and rotates at a constant rate, $\vec{\omega}$ measured in rad/s, between the angles θ_{min} and θ_{max}. The acceleration experienced by the device, in m/s^2, is given by:

$$\vec{a}_r(t) = -\vec{o}(t)\,\|\vec{\omega}(t)\|^2. \tag{4.10}$$

The resulting acceleration is given the subscript r to indicate that it acts in a radial direction.

Equation 4.10 assumes that the offset vector is orthogonal to the rotational rate vector. When this is not the case the offset vector must be projected on to the plane normal to ω. The generalised equation for radial acceleration is defined as:

$$\vec{a}_r = (\vec{\omega}\cdot\vec{o})\,\vec{\omega} - \vec{o}\,\|\vec{\omega}\|^2. \tag{4.11}$$

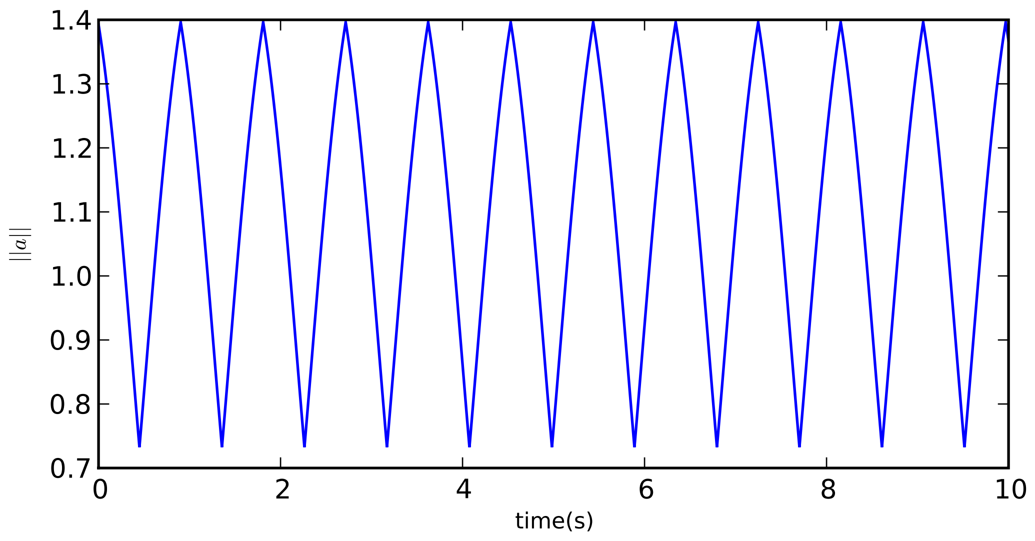


Figure 4.8: Simulated Acceleration Magnitude

$o = 40\text{cm}, \omega = 200^\circ/\text{s}, -45^\circ \leq \theta \leq 45^\circ$

Finally, to convert to units of g for acceleration requires scaling $\vec{a}_r$ by the nominal value of $g = 9.81\text{m}/\text{s}^2$.

It is clear from Equation 4.11 that the magnitude of the linear acceleration is constant, however, the direction is a function of time. Therefore the magnitude of the acceleration experienced by the device is also a time varying function. An example trace of the magnitude of the linear acceleration experienced by a device, at an offset of 40cm, rotational rate of $200^\circ/\text{s}$ and $-45^\circ \leq \theta \leq 45^\circ$, is shown in Figure 4.8.

The result of this linear acceleration on orientation estimation accuracy is shown in Figure 4.9. The error is calculated by:

$$\theta_\varepsilon = \theta - \hat{\theta}, \tag{4.12}$$

where θ_ε is the error angle, θ is the true angle and $\hat{\theta}$ the output of the filter. A gated and non-gated implementation of the filter was simulated, in both cases the filter co-efficient, k, was set to 128. From the figure it can be seen that the non-gated filter converges on an error of approximately 25°, while the gated filter converges on an error of approximately 28.5°. Calculating the worst case error, using Equations 3.71 and 4.11, results in $\theta_{max} = 29.8^\circ$.

Although the gated filter converges on a greater error it does show two interesting properties. Firstly, the rate of error growth is less than that of the non-gated filter. Secondly, the gated error shows less variation in the steady state region. To understand the cause of these properties it is useful to consider Figure 4.10 which shows the error angle, ϕ, measured between the gravity vector and the experienced acceleration.

$$\phi = cos^{-1}\left(\frac{\vec{a}\cdot\vec{g}}{||\vec{a}||\,||\vec{g}||}\right) \tag{4.13}$$

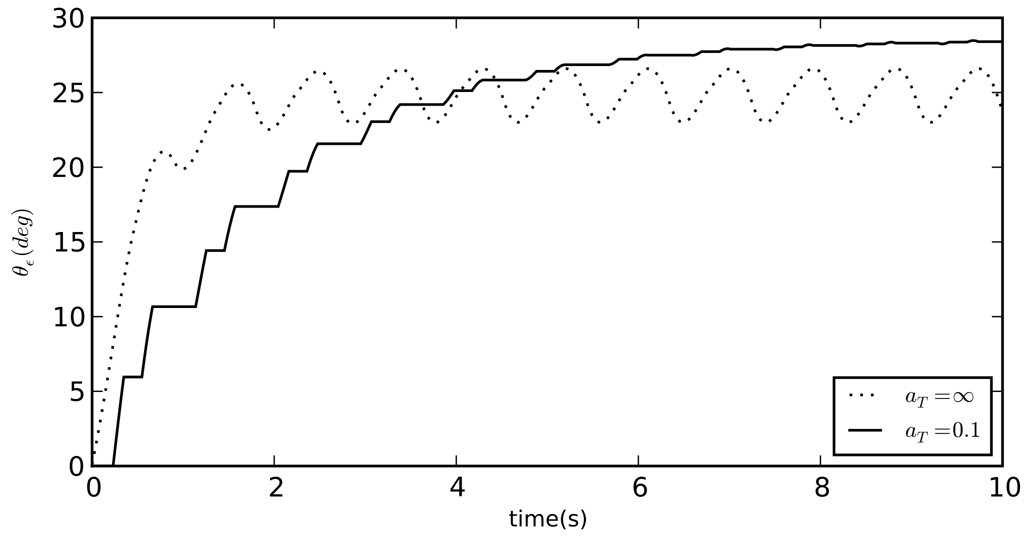


Figure 4.9: Simulated Error

$k = 128, o = 40\text{cm}, \omega = 200^\circ/\text{s}, -45^\circ \leq \theta \leq 45^\circ$

The solid regions of the trace in Figure 4.10 indicate those regions that are accepted by the gate condition. The mean value of the errors accepted by the gate is 28.57°, which corresponds to the estimated limit of the gated error. Similarly the mean value of the non-gated errors is 24.85°. This illustrates a limitation of the gating process in that, in addition to discarding erroneous vector observations, it potentially rejects vector observations with lower error than the current estimate.

Repeating the simulation with the range of motion modified to $-135^\circ \leq \theta \leq -45^\circ$ results in the errors shown in Figure 4.11. The range of motion is the same but the entire system has been rotated through 90°. Under these conditions the gated filter accepts zero vector observations while the non-gated filter continues to accept all observations. In this scenario the mean error between the gravity and acceleration vectors is zero but has a substantial variance resulting in the peak to peak error of the non-gated filter.

For the non-gated filter varying the value of the filter co-efficient, k, affects the variation in the stable state as shown in Figure 4.12. Larger values of k result in less variation and a longer convergence time. Similarly, for the gated filter, increasing the value of k results in increased convergence time.

To investigate if the problems related to linear acceleration are restricted to the proposed filter model two alternative filters were implemented: an Extended Kalman Filter (EKF) developed Yun et al. [66], and the Complementary Kalman Filter (CKF) developed by Foxlin [63].

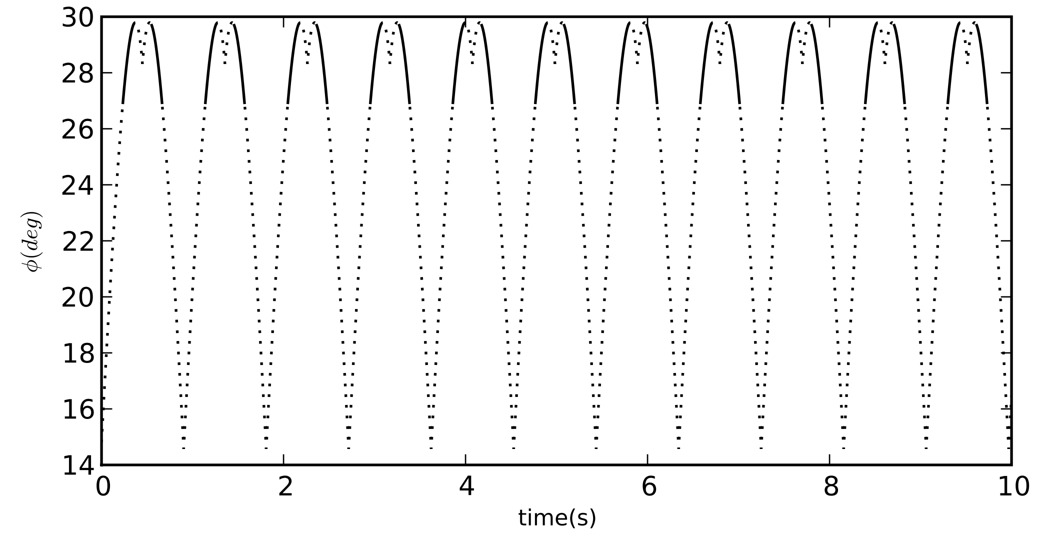


Figure 4.10: Simulated Error Angle between $\vec{a}$ and $\vec{g}$
$o = 40\text{cm}, \omega = 200°/\text{s}, -45° \leq \theta \leq 45°$

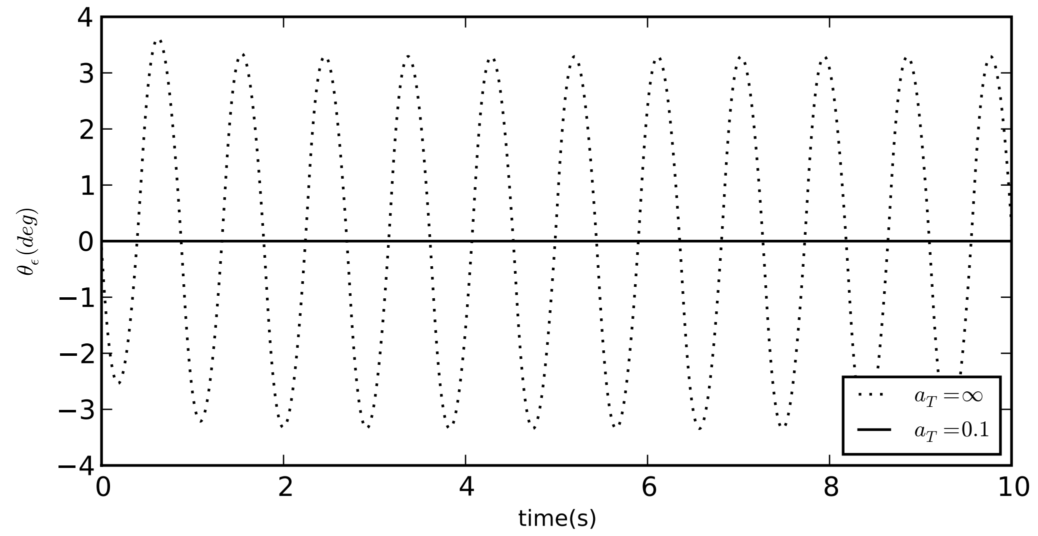


Figure 4.11: Simulated Error
$k = 128, o = 40\text{cm}, \omega = 200°/\text{s}, -135° \leq \theta \leq -45°$

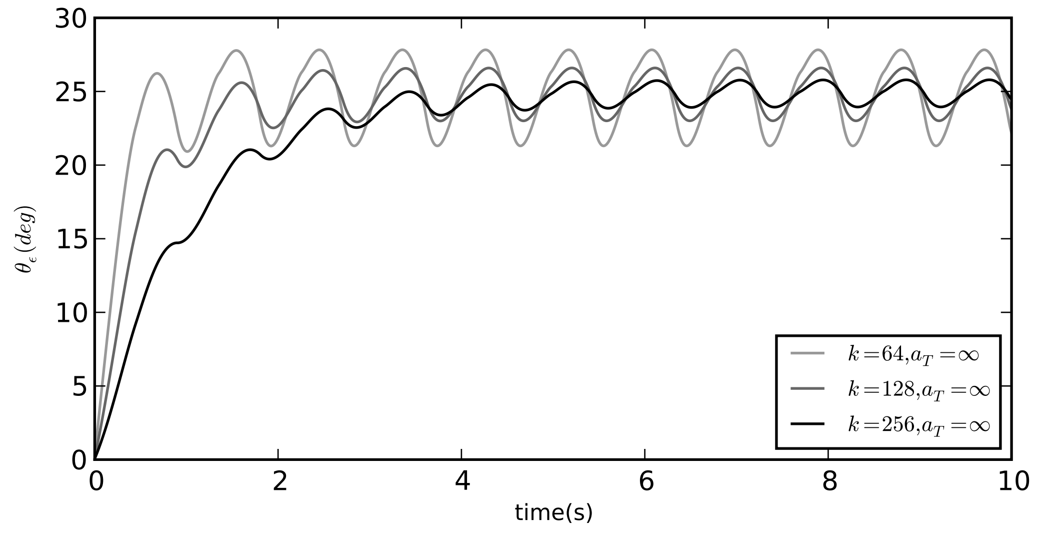


Figure 4.12: Effect of k on θ_ε

$o = 40\text{cm}, \omega = 200°/\text{s}, -135° \le \theta \le -45°$

The resulting error plots for the two scenarios are shown in Figure 4.13 and Figure 4.14 respectively. The figures indicate that the errors associated with linear accelerations are common to all of the tested filters. The CKF implementation performs almost identically to the non-gated filter implementation, while the EKF shows significantly increased error in both scenarios. This is attributed to the mismatch between the movement scenario tested and the scenario assumed in the development of the EKF process model, where rotational rate is modelled as a coloured noise signal.

The previous movement scenario, while having analogs in typical human motions, is not characteristic of all motions. It is therefore interesting to investigate alternative motions. The second scenario tested is that proposed by Yun et al. [66] of a limb segment undergoing random variations in rotational rate. This model is based on the movements of the arms in everyday motions such as gesturing.

As with the previous movement scenario the sensor is taken to be mounted on the end of a rigid beam. The rotational rate of this beam is driven by the coloured noise process:

$$\omega_{t+1} = e^{-\frac{\Delta t}{\tau}} \cdot (\omega_t + \mathcal{N}(0, D\Delta t)), \tag{4.14}$$

where Δt is the time step, τ is a time constant controlling how fast a limb segment can move, and $\mathcal{N}(0, D\Delta t)$ is a normally distributed white noise source with standard deviation D rad$^2/\text{s}^2$. An example rotational rate trace is shown in Figure 4.15 with parameters from [66].

The changes in rotational rate, ω, correspond to angular accelerations.

$$\alpha(t) = \frac{d}{dt}\omega(t) \tag{4.15}$$

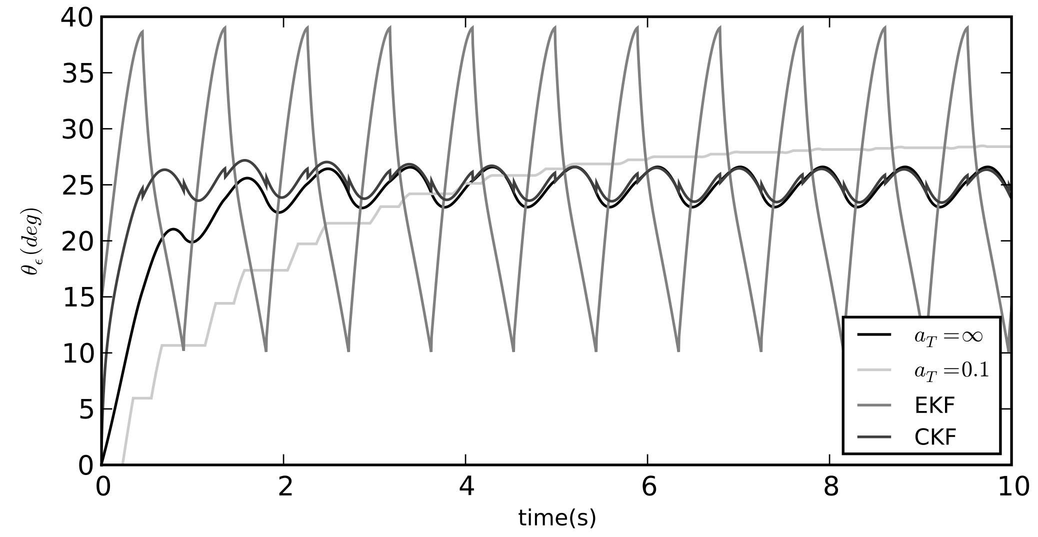


Figure 4.13: Simulated Error

$o = 40\text{cm}, \omega = 200^\circ/\text{s}, -45^\circ \leq \theta \leq 45^\circ$

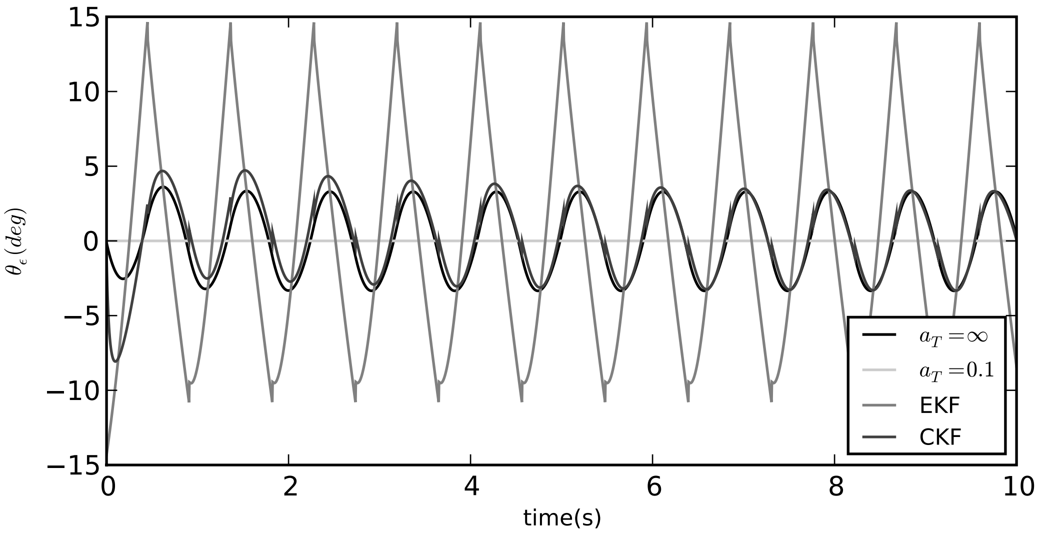


Figure 4.14: Simulated Error

$o = 40\text{cm}, \omega = 200^\circ/\text{s}, -135^\circ \leq \theta \leq -45^\circ$

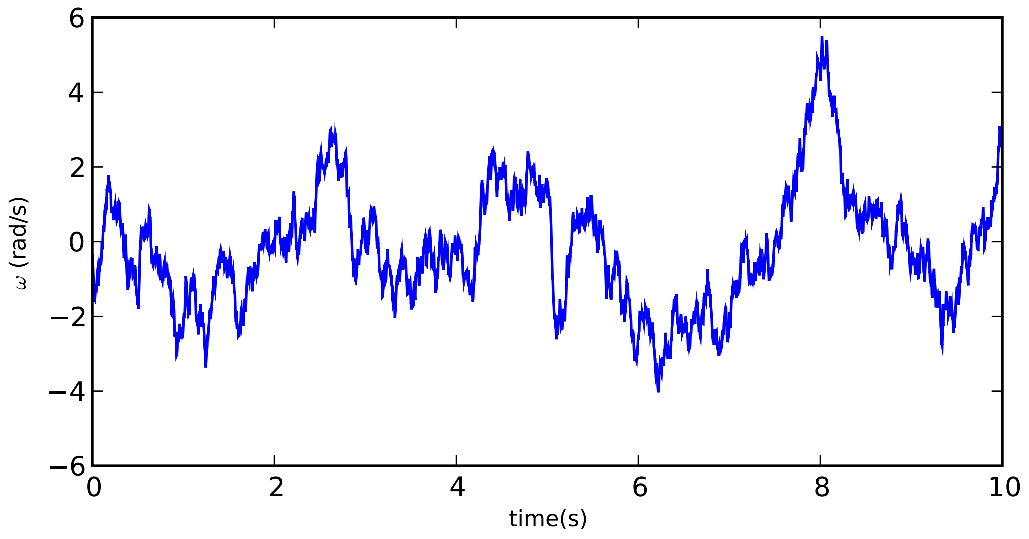


Figure 4.15: Simulated ω

$\Delta t = 1/256\text{s}, \tau = 0.5\text{s}, D = 50\text{rad}^2/\text{s}^2, o = 40\text{cm}$

These changes in rotational rate result in tangential accelerations, a_t, given by:

$$\vec{a}_t = \vec{o} \times \vec{\alpha}. \tag{4.16}$$

The total linear acceleration, in m/s^2, experienced by the device is the sum of the radial and tangential accelerations:

$$\vec{a} = \vec{a}_t + \vec{a}_r = \vec{o} \times \vec{\alpha} + (\vec{\omega} \cdot \vec{o})\,\vec{\omega} - \vec{o}\,\|\vec{\omega}\|^2. \tag{4.17}$$

The results of simulating the filters are shown in Figure 4.16. The gated, non-gated and EKF filters all track the reference angle closely. The CKF implementation tracks well until the sudden change in θ as the Euler angle wraps from -180° to 180°. This demonstrates a major flaw of the CKF implementation that, as it uses Euler angles internally to track orientation, is unable to track through Euler discontinuities. The other filters, using quaternions internally to represent orientation, have no difficulty with these transitions. This limitation of the CKF filter prevents its use in full body motion tracking where the orientations of the limb segments may not be covered by Euler angles without discontinuities.

To compare the errors between the gated, non-gated and EKF filter a Monte Carlo simulation was performed. The constraint of rotation around a fixed axis was removed allowing the simulated device to rotate in all directions. The driving function of Equation 4.14 was extended to cover the three orthogonal rotation axes. The input noise signal for each axis was independent.

The error function was redefined as:

$$\theta_\varepsilon = 2\cos^{-1}\left(q_{ref} \otimes q^*_{filt}\right)_w, \tag{4.18}$$

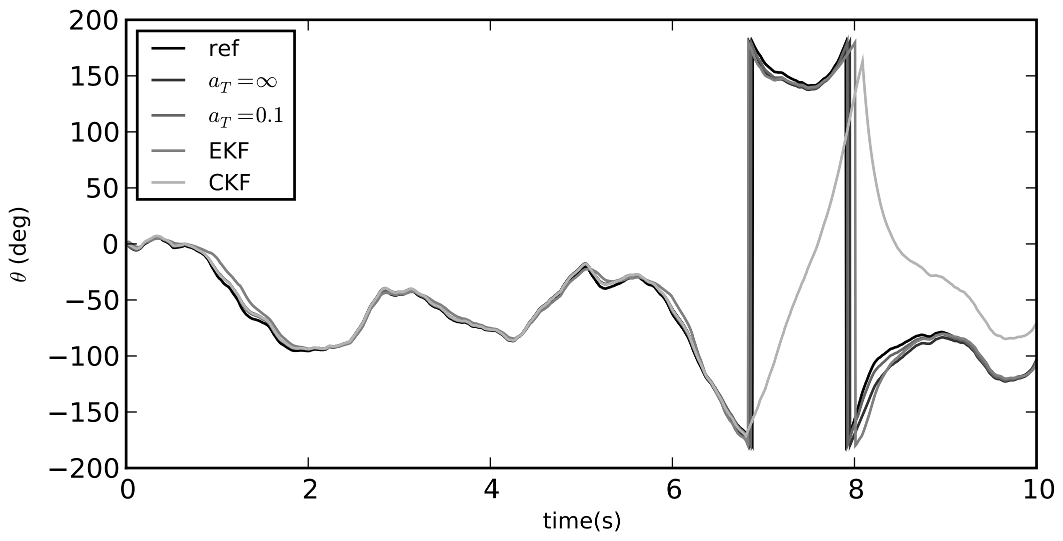


Figure 4.16: Simulated Random Walk

$\Delta t = 1/256\text{s}, \tau = 0.5\text{s}, D = 50\text{rad}^2/\text{s}^2, o = 40\text{mm}$

where q_{ref} is the reference orientation of the device, calculated directly from the rotational rate, and q_{filt} is the output of the orientation filter. The quaternion product $q_{ref} \otimes q^*_{filt}$ encapsulates the axis and angle of the orientation error. The error function therefore returns the absolute error angle about a time-varying axis.

The filter implementations where simulated for 1000 runs, each of thirty simulated seconds. The offset of the sensor from the point of rotation was additionally varied for each run. The offset was distributed uniformly over the range $[0, 50]$ cm. The same data was used to drive each filter in each iteration. The estimated Cumulative Distribution Function (CDF) of the output error for each filter was generated from the resulting 7.68 million error estimates. The resulting estimated error distribution functions are shown in Figure 4.17.

Examination of Figure 4.17 clearly reveals that the gated filter performs the best with 90% of the errors less than 7.1°, compared to 10.5° for the non-gated filter and 16.4° for the EKF.

4.2 Latency and Processing Cost

The latency of a digital filter is comprised of two elements: processing delay and group delay.

The group delay of a filter is the number of samples delay between an input sample being captured and its maximum effect on the output. This delay is related to the implementation of the filter. In the case of the complementary filters there is no group delay as gyroscope data is directly integrated, so there is no delay between a gyroscope sample and its maximum effect on the filter output. Figure 4.18 illustrates the latency of the complementary filter compared to the latency of the naive estimate based solely on filtered accelerometer and magnetometer data.

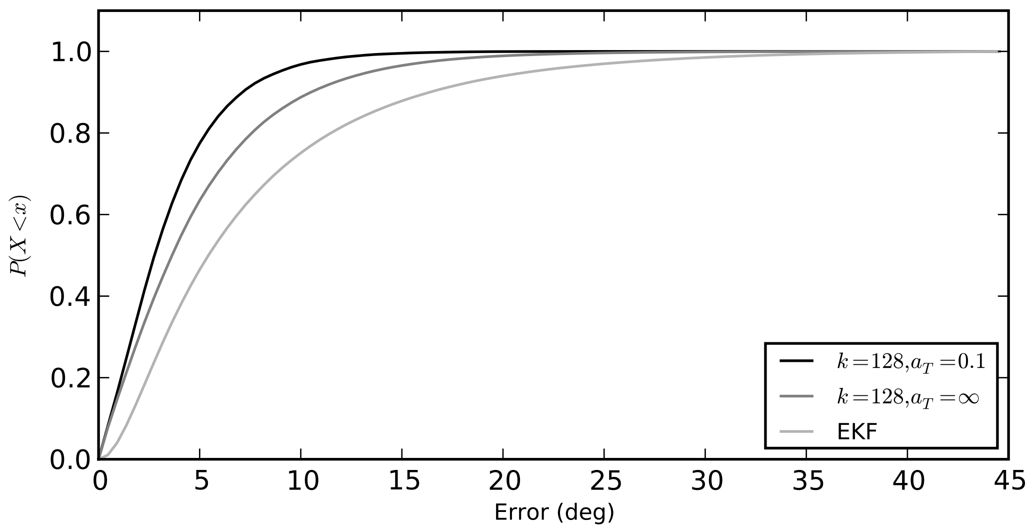


Figure 4.17: Estimated Error Cumulative Distribution Functions

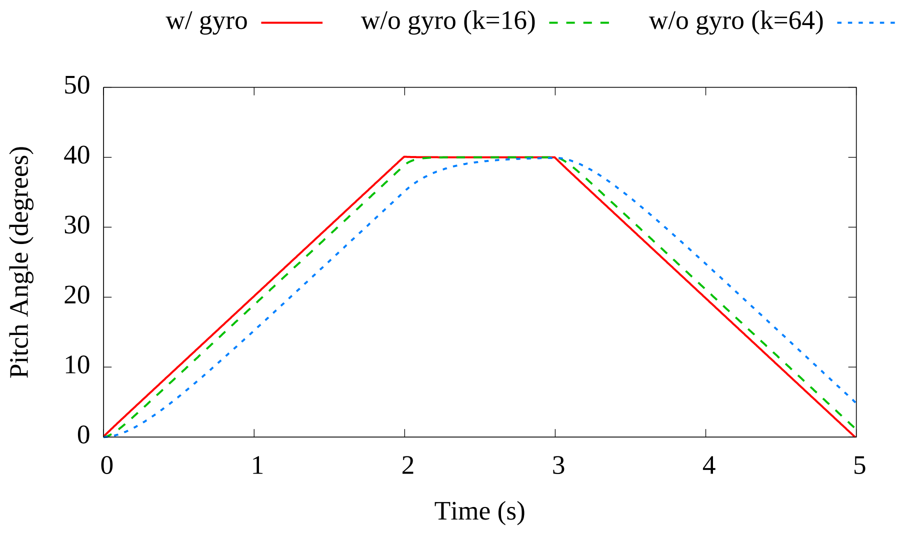


Figure 4.18: Simulated Latency Comparison

The device was simulated to perform a $20^{\circ}/s$ turn around the y-axis starting at $t = 0$ stopping at $t = 2$. The simulated device was then held steady for one second before returning to its start position with a $-20^{\circ}/s$ turn. From the graph we can clearly see that the complementary filter exactly follows the motion of the simulated device while the naive estimate displays significant delay. The delay in the naive estimate is controlled by the filter co-efficient k. Smaller values decrease the group delay but also increase the effect of dynamic movement noise on the output.

The processing delay is the time between a sample being captured and the update of the filter state and is related to the complexity of the filter. In the case of the direct conversion filter this delay is effectively constant as there is no iteration in the design.

The complexity of the filter implementation is very important. In order to reduce the required network data rate it is desirable to implement the filter locally on the sensing device. As such devices are power constrained due to small batteries they must use low power embedded processors. These processors have no acceleration for floating point math functions and require algorithms to use fixed point maths for speed. The complexity of the filter implementation directly affects both the ease of implementation and the performance of the filter.

The complementary filter can be implemented using only basic vector operations, such as cross and dot products, quaternion multiplies and vector normalisations. In contrast, the Kalman filter structures require complex operations, such as matrix inversion, and in Foxlin's CKF, numerous trigonometric functions.

In order to evaluate the relative performance of the proposed complementary filter and the extended Kalman filter they were implemented in Python. The implementations were done in floating point using the NumPy library functions where possible to ease coding. In addition to the basic complementary and EKF filter structures various vector observations algorithms were implemented, including TRIAD [67], FQA [54], and QUEST [57].

The performance of the different implementations was compared [20] using 1000 samples of data from a device at rest. The normalised execution time compared to the proposed filter are shown in Figure 4.19. The results indicate the best of ten runs of each of the filter implementations. It is clear that the complementary filter is significantly faster than the EKF. It is this reduction in processing requirements that allows the use of the complementary filter in a low power sensor network device.

4.3 Noise

The primary source of high frequency noise in an inertial tracking system is related to the accelerometers measuring movement induced accelerations. In the complementary filter this noise in the acceleration vector becomes noise in the error quaternion $\varepsilon_{\hat{q}}$.

Considering the steady state of the filter the noise transfer function applied to the ac-

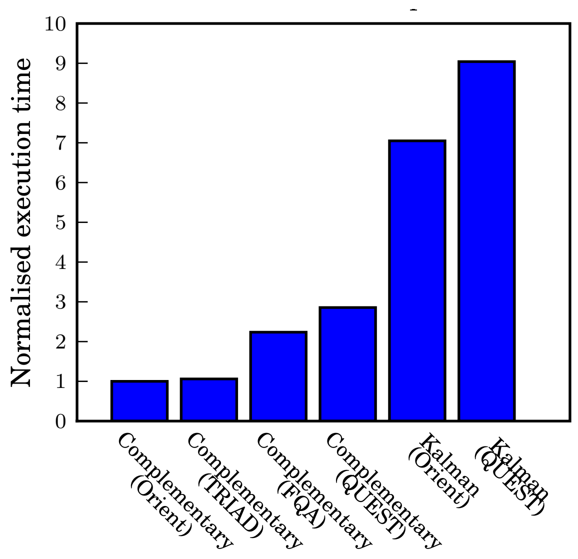


Figure 4.19: Normalised Execution Time to Process 1000 Samples

celerometer and magnetometer signals is

$$H(s) = \frac{1}{k(1 - s(1 - k^{-1}))} \tag{4.19}$$

which is a standard first-order low-pass filter with cut-off determined by k. The effect of varying k is illustrated in Figure 4.20. In performing this simulation it was assumed that the accelerometers were corrupted with white noise with a standard deviation equivalent $0.01g$ and the gyroscopes by white noise with standard deviation equivalent to $2^{\circ}/s$

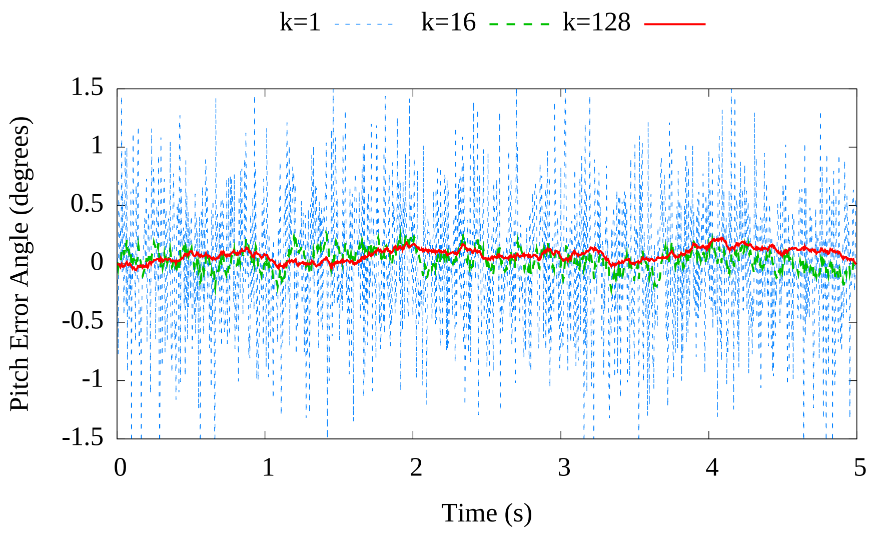


Figure 4.20: Simulated Noise Reduction

4.4 Summary

The analysis of the proposed filter shows that it meets many of the requirements set out in Section 3.3.2.

The processing cost of implementing the filter compares very favourably with other proposed solutions. In addition the simplicity of the filter structure aids in the development of an embedded fixed point implementation.

The filter displays a bounded error in the estimation of static orientations controlled by two parameters: the filter co-efficient, k, and the gyroscope gating threshold. The maximum static error is directly proportional to the value of k without gyroscope gating. The introduction of gating allows the filter to converge to the best estimate of the true orientation available from the vector observation algorithm. The gating value chosen for the gyroscope should be set to the absolute value of the maximum deviation of the gyroscope signal while at rest.

The difficulty of estimating dynamic orientations is common to all the examined filters. The cause of this difficulty is the presence of linear accelerations due to movement. The vector observation gating parameter a_T can be used to reduce the probability of linear accelerations corrupting the estimated state. In general a_T should be set at approximately the deviation in the accelerometer magnitude due to random noise. Setting a_T to this value gives the highest probability of accepting accurate vector observations when the device is stationary while rejecting inaccurate observations during movement. For randomised motions the gating process generally reduces errors, however, for continual oscillatory motions the gating can lead to a marginally increased error.

Selecting a value for k is a more subjective process. Too small a value leads to noise in the output estimate and higher sensitivity to linear accelerations but faster recovery from errors, while too large a value leads to reduced error occurrence but slower recovery. Setting k to too large a value will cause the filter to become unstable as the per time step drift becomes greater than the maximum correction. Choosing a value for k is therefore more a matter of trial and error to achieve a qualitatively satisfactory result.

The dynamic accuracy is affected greatly by the motion of the device and it is not possible to state an absolute bound on the error other than for very well defined motions. In the case of a well defined motion then Monte Carlo simulations can be effectively used to assess error probabilities and determine filter parameters.

Part III

Implementation & Performance Evaluation

Chapter 5

Implementation

5.1 Design Process and Goals

The goal of the implementation stage of the project was to build a relatively low cost, wirelessly networked, orientation sensor for use in real-time postural tracking applications. The implementation was conducted in parallel with the development of the theoretical aspects discussed in Chapter 3. The parallel development of these aspects allowed each one to inform the design of the other.

5.1.1 Initial Prototype Design: *Orient-1*

In order to test the practicality of the overall goal an early prototype was developed that performed orientation estimation, using the algorithm described in Section 3.3.3.2, on a host PC, based on data from accelerometers and magnetometers. The prototype was developed during the summer of 2005, and was demonstrated at the Fourth Workshop in Speckled Computing, held in September 2005.

The prototype was implemented as an expansion board for the existing ProSpeckz-IIK [68]. The ProSpeckz-IIK platform is the standard prototype platform used within the Speckled Computing Consortium. It has a Cypress PSOC processor, that provides re-configurable analogue and digital elements in addition to an 8-bit micro-controller core, and a Chipcon CC2420 250 kbps 2.4GHz Zigbee compatible radio. The prototype board is shown in Figure 5.1 , mounted on a ProSpeckz-IIK. A block diagram of the system is shown in Figure 5.2.

The prototype device used a single STMicroelectronics LIS3L02AQ three-axis accelerometer and two Honeywell HMC105X series magnetometers. The components were chosen for their small size and availability. The PSOC was configured to have three 12-bit ADCs. At the input to each ADC there was a Programmable Gain Amplifier (PGA). The PGA was used to adjust the gain applied to each sensor set to maximise the resolution of the ADC process. The ADCs were used to sample the accelerometers and magnetometers in turn. Complete sample

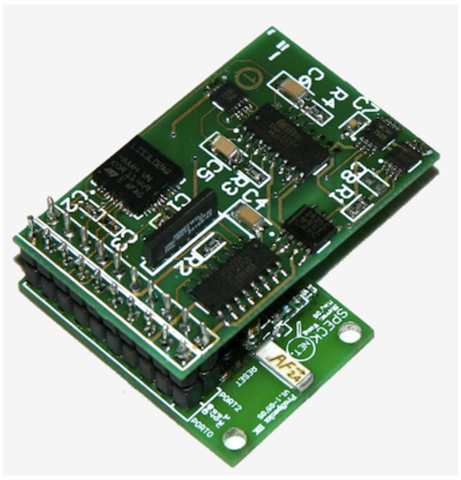


Figure 5.1: Prototype Device: *Orient-1*

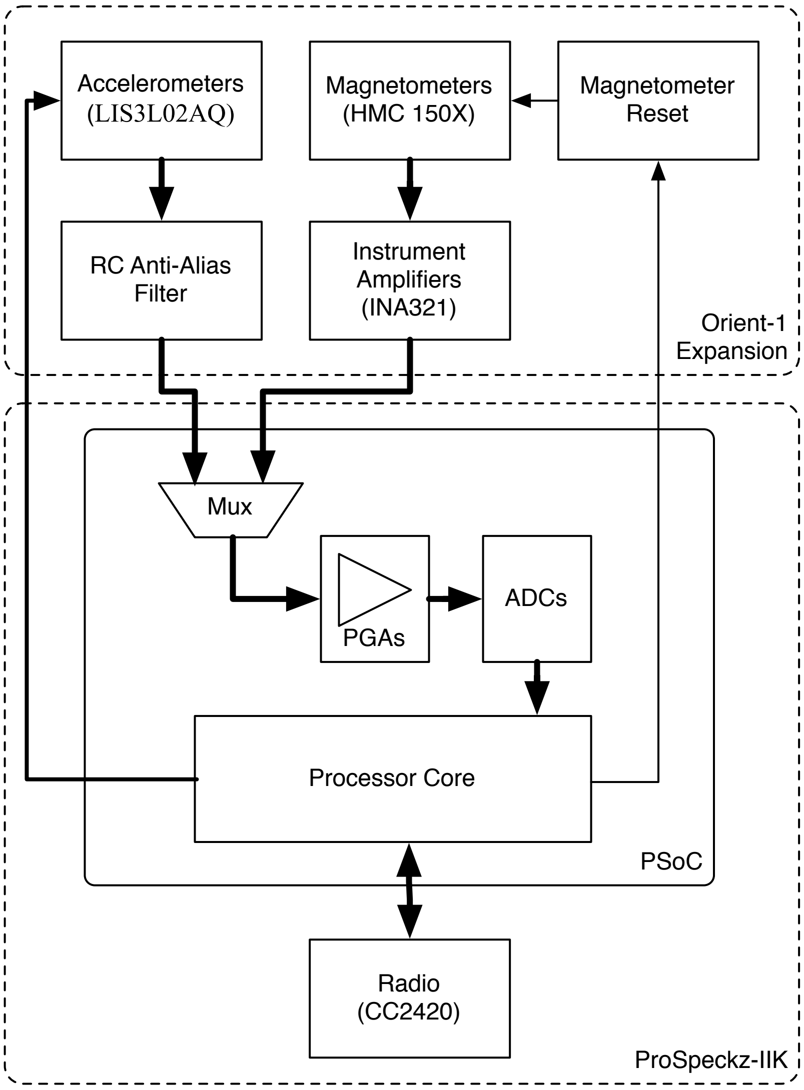


Figure 5.2: *Orient-1* Block Diagram

sets of all six inputs were gathered at approximately two hundred samples per second.

The accelerometer was directly connected to the processor through a first-order, low-pass filter specified in the device data sheet. The accelerometer provided two selectable sensitivity values: $2g$ and $6g$. The magnetometers were connected to the PSOC through a high gain instrument amplifier to convert the differential magnetometer output to a single-sided voltage for AD conversion.

5.1.2 Lessons Learned From *Orient-1* Prototype Design

Four prototype units were constructed; these were used to drive a simple body model with two arms. The experience of using these devices provided many useful lessons:

1. Sensor configuration

   The system without rate gyroscope displayed poor dynamic response due to the high degree of low pass filtering required to reduce jitter, caused by dynamic accelerations, to acceptable levels. In addition to loss of useful high frequency information the low pass filtering introduced noticeable delays between subject movement and body model updates.

2. Accelerometer device selection

   The original accelerometer chosen had a range of $\pm 2g$ at its most sensitive setting. As the orientation estimate is inaccurate during periods of dynamic acceleration, the sensitivity of the accelerometer does not need such a large range. In fact greater range decreases the accuracy of the orientation estimate as fewer ADC values are used to represent the range of interesting sensor values.

3. Magnetometers and support circuitry

   The magnetic sensing was provided by a two-axis magnetometer in a surface mount package and a single axis device in a through-hole Single Inline Package (SIP). The use of a SIP mounted single axis magnetometer to complete three axis sensing proved problematic as it was hard to ensure that the magnetometer axes were orthogonal. In addition to being easy to knock out of alignment, the SIP package required much greater board area than the two-axis surface mount device.

   The single-stage amplifier design posed problems due to bias offset in the input stage of the amplifier. Very high precision amplifiers where required to allow reasonable gain without variations in input bias between different devices causing clipping in the magnetometer data. This could have been reduced by using a two-stage amplifier design combing two, medium-gain stages, as opposed to a single, high-gain stage.

4. Medium access control

   Attempting to run multiple devices sharing a single radio channel in an uncoordinated manner proved a significant problem. The fixed packet transmission period employed by all devices caused problems with network access starvation unless the devices were luckily initialised in such a way that they did not attempt to access the channel simultaneously.

5. Processor choice

   The small 8-bit core of the PSOC was found to be completely unsuitable for implementation of even simple orientation filtering algorithms. 16-bit integer operations such as multiplications and additions required many processing cycles to complete.

   The necessity of manually managing sampling to ensure that the accelerometers and magnetometers could share access to the ADCs increased the complexity of the code and its susceptibility to timing variations.

5.2 *Orient-2* Hardware

5.2.1 Design of *Orient-2*

The lessons learnt from the development of the original prototype were employed in the design of its successor, *Orient-2* [19]. For this design it was decided to go with a full custom redesign in order to overcome the limitations of the early prototype. The new design would aim to be a completely self-contained three degree of freedom orientation tracker. The design of *Orient-2* began in December 2005 and functional devices were produced in time for the 5th Workshop in Speckled Computing, held in September 2006.

In order to improve the dynamic response of the orientation estimation process it was decided to augment the existing sensor configuration with three axis rotational rate sensing. Due to the lack of available multi-axis miniature gyroscopes it was decided to go with a three dimensional structure composed of multiple interlocking Printed Circuit Boards (PCBs). This structure would allow the use a second surface mount magnetometer to overcome the difficulties encountered with the original SIP device.

A block diagram of the *Orient-2* device is shown in Figure 5.3. The complexity of the system is greatly increased over that of the earlier prototype. This is due to the inclusion of the rotational rate gyroscopes and associated support circuitry, and the extra power control circuitry required to support low power modes.

A complete *Orient-2* device is shown in Figure 5.4(a). The device measures $36x26x11$mm including an internal battery but excluding the radio whip antenna. The final device is composed of four PCBs (Figure 5.4(b)):

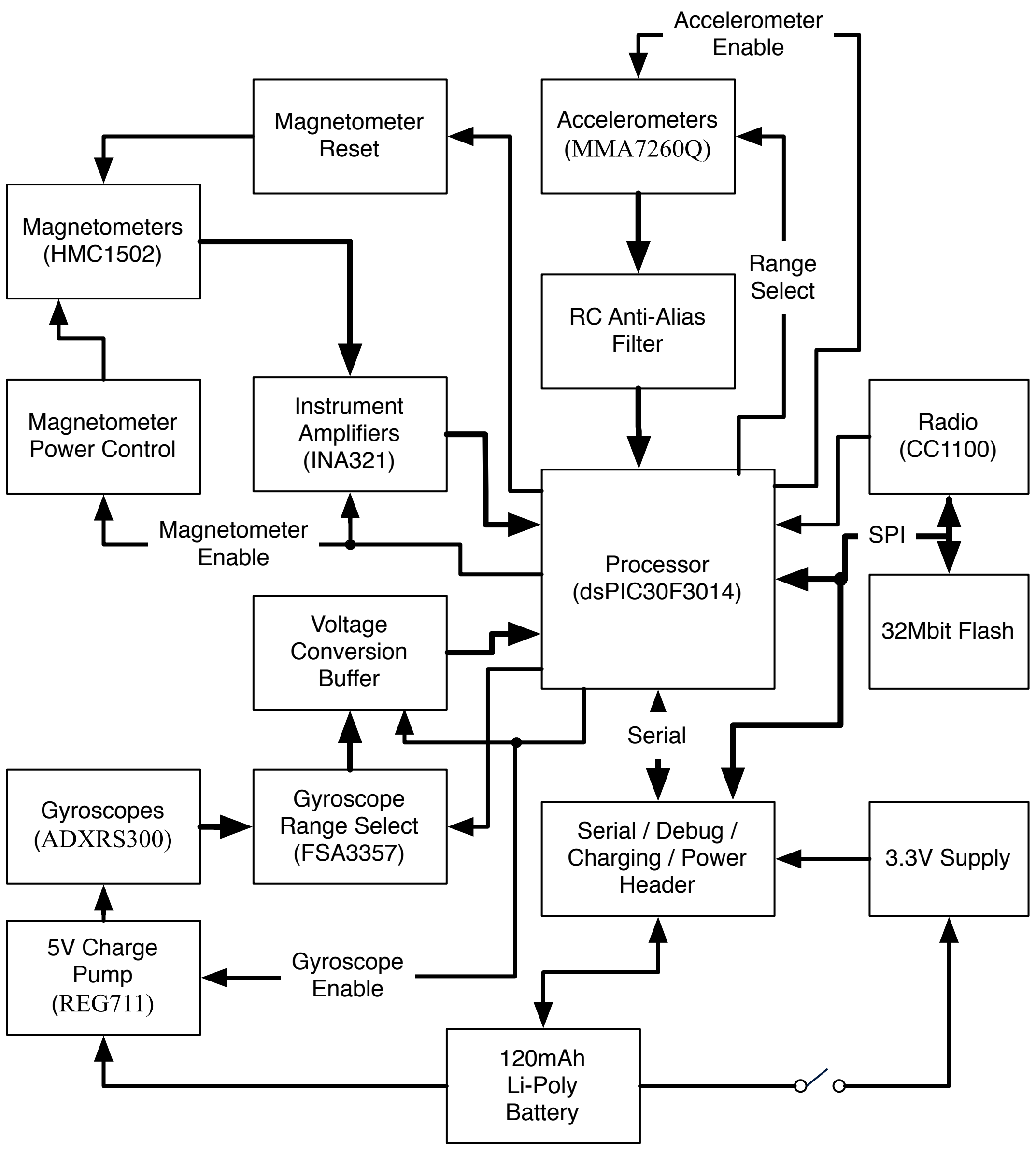
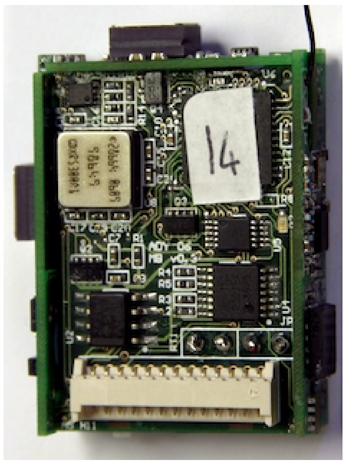
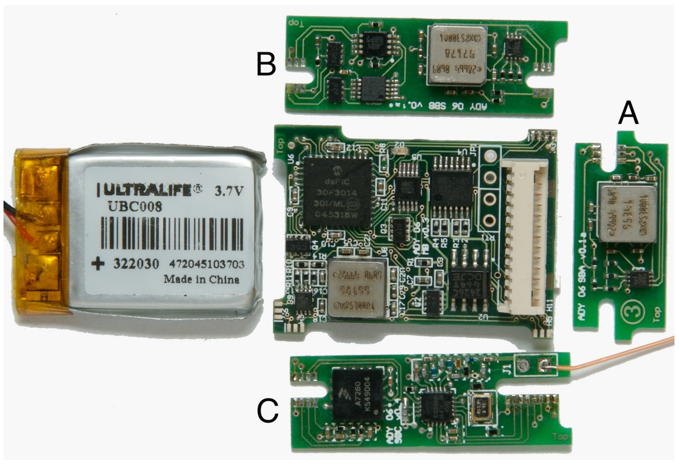


Figure 5.3: *Orient-2* Block Diagram



(a) Complete Device



(b) Component Boards and Battery

Figure 5.4: *Orient-2* Device

Mainboard the largest circuit board in the device. It is a four layer PCB to which the processor; flash memory; power supplies; **X** and **Y** magnetometers, and supporting amplifiers; **Z** rate gyroscope and range selection circuitry; and debugging and interface header are mounted.

Sideboard A the smallest of the sideboards holds the **X** rate gyroscope and range selection circuit. Like the other sideboards Sideboard A is constructed from a two layer PCB.

Sideboard B holds the **Y** rate gyroscope and **Z** magnetometer and supporting amplifier.

Sideboard C holds the three axis accelerometer, and the radio and antenna matching network.

The device is powered by a single cell 120mAh Lithium Polymer battery. This provides a voltage ranging from 4.2 to 2.7V; the nominal voltage is 3.7V. The system has four voltage rails: ground, 3.3V main supply, 1.65V analogue ground, and 5V rate gyroscope supply. The majority of the components run from the 3.3V supply regulated by a low drop-out (LDO) voltage regulator. Additional supplies will be discussed in detail in Section 5.2.4.

The device dimensions were designed around the size of the battery in order to produce the most compact design possible. The battery itself was selected as it was the smallest battery, available in small numbers, deemed suitable of powering the system for a reasonable time. The design went through several iterations during initial PCB layout to fit all of the required components into the available board area.

A challenge in the design of the final device was how to create electrical connections between the various PCBs. Traditional PCB header connectors would have taken up substantial, valuable, board area. In addition it was found to be difficult to source inter-board headers with suitable numbers of connections. To overcome this problem the boards where designed to have interlocking slots which provided ample room to build edge connections directly onto the PCB. After the boards were populated they could be slotted together and the edge connections soldered together to complete the construction. The slotted design of the boards also made it easier to align all the boards into the correct three-dimensional structure.

5.2.2 Processor

The experience of developing the initial prototype had made it clear that at least a 16-bit processor would be required to allow the development of local orientation estimation. The choice of using a 16-bit processor over a 32-bit processor was taken in order to reduce device cost and power consumption. Two competing 16-bit architectures were identified: the Texas Instruments MSP430 family and the Microchip dsPIC family. Both families supported automatic sampling of multiple inputs, single cycle multiplies, and a large range of input/output options.

The processor finally chosen for the device was a Microchip dsPIC 30F3014 [69]. This device is a 16-bit microprocessor with enhanced support for digital signal processing (DSP). The device has 24K of program FLASH, 1K of EEPROM, and 2K of SRAM and can operate up to speeds of 30MIPS.

The dsPIC was selected for its large number of analogue inputs and the powerful DSP core. A single cycle 16-bit multiply-accumulate function and dual 40-bit accumulators make the dsPIC highly suitable for performing fast vector math operations. In comparison to the MSP430, the dsPIC supports more analogue input channels, a useful feature as it allowed for fully automatic sampling of all sensors without the need to switch inputs during sampling.

The processor core is clocked by an internal RC clock at approximately 7.37MHz allowing the device to achieve almost 8 MIPS. A low power oscillator, utilising a 32.768KHz crystal, provides a Real Time Clock (RTC) used for accurate timing of sampling and networking operations. Idle and sleep modes allow power savings to be made when the processor is inactive.

Analogue input is provided by an internal 12-bit ADC. The ADC supports automatic sequential sampling of up to thirteen inputs. Samples are placed into an internal buffer and an interrupt is fired on completion of the scan. Sampling can be performed while the core is in low power idle mode.

An external 32Mbit FLASH memory device was included to allow a single device to log data for standalone applications. Interfacing to digital components, such as the external FLASH and radio transceiver, uses a three-wire Serial Peripheral Interface (SPI) bus. The SPI bus was clocked at the processor core frequency of 7.37MHz allowing rapid data transfer.

Processor General Purpose Input/Output (GPIO) pins are used to provide control signals to external sub-circuits and to provide an interrupt source for the radio.

A three-wire in-circuit programming and debugging interface is provided by the processor. The processor can be programmed using the low cost Microchip ICD2 debugger. This tool allows for the firmware to be debugged in real-time on an active device.

5.2.3 Communications

The *Orient-2* platform supports two communication interfaces. The primary interface is provided by a Chipcon CC1100 [70] low power radio transceiver. Secondary communications are provided by a two-wire serial bus for connection to a host PC.

5.2.3.1 Radio

The CC1100 is a flexible low cost, low power, narrowband radio transceiver designed for use in the 315, 433, 868 and 915MHz frequency bands. The CC1100 is capable of a maximum transmission rate of 500kbps. However, in Europe the maximum transmission rate is limited to 250kbps in the 433MHz band [71].

Initial *Orient-2* designs were based on a Chipcon CC2500 [72] 2.4GHz narrowband transceiver. The CC2500 was chosen over the CC2420 used by the *Orient-1* prototype for its smaller size and lower power requirement. Early testing demonstrated that the CC2500 device suffered greatly from interference with other 2.4GHz systems, such as 802.11 wireless networks and bluetooth. Additionally the 2.4GHz band is greatly attenuated by the human body, limiting its suitability for body area networks. The CC1100 was chosen as it required only minor changes to the antenna matching network and no changes to the existing firmware.

The CC1100 uses a 26MHz crystal oscillator as a reference to derive all radio frequencies. The radio is connected to the processor using a shared SPI bus for data transfer and a single GPIO line to flag interrupts. Additionally an output from the radio is routed to the external clock input of the processor to provide an accurate fast clock if required.

5.2.3.2 Serial

Serial communications, for connection to a PC or other system, use a standard Universal Asynchronous Receiver Transmitter (UART). The maximum transfer rate is limited to 230,400 baud by the processor clock rate. This rate was deemed close enough to the maximum transfer rate of the radio to not act as a substantial bottleneck.

5.2.4 Sensors

The device measures static and dynamic acceleration, magnetic field and rotation rate around three orthogonal axes.

5.2.4.1 Accelerometers

Acceleration is sensed using a Freescale MMA7260Q tri-axial accelerometer [73]. This is a single chip micro-machined device capable of sensing both static acceleration, due to the Earth's gravitational field, and dynamic acceleration, due to movement.

The device operates by capacitive sensing of the force applied to a proof-mass. The device performs conversion to a single-sided voltage output that is ratiometric to the power supply. The only external circuitry required for connection to a processor analogue input is an RC anti-alias filter for each axis output. The device supports selectable output gains allowing for output ranges of $\pm 1.5, 2, 4$ and $6g$.

The accelerometer is packaged in a $6 \times 6 \times 1.45$mm QFN package and is factory calibrated for offset, scaling factor and temperature drift.

The Freescale device was chosen over the STMicro device used in the *Orient-1* prototype as it requires less power, is slightly smaller and has increased sensitivity.

5.2.4.2 Magnetometers

The Earth's magnetic field is measured using a pair of the Honeywell HMC1052 [74] dual-axis magnetometers used in the *Orient-1* prototype. The two magnetometers are mounted on different boards to provide complete three axis sensing. The HMC1052 is a magneto-resistive wheatstone bridge device producing a differential output proportional to the applied magnetic field in the sensitive axis. The HMC1052 has a sensitivity of 1mV/V/gauss. The Earth's magnetic field is typically in the range of ± 0.6gauss [64]. This leads to a voltage output of approximately ± 2mV.

The differential output voltage of the magnetometer was amplified by a single stage high-gain instrument amplifier for input into the ADC. The single stage amplifier configuration limits the maximum dynamic range of the signal input to the ADC due to offset errors in the amplifier stage. A dual stage amplifier design could have reduced this problem but was ruled out due to the extra area required for the additional amplifiers.

The instrument amplifier chosen for the final design was a Texas Instruments INA321 [75]. This part was chosen for its very low input offset of just 200μV. The amplifier stage was configured with a gain of 680 resulting in a voltage range of approximately ± 1.36V at the ADC input. The instrument amplifier is referenced to a precision analogue ground [76] regulated to 1.65V.

A reset circuit [74] is included to allow periodic conditioning of the magneto-resistive sensors. This circuit ensures that the sensor maintains maximum sensitivity.

Subsystem power is under the control of the processor. The amplifiers support a low power standby mode while the supply to the magnetometer bridges are controlled by power transistors. The subsystem has a very fast start up, measured in tens of micro-seconds, allowing it to be turned off in between samples to save power.

5.2.4.3 Rate Gyroscopes

Rotational rate data is gathered using three Analog Devices ADXRS300 [77] rate gyroscopes mounted orthogonally. The ADXRS300 is a 300° per second yaw rotation rate sensor. Device operation is based on the Coriolis force applied to a resonating structure during rotation. The device senses the applied force and produces a corresponding output voltage. The ADXRS300 was chosen as it was the smallest rate gyroscope that could be found at the time of design.

The gyroscopes require a 5V supply generated from the battery using a charge pump regulator. The resonating structure is an electrostatic device requiring a voltage of 14-15V to drive it. This is generated from the 5V supply by an internal charge pump circuit. The 5V supply operates at an efficiency of 60-70%.

Each gyroscope draws approximately 6mA at 5V. Start up time is 35ms, meaning that the gyroscopes must remain active at all times. Combined with the poor efficiency of the 5V supply, this causes the gyroscopes to draw the greatest proportion of the power budget. The 5V supply is under processor control allowing substantial power savings to be made while the device is idle.

The gyroscopes produce an output voltage of 2.5 ± 2.25V. In order to input into the 3.3V ADC the gyroscope output is level shifted by a resistor divider and buffered through a unity gain amplifier.

Following an application note from Analog Devices [78], the gyroscope output range is selectable between ± 300, 600, 900 and $1200^{\circ}/\mathrm{s}$ using an analogue switch to select feedback resistors. The control signals to the switch are translated to the 5V domain using transistors and pull-up resistors.

5.3 *Orient-2* Firmware

5.3.1 Overview

The firmware for the *Orient-2* platform is responsible for controlling all aspects of device operation. The firmware comprises approximately 5000 lines of code and was written in a mixture of C and assembly language. The firmware is common to all devices with node specific data, such as identification number and sensor calibration data, stored in the dsPIC EEPROM.

The decision to implement the system from scratch, rather than use an existing realtime operating system or scheduler, was taken in order to optimise device performance. The most common wireless sensor network operating system at the time of design, TinyOS, does not support the dsPIC. Porting the operating system and integrating use of required dsPIC assembly routines into the build environment would have required significant development time with little immediate benefit in code reuse.

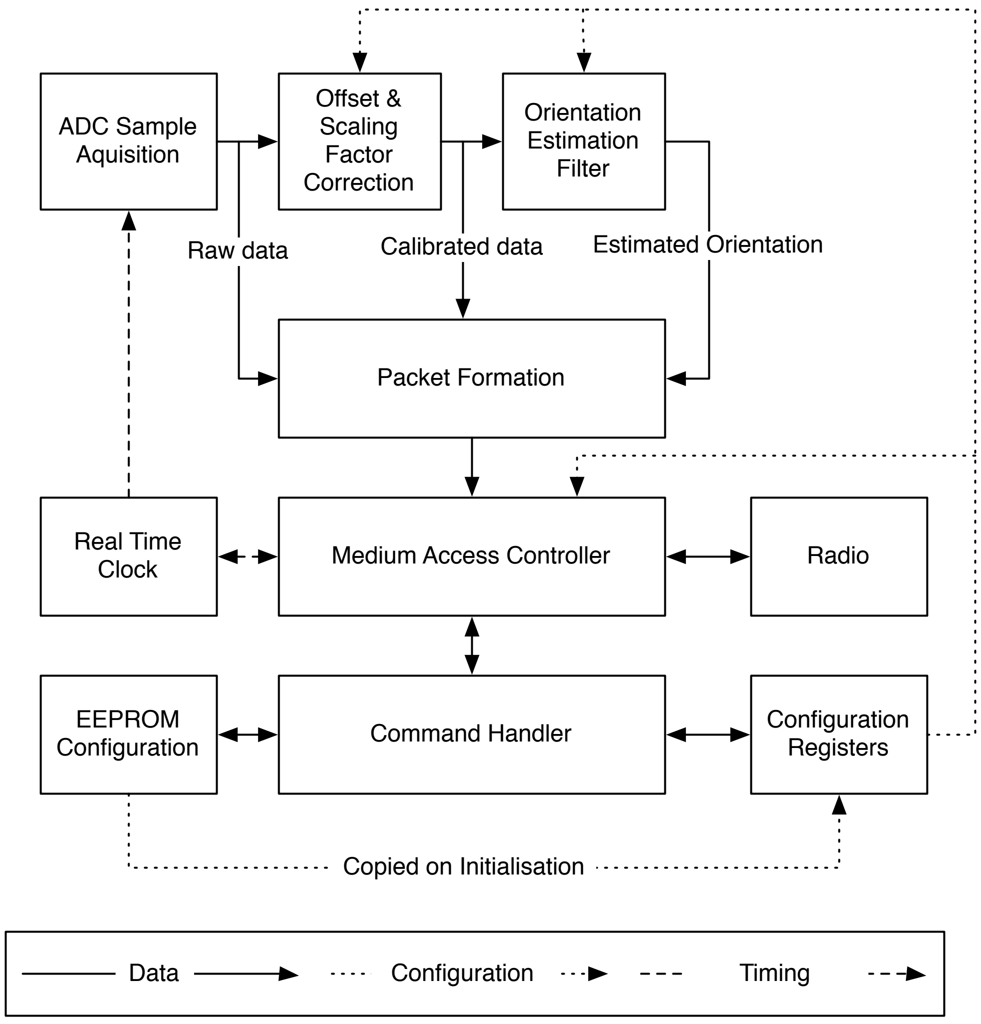


Figure 5.5: *Orient-2* Firmware Overview

An overview of the main firmware components and their interconnections is shown in Figure 5.5.

The operation of the various sub-systems is controlled by a state machine that encodes the major system modes, as outlined in Table 5.1.

The firmware is interrupt driven with Interrupt Service Routines (ISRs) handling immediate processing and raising flags to request background processing of tasks. Tasks are executed by a simple priority scheduler. Once no task flags remain set the system firmware enters a low power state.

Table 5.1: Operating Modes

Mode	Description
Run	Sampling, processing and transmitting.
Flash Log	Sampling and storing raw data to Flash memory.
Flash Readout	Reading samples from Flash, processing and transmitting.
Idle	Awaiting commands with receiver on.
Sleep	Low power sleep with Wake-On-Radio.
Calibrate	Sensors active ready to perform calibration cycle.
Basestation	Bridging radio and serial interfaces; acting as TDMA master.

Table 5.2: Device Settings

Setting	Description
Device ID	Unique identifer for a device
Calibration	Sensor offset and scale calibration values
Output Type	The data outputs used in run modes
Subsample	Filter output subsampling rate used in CSMA run mode
Radio Channel	The radio channel to be used
TDMA Slot	The slot in a TDMA frame to use for transmission
TDMA Slot Time	The length of a TDMA slot
TDMA Frame Length	The number of slots in a TDMA frame

5.3.2 Device Settings and Commands

In addition to the major mode, device operation is controlled by the settings described in Table 5.2. Settings can be configured at run time or saved to the EEPROM to be automatically applied at device start up.

5.3.3 Communications

The *Orient-2* platform supports three radio network modes: Wake on Radio (WoR), Carrier Sense Multiple Access (CSMA) and Time Division Multiple Access (TDMA). Each device can also act as a basestation allowing routing of radio messages to and from the serial port for communication with a PC or other host system.

5.3.3.1 Wake on Radio

WoR mode is supported natively by the CC1100 radio. In this mode the processor is placed in its lowest power sleep mode while the radio periodically samples the radio channel looking for packets. Channel polling is performed automatically using an internal low power oscillator in the CC1100. If channel activity is detected the radio will automatically receive a packet and then wake the processor from sleep. The processor checks the received packet to see if it should handle the packet, based on the packet destination address, and either handles the packet or returns to sleep.

This mode allows the device to be placed in a very low power state while still supporting remote activation. This mode could be used to preserve power during breaks between capture takes.

5.3.3.2 Carrier Sense Multiple Access

CSMA mode is used to configure device settings. In this mode all devices remain in receive mode except while transmitting or handling a packet. This means that there is no requirement for device synchronisation to allow communication but results in high power requirements as the radio is always active. CSMA mode supports both unicast and broadcast transmissions based on the destination address field in the command packet type (see Section 5.3.3.4).

Multiple access is achieved by listening for a clear radio channel before transmission and using a randomised backoff in the event of a busy channel. All command packets in CSMA mode are acknowledged with a pre-backoff based on node ID applied to broadcast acknowledgements to prevent collisions.

CSMA mode can be used for quick data capture from individual devices but is not suitable for capturing data from multiple devices as there is a high probability of packet collision or channel starvation with many devices running using isochronous sampling clocks. Additionally, with no scheduled time for commands, devices would waste precious energy looking for command packets amongst the data packets.

5.3.3.3 Time Division Multiple Access

TDMA overcomes the problems encountered during multiple device capture by providing time synchronisation between nodes.

In TDMA mode, radio channel access is divided into frames, which are further sub-divided into slots. At the start of each frame there is a synchronisation slot in which all devices synchronise their local clocks to the reception of a “sync” packet from the basestation. Subsequent slots are uniquely allocated to devices to allow data transmission.

Time synchronisation presents advantages in terms of power saving, as now device radios

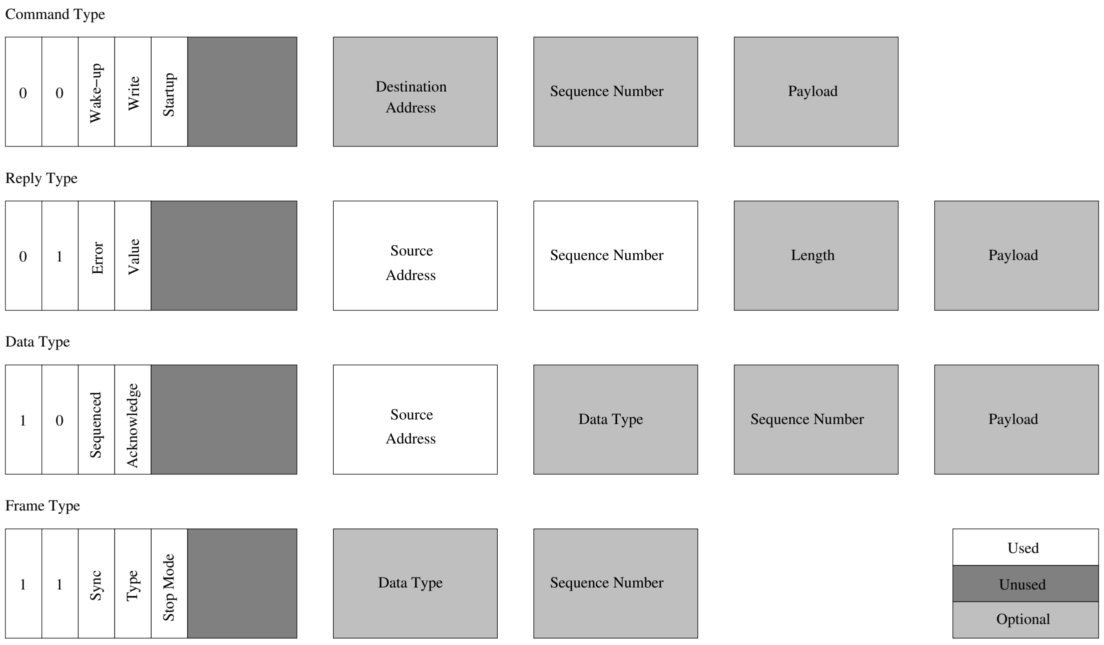


Figure 5.6: Packet Formats

are only active during the synchronisation and transmission slots; fairness, as each device has a unique pre-allocated transmission slot; and guaranteed system latency.

In order to increase the robustness of the network devices may continue to transmit in their pre-allocated slot even if they miss the synchronisation packet. If a device misses too many synchronisation packets it will return to synchronisation mode, and stop transmitting data, until it can re-synchronise with the basestation master signal. The number of synchronisation packets that a device can miss before re-synchronising was determined empirically in order to balance robustness to temporary network packet loss with collisions caused by unsynchronised transmission.

5.3.3.4 Packet Format

A common packet format is used for all communications. The format is designed to minimise header transmission overheads while maintaining sufficient flexibility to cover all required packet types. The packet formatting was altered at various times in the design cycle in order to accommodate new features as their necessity became apparent and to optimise the amount overhead.

Four packet types, illustrated in Figure 5.6, are used:

Command Used to wake a device, or request a read from or write to device setting registers.

Single byte “wake up” packets are used to wake the device from wake-on-radio mode. All other command packets use the address and sequence number bytes.

Device register writes, indicated by setting the write bit, take a setting identifier and a variable number of arguments as a payload. The number of arguments depends on the setting being changed. Register reads require only a single byte setting identifier as a payload.

Each device in the network has a unique identifier that allows commands to be sent to specific devices. Two special identifiers are provided to allow for automatic network discovery: a basestation address and a broadcast address. The basestation address, only used with serial communications, provides a way to communicate with a host connected device without requiring knowledge of its unique identifier. The broadcast address allows a command to be sent to all nodes except the basestation.

Reply Used to acknowledge commands and to return device settings.

Replies send the device identifier of the replying device and the sequence number of the command being replied to. An error bit in the type byte is used to indicate a command failure. If the value bit is set then the optional length and payload bytes are used. The length of the payload is specified by the length byte.

In the case of replies to broadcast commands, devices employ a pre-backoff based on their identifier in order to avoid collisions.

Data Used to send data, such as orientation updates.

All data packets have the device identifier as the source address of the packet. If data is sent in sequenced mode then the optional data type and sequence number bytes are used. The data type is used to specify the type of data in the packet and therefore its length. Data packets can request acknowledgement by setting the acknowledge bit.

The *Orient-2* firmware supports several different data types. These are: raw sensor data, calibrated sensor data, estimated orientation, and estimated dynamic acceleration vector.

Frame Used in TDMA mode to synchronise devices.

Single byte "sync" radio packets are used to synchronise devices in TDMA mode. Frame packets are also sent to the serial port and include the sequence number of the following data packets and their data type.

Frame packets support a simple command strobe for exiting TDMA mode. This is indicated by clearing the sync bit. Two options are available: idle and sleep. When idle is selected the device returns to its idle mode and can receive CSMA packets. Sleep mode puts the device to sleep, ready to quickly resume TDMA mode with the current settings.

In addition all packets have a two byte preamble, two byte Start of Frame Delimiter (SFD), one byte length and two byte Cyclic Redundancy Check (CRC) provided by the CC1100 packet engine.

5.3.4 Orientation Estimation

Orientation estimation is performed locally on the device using the direct conversion filter described in Section 3.3.6.

Input sampling is triggered by the RTC at a rate of 256Hz. Approximately 250µs before sampling the magnetometers are activated. The processor automatically scans the sensors in quick succession interrupting on completion. The ADC ISR moves data to main memory, shuts down the magnetometers, sets a scheduler flag indicating new data available and turns off the ADC.

The scheduler detects that new sensor data is available and executes a task to process the data. The sensor data is first corrected for offset and scaling errors using device specific calibration data. The calibrated data is then processed by the orientation estimation filter.

The filter implementation makes extensive use of the dsPIC's DSP ability to perform fast multiplies and accumulations. All orientation calculations were performed using fixed-point numbers in a 2.14 format. This format provides a range of $-2 \rightarrow \left(2 - \frac{1}{2^{14}}\right)$ and a precision of $\frac{1}{2^{14}}$.

In order to minimise processing delays in orientation estimation various key functions were written in assembly code. These include functions implementing quaternion multiplication and a fast square root algorithm. The digital signal processing ability of the dsPIC was used to optimise these functions. The ability to perform fast square roots is key as four are required in the existing filter implementation: two to normalise the measured basis vectors to compute an estimated rotation matrix, one in matrix to quaternion conversion and one in quaternion normalisation. The provided square root function in the dsPIC library is implemented in floating point and requires approximately 650 cycles to compute the square root of a 2.14 fixed point value. A carefully written assembly function was designed that could perform the same operation in 93 cycles.

5.4 Host PC Software

In addition to the *Orient-2* hardware and firmware, software was written to interface the devices to a host PC in order to remotely control devices and to visualise their output. Two interfaces were written: a *Python* module and a *Java* application.

5.4.1 *Python* Software

The *Python* module was written early during the implementation testing to provide a simple way to communicate with test devices. The module allowed for both interactive control and scripted testing. The use of *Python* helped to reduce development time as communication protocols changed.

The *Python* module is composed of two classes. A low level class, *OrientComms*, handles sending pre-constructed packets to the serial port and receiving and reconstructing packets from the serial port. A higher level class, *OrientControl*, provides methods to read and write device registers, both singly and in bursts, and to initialise TDMA data capture operations. The *OrientControl* class uses two callback functions to allow easy development of extensions using data from the *Orient* devices.

A calibration script was built on top of the *Python* module that talks the user through a simple calibration process to calculate appropriate offset and scaling factors for a device. The exact calibration procedure will be discussed further in Section 6.2.

5.4.2 MotionViewer Software

The construction of a three dimensional body model from sensed orientation data is performed on a PC. The 'MotionViewer' software is written in *Java* and has been tested on Windows, Linux, and Macintosh platforms. An overview of software components is shown in Figure 5.7.

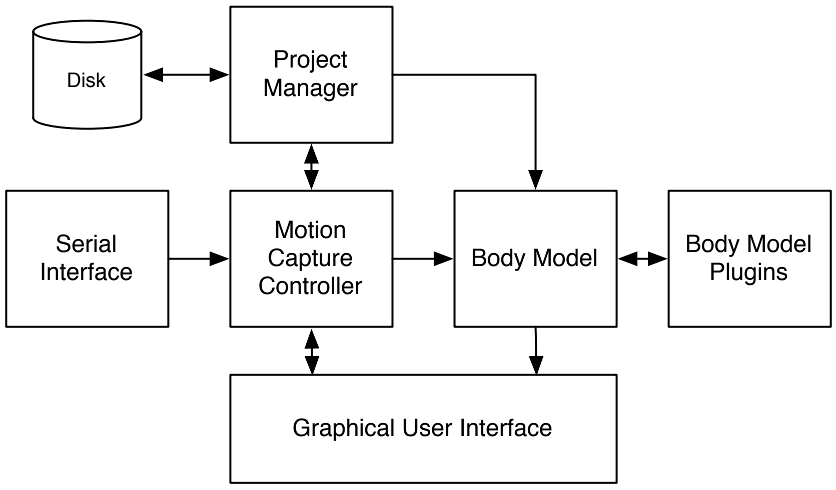


Figure 5.7: MotionViewer Block Diagram

5.4.2.1 Serial Interface

In early versions of the software, device communications were conducted through the *Python* module as an intermediate. This allowed for alterations in communications protocol without requiring alteration of the more complex *Java* application. The two programs were connected by the use of UNIX pipes to allow ASCII data to be communicated.

Once the communications protocol had stabilised a *Java* implementation of the communications protocol was written using the RXTX implementation of the Java Communications API. The RXTX implementation was chosen for its availability on multiple operating systems.

Early versions of the serial communications module required manual entry of basestation and device identifiers. In order to reduce system setup time, and increase ease of use, an automatic discovery mechanism was implemented. The discovery process involves two steps:

1. Communications are established with the device connected to the host's serial port. Using the special basestation identifier built into the packet format allows communications with the basestation device regardless of its unique identifier.

2. A broadcast command is sent to all remote devices asking for them to return their identifiers. The remote devices each reply after a pre-backoff based on their identifier. In this manner a list of remote devices can be constructed.

In addition to automatic device discovery the serial communications interface supports automatic TDMA scheduling. A method is provided that will generate a schedule based on a list of active devices and an output data type. Empirical measurements of packet timing requirements allow an optimal schedule to be automatically devised that minimises the network latency and maximises the update rate.

While the serial interface implementation handles transmission of commands and acknowledgements it does not handle the decoding of data packets. Instead it provides an interface that allows other classes to handle data packets as they are received. This allows the serial interface to be used in different applications that may want to handle data in differing ways. A reference data handler is included in the package that shows how to decode that packet into standard types from the *javax.vecmath* package.

5.4.2.2 Motion Capture Controller

The motion capture controller is responsible for managing the playback and recording of motion capture data. It includes a simple state machine that controls whether the application should use live data capture or replay data contained within an existing project.

The motion capture controller implements *Orient* packet decoding and makes data available to the body model for processing. The packet decoding implementation is common to both live and recorded data.

In addition to handling motion data decoding the motion capture controller class provides two important mappings:

1. Sensor-Joint Map

   Provides a mapping between sensor identification numbers and *Joint* objects in the body model.

   Mappings may be changed between captures and are saved with motion capture data for later replay. The mapping is stored along with the motion data as metadata (see Section 5.4.2.3) and can be updated at any time.

2. Sensor-Alignment Map

   Provides a mapping between sensor identification numbers and alignment quaternions specifying the rotation between the device local co-ordinate frame and the joint local co-ordinate frame.

   Alignment calibration is performed by having the user stand in a calibration stance. The calibration stance is defined by the stance of the body model when all joints are in their default orientations. With the user in the calibration stance the orientation of the sensor devices is measured. The alignment between the sensor and joint is then the difference between the joint orientation and the sensor orientation, i.e. the inverse of the measured sensor orientation. As all joints are assumed to be in their default orientation the mapping of sensor identifiers to joints is not important during the alignment calibration process. Provided the physical attachment of sensors to limbs is unaltered, the sensor mapping may be freely changed without requiring repetition of the alignment calibration process.

5.4.2.3 Project Manager

The project manager allows the saving of motion capture data to disk for playback at a later time.

Motion capture data is stored in binary files; the data is stored in the same format as that in which it is presented by the serial interface. The advantage of using a common format is that the same packet decoding code can be used for both live and recorded data. Storing the motion data as it comes from the device, rather than the resulting motion of the body model, allows corrections to be made to sensor-joint mappings and sensor alignments after capture takes place.

In addition to the binary motion data the project manager uses XML files to record capture metadata. The metadata contains information such as: the time of the capture, the number of sample frames, the frame rate, sensor alignments, and the mapping between *Orient* device identifiers and body model joints. The combination of the metadata and the binary motion data is enough to perform exact replays of recorded capture sessions.

MotionViewer projects are stored as compressed ZIP files. Each project contains a BioVision Hierarchy (BVH) file specifying the joints of the body model and its default stance, and a collection of motion data files and corresponding metadata. The decision to use a ZIP file was made as it, in addition to reducing the required disk space, allows the project to be treated by the user as a single discrete file.

5.4.2.4 Body Model

The body model is represented as a collection of *Joint* objects stored in a tree structure. Each *Joint* object has an associated co-ordinate frame. The position of a child *Joint* object is given by a relative offset from the parent in the parent's local co-ordinate frame. This allows the construction of groups of joints attached to a single parent. This is useful in construction of complex bone structures such as the pelvis, where the left and right hip joints are both offset from a common point at the base of the spine. This model has the additional advantage that it simplifies export to existing animation formats such as BVH.

As joints are specified by an offset from their parent joint the model has a default stance when all joint orientations are set to their default. This stance is typically chosen to be a standing posture with the arms outstretched, as shown previously in Figure 3.4. This stance is chosen as it is the standard stance used for existing animation work flows. However, apart from ease of integration with existing tools, there is nothing special about this stance; any stance that the user is comfortable to hold can be used for calibration.

The full posture of the body model is updated at the end of each TDMA frame when data has been received from all devices. The body model tree is traversed in a pre-order depth-first manner. The position of a *Joint* is calculated using the forward kinematic process discussed in Section 3.2.1.

The body model design chosen is generic and is capable of describing any set of articulated rigid bodies. It is therefore not limited to human motion tracking applications. Models can be imported from existing BVH files, allowing the use of highly detailed models produced in other tools.

5.4.2.5 Body Model Plug-ins

Interaction with the body model is achieved through the use of classes extending the abstract class *FrameUpdateTask*. At the end of each capture frame the set of update tasks is iterated through, allowing each one to update its state. Each task has access to the complete body model and can modify this in addition to its own internal state.

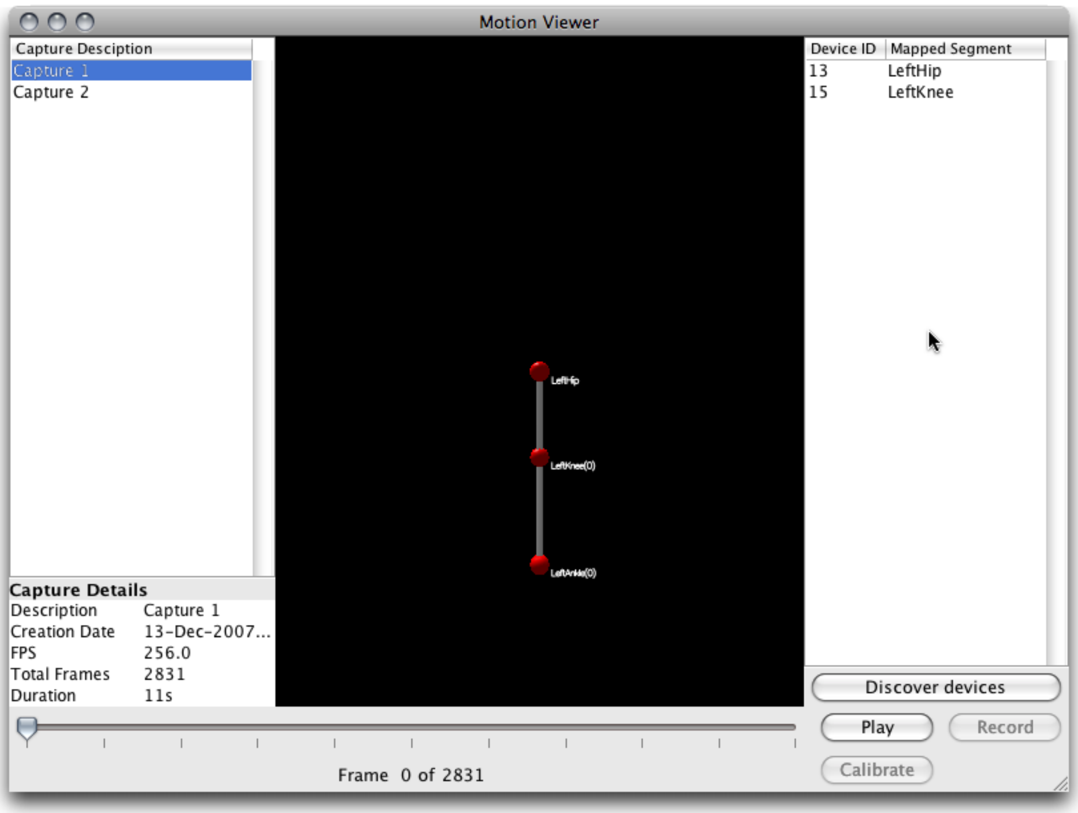
Example frame update tasks are:

Visualisation Updates the state of the 3D visualisation.

BVH Output Outputs the current state of the body model in BVH format suitable for import into other animation software.

5.4.2.6 Graphical User Interface

A graphical user interface, shown in Figure 5.8, allows visualisation and control of the system.



Capture Desciption
Capture 1
Capture 2

Capture Details

Description	Capture 1
Creation Date	13-Dec-2007...
FPS	256.0
Total Frames	2831
Duration	11s

Device ID	Mapped Segment
13	LeftHip
15	LeftKnee

Frame 0 of 2831

Figure 5.8: MotionViewer Interface

The body model state can be viewed in realtime using an OpenGL based 3D visualisation tool. The visualiser is triple buffered allowing it to be updated at an independent rate from that of the motion capture system. The visualisation of the body model is scalable allowing arbitrary units to be used in the internal model while still fitting within the standard camera view.

User interface elements allow the user to control mapping of the sensors, start and stop motion capture, perform stance calibration and to replay pre-recorded motion capture files.

Chapter 6

Calibration & Accuracy

6.1 Introduction

The performance of the *Orient* motion tracking system as a whole depends on the ability of an individual device to accurately estimate its orientation with respect to the fixed world co-ordinate frame. This chapter presents an experimental evaluation of the *Orient* platform. A calibration procedure is discussed, and experiments to evaluate static accuracy and dynamic accuracy are presented.

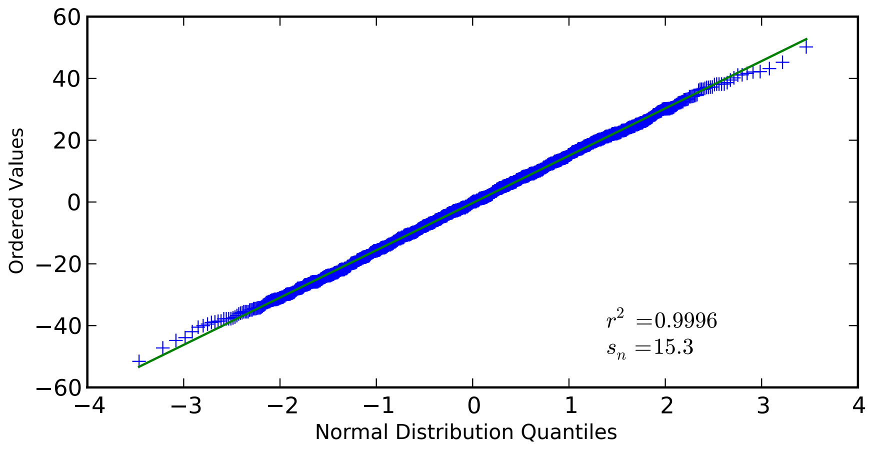
6.2 Sensor Calibration

The estimation of orientation from raw sensed data requires that the data be normalised to correct for variations between each sensed axis. These variations, caused by tolerances in device components, must be corrected in order to produce undistorted three-dimensional data. As the sensors used have linear response to inputs, this amounts to experimentally determining suitable offset and scaling factors to apply to each sensed axis.

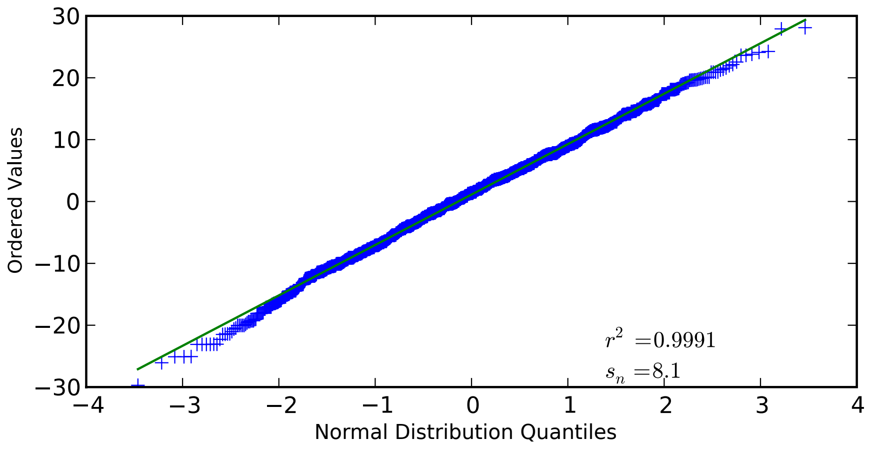
6.2.1 Sensor Characteristics

To estimate the number of samples required to accurately estimate the mean ADC readings for each orientation an initial experiment was performed with ten devices. 5000 samples of raw sensor data were captured for each sensor axis of each device at the native sample rate of 256Hz with the device at rest on a table. The temperature of the room was measured to be $20 \pm 1°$C.

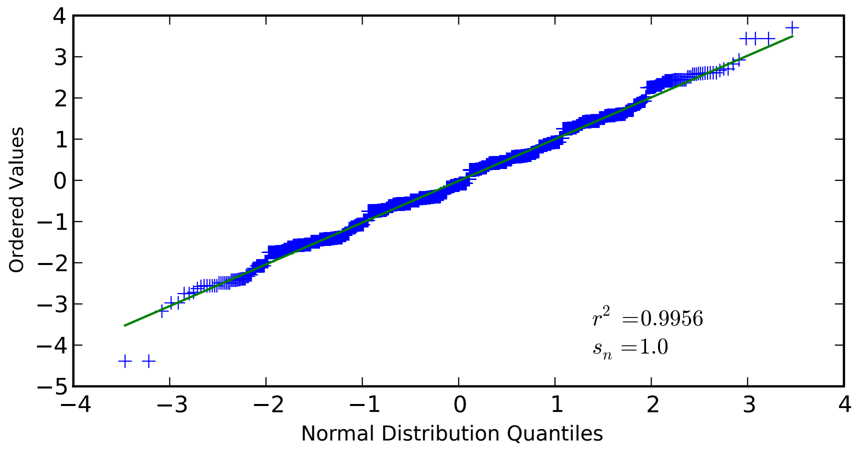
It was expected that the noise in each sensor should follow a normal distribution. To confirm this, quantile-quantile probability plots were generated as shown in Figure 6.1. The mean value for each axis of the accelerometer and the magnetometer data varies depending on the orientation of the device. Each sensed axis will also have a unique offset error. Therefore the raw sample data was normalised by removing the sample mean from each axis.



(a) Raw Accelerometer Data



(b) Raw Magnetometer Data



(c) Raw Gyroscope Data

Figure 6.1: Normal Quantile-Quantile Probability Plots

Sensor	**Noise s_n (ADC points)**
Accelerometers	15.3
Magnetometers	8.1
Rate Gyroscopes	1.0

Table 6.1: Sensor Noise Floor

The high regression fit parameter, r^2, for each quantile-quantile plot indicates that the sensor noise is well modelled by a normal distribution. The sample standard deviations of the sensor noise can be determined from the slope of the plots and are shown in Table 6.1.

As discussed in Section 3.3 each of the sensed values can be expressed by the general formula:

$$\vec{v} = S\left(\vec{v_M} + \vec{O} + \vec{n}\right), \tag{6.1}$$

where $\vec{v}$ is the vector quantity being measured, S is a diagonal scaling factor matrix, $\vec{v_M}$ is the sensor output vector, $\vec{O}$ is an offset vector and $\vec{n}$ is a random noise vector. In order to calibrate a device it is therefore necessary to estimate these parameters for each sensor type.

6.2.1.1 Accelerometer Calibration

The accelerometers are scaled such that the magnitude of the measured acceleration is $1g$ for the device in a static orientation. To achieve this the device is placed in six static orientations: z-axis down, z-axis up, y-axis down, y-axis up, x-axis down and x-axis up. In each orientation the raw ADC value for the vertical axis is averaged to estimate the mean value. The offset and scale factors are calculated for each axis using:

$$S_i = \frac{2}{\max_i - \min_i} \tag{6.2a}$$

$$O_i = \frac{\max_i + \min_i}{2}. \tag{6.2b}$$

As the sensor values are corrupted with noise it is necessary to average over a number of samples to estimate the mean value. The standard deviation of the resulting population mean is defined by:

$$SE = \frac{\sigma}{\sqrt{N}}, \tag{6.3}$$

where SE is the standard error, σ is standard deviation of the population and N is the number of samples. The minimum number of samples required, for a given confidence level, α, and error, δ, is [79]:

$$N_z \geq z^2_{\alpha/2}\left(\frac{\sigma}{\delta}\right)^2. \tag{6.4}$$

The true value of σ is unknown but the sample standard deviation, s_n, can be substituted. In this case the number of samples required can be estimated using the Student-t distribution with $N_z - 1$ degrees of freedom:

$$N \geq t^2_{\alpha/2,N_z-1}\left(\frac{s_n}{\delta}\right)^2. \tag{6.5}$$

For the accelerometers to achieve a population mean with 95% confidence that the error is less than 1 ADC point requires:

$$\begin{aligned} N_z &\geq z^2_{0.05/2}\left(\frac{\sigma}{\delta}\right)^2 = z^2_{0.025}\cdot 15.3^2 = 900 \text{ samples} \\ N &\geq t^2_{0.05/2,N_z-1}\left(\frac{s_n}{\delta}\right)^2 = t^2_{0.025,899}\cdot 15.3^2 = 902 \text{ samples.} \end{aligned} \tag{6.6}$$

As the noise standard deviation for the other sensors is less than that of the accelerometers 902 samples is enough to be confident that the samples means are accurate to within 1 ADC point. As native sampling rate of the *Orient* is 256Hz the minimum sampling time is $902/256 = 3.53$s. The sampling time used in the calibration tool was set to five seconds equivalent to 1280 samples.

The accelerometer used in the *Orient* has a sensitivity of $800\text{mV}/g \pm 7.5\%$ [73] for a $1.5g$ range. The output of the accelerometer is passed though a low pass filter with 0dB gain at 0Hz before being sampled by a 12-bit ADC. The supply voltage of the system is regulated to $3.3\text{V} \pm 2\%$. The relationship between ADC points and g is therefore:

$$\begin{aligned} 1 \text{ ADC point} &= \frac{0.8\times 3.3}{2^{12}} \pm \sqrt{7.5^2+2^2} \\ &= 1\text{m}g \pm 8\%. \end{aligned} \tag{6.7}$$

Therefore we have a 95% confidence that the standard deviation, $\delta\bar{x}$, of the measured minimum and maximum values due to sensor noise is less than $1\text{m}g \pm 8\% \equiv 0.1\%$of full scale.

In order to measure the minimum and maximum accelerometer value for each axis it is necessary to align the sensor with the local gravity vector. This alignment, particularly when calibrating by hand, will also be subject to error. An ideal single-axis accelerometer when rotated around an axis orthogonal to gravity will produce an output:

$$a = \cos(\theta), \tag{6.8}$$

where θ is the rotation angle. The error introduced by a small alignment error, $\delta\theta$ is therefore:

$$\delta a = \left|\frac{\partial}{\partial\theta}\right|\delta\theta = |-\sin(\theta)|\,\delta\theta, \tag{6.9}$$

provided $\delta\theta$ is sufficiently small. Evaluating Equation 6.9 at $\theta = 0 \pm \delta\theta$ indicates that no error is propagated. In practice $\delta\theta < 5^\circ$ results in a maximum error of 0.4%, equivalent to a standard deviation of 0.13%. This level of accuracy can be achieved easily using a standard spirit level.

The errors due to accelerometer alignment and population mean estimation are independent. The errors in the minimum and maximum accelerometer measurements are therefore:

$$\delta\min,\delta\max = \sqrt{\delta\bar{x}^2+\delta a^2} = \sqrt{0.001^2+0.0013^2} \approx 0.0016 \equiv 0.16\%. \tag{6.10}$$

The scale error, δS_i, for each axis is:

$$\begin{aligned}\delta S_i &= \sqrt{\left(\left|\frac{\partial S_i}{\partial \max_i}\right|\delta\max_i\right)^2+\left(\left|\frac{\partial S_i}{\partial \min_i}\right|\delta\min_i\right)^2}\\ &= \sqrt{\left(\frac{2}{(max-min)^2}\right)^2\left(\delta\max_i^2+\delta\min_i^2\right)}\\ &= \sqrt{\frac{1}{4}\left(\delta\max_i^2+\delta\min_i^2\right)}\\ &= 0.1\%,\end{aligned} \tag{6.11}$$

similarly, the offset error, δO_i, is:

$$\begin{aligned}\delta O_i &= \sqrt{\left(\left|\frac{\partial O_i}{\partial \max_i}\right|\delta\max_i\right)^2+\left(\left|\frac{\partial O_i}{\partial \min_i}\right|\delta\min_i\right)^2}\\ &= 0.1\%.\end{aligned} \tag{6.12}$$

The effect of accelerometer scale and offset errors is to cause an error in the estimated gravity vector. This orientation error can be defined as the error angle between the true gravity vector, $\vec{g}_{ref}$, and the measured vector, $\vec{g}_{meas}$:

$$\theta_\varepsilon = \cos^{-1}\left(\frac{\vec{g}_{ref}\cdot\vec{g}_{meas}}{\left\|\vec{g}_{ref}\right\|\cdot\left\|\vec{g}_{meas}\right\|}\right). \tag{6.13}$$

The error is also dependent on the orientation of the device. This dependence occurs as the gravity vector is measured by a combination of device axes.

A Monte Carlo simulation was performed to generate the expected error distribution. One thousand simulations were performed with the scale and offset errors normally distributed with $\sigma = 0.001$. For each simulation the orientation of the device was parameterised by two angles: $\phi \in [-45°,-44°,\ldots,44°,45°]$ and $\theta \in [-45°,-44°,\ldots,44°,45°]$. The resulting estimated CDF is shown in Figure 6.2. It can been seen that 90% of the resulting errors are less than $0.15°$.

6.2.1.2 Magnetometer Calibration

As with the accelerometers, suitable scale and offset must be determined for the magnetometers. The calibration is complicated by the fact that the exact direction of the magnetic field vector is hard to estimate due to the inclination angle. Measuring the minimum and maximum values for each axis is not simple.

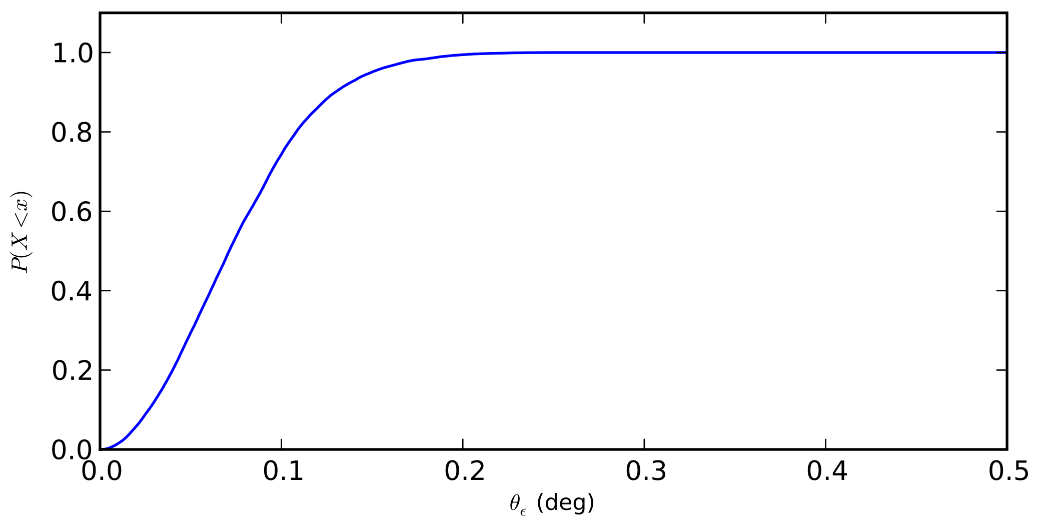


Figure 6.2: Estimated Effect of Accelerometer Scale and Offset Errors

The magnitude of the magnetic field vector is not important for the vector observation algorithm of Section 4.1.2.2. While the scale and offset equations, (6.2), were defined in terms of the minimum and maximum values for an axis, it is only necessary that the two values be symmetric about the null point. Calibration of the magnetometers is therefore performed at the same time as the accelerometers by aligning one of the horizontal device axes with the local magnetic north as measured by a normal magnetic compass. The full set of six static calibration positions are therefore: z-axis down, x-axis north; z-axis up, x-axis south; y-axis down, z-axis north; y-axis up, z-axis south; x-axis down, y-axis north; and x-axis up, y-axis south.

As the magnetometers have a lower noise deviation than the accelerometers the mean measured value for each calibration should have a deviation well within 1 ADC point. Repeating the calculations of Equation (6.6) for the sample standard deviation $s_n = 8.1$ results in 95% confidence that the mean value for the magnetometer is within 0.5 ADC points after 1011 samples.

The magnetometers have a sensitivity of $1\text{mV}/\text{V}/\text{gauss} \pm 20\%$ [74], this signal is amplified by a precision instrumentation amplifier with a gain of [75]:

$$G = 5 + 5\left(\frac{R_2}{R_1}\right) \pm 0.1\%, \tag{6.14}$$

where $R_1 = 200\Omega \pm 1\%$ and $R_2 = 15\text{k}\Omega \pm 1\%$. As before the voltage supply is $3.3\text{V} \pm 2\%$ and the ADC has a precision of 12-bits. The relationship between ADC points and gauss is therefore:

$$\begin{aligned} 1 \text{ ADC point} &= \frac{3.3}{1 \times 10^{-3} \times G \times 3.3 \times 2^{12}} \pm \sqrt{1^2 + 1^2 + 0.1^2 + 2^2 + 20^2} \\ &= 642\mu\text{gauss} \pm 20\%. \end{aligned} \tag{6.15}$$

The Earth's magnetic field varies between 24μT and 60μT [1] where 1gauss = 100μT. A 0.5 ADC point deviation therefore represents a worst case deviation of:

$$\delta\bar{x} \leq \frac{0.5 \times 642 \times 10^{-6} \times 1.2}{0.24} \leq 0.16\%\text{Full Scale.} \tag{6.16}$$

The magnitude of the magnetic field in the vertical and horizontal directions are given by:

$$m_H = \cos(I)\,\|m\| \tag{6.17a}$$

$$m_V = \sin(I)\,\|m\|\,, \tag{6.17b}$$

where I is the inclination angle of the field measured from the horizontal. In Edinburgh the inclination angle is approximately 70° and so the downward facing axis has the greatest value representing approximately 94% of the true magnitude. In areas where the inclination angle is less than 45° the horizontal component should be measured.

If we again assume that the calibration surface is within 5° of level then the standard deviation in measuring the vertical component of the magnetic field is 0.13%.

As with the accelerometer the deviations in the minimum and maximum values can be calculated as:

$$\delta\text{min}, \delta\text{max} = \sqrt{\delta\bar{x}^2 + \delta a^2} = \sqrt{0.0016^2 + 0.0013^2} \approx 0.002 \equiv 0.2\%, \tag{6.18}$$

and the resulting scale and offset deviations calculated:

$$\begin{aligned}\delta S_i &= \sqrt{\left(\left|\frac{\partial S_i}{\partial\text{max}_i}\right|\delta\text{max}_i\right)^2 + \left(\left|\frac{\partial S_i}{\partial\text{min}_i}\right|\delta\text{min}_i\right)^2} \\ &= 0.14\%,\end{aligned} \tag{6.19}$$

$$\begin{aligned}\delta O_i &= \sqrt{\left(\left|\frac{\partial O_i}{\partial\text{max}_i}\right|\delta\text{max}_i\right)^2 + \left(\left|\frac{\partial O_i}{\partial\text{min}_i}\right|\delta\text{min}_i\right)^2} \\ &= 0.14\%.\end{aligned} \tag{6.20}$$

6.2.1.3 Gyroscope Calibration

The calibration of the rate gyroscope is significantly different to that of the other sensors.

The offset is estimated directly from the average of the mean gyroscope outputs, $\bar{\omega}_i$, during the six static calibrations:

$$O_i = \sum_{j=0}^{5} \bar{\omega}_{i_j}/6. \tag{6.21}$$

With the low noise present in the gyroscope signal just 387 samples are required to obtain a 95% confidence of an offset error less than 0.1 ADC point. Although there are many more samples available, allowing for a better estimate of the mean offset, the device is not capable of representing these due to the limitations of fixed point arithmetic.

The rate gyroscopes are normally operated with an extended range of $\pm 600^\circ/\mathrm{s}$ [78]. The sensitivity of the gyroscopes is $2.5\mathrm{mV}/^\circ/\mathrm{s} \pm 8\%$ [77]. The gyroscope signal is passed through a resistor divider, composed of two 1% resistors in a 3/5 ratio, and unity gain buffer to convert to the 3.3V domain of the processor. One ADC point therefore represents a rotational rate of:

$$\begin{aligned} 1 \text{ ADC point} &= \frac{3.3}{1.5 \times 10^{-3} \times 2^{12}} \pm \sqrt{8^2 + 2^2} \\ &= 0.53^\circ/\mathrm{s} \pm 8\%. \end{aligned} \tag{6.22}$$

The offset error, therefore, is expected to have a standard deviation:

$$\delta O_i = 0.1 \times 0.53 = 0.05^\circ/\mathrm{s}. \tag{6.23}$$

The scale factor is calculated by rotating the device through a known orientation and comparing the integral of the offset corrected gyroscope data with the known angle, θ. The scale factor should result in the gyroscope measurement being in half rad/timestep. Therefore:

$$S_i = \frac{\theta}{2\int_{t=0}^{T} (\omega_i - O_i)\, dt} = \frac{\theta}{2\hat{\theta}}. \tag{6.24}$$

As the offset is not perfectly known there will be an error in the integral:

$$\int_{t=0}^{T} (\hat{\omega}_i + \delta O_i)\, dt = \hat{\theta} + T\delta O_i. \tag{6.25}$$

An additional error will be present in the integral as it is evaluated numerically. The bound on the numerical integration error depends on the integration method used and the derivatives of the integrand. The current implementation uses the rectangle rule for integration resulting in an error directly proportional to the sample period.

The integration error at each sample, E_i, is the error between the true integral and the estimated integral as illustrated in Figure 6.3. Based on the Taylor expansion of the integrand, the integration error may be approximated to a first order as the area of a triangle:

$$\begin{aligned} E_i &\approx \frac{1}{2}\left(f(x_{i+1}) - f(x)\right)\Delta t \\ &\approx \frac{1}{2} f'(x_i)\Delta t^2, \end{aligned} \tag{6.26}$$

where $f'(x)$ is the first derivative of $f(x)$, and Δt is the sampling period.

The resulting worst case error, E, in the definite integral:

$$\int_a^b f(x)dx, \tag{6.27}$$

is given by:

$$|E| \leq \frac{1}{2} M (b-a)\Delta t \quad \left|f'(x)\right| \leq M \forall x \in [a,b]. \tag{6.28}$$

This worst case error only occurs when $f'(x) = M \forall x \in [a,b]$. In the specific case of gyroscope integration the angular acceleration, equivalent to $f'(x)$, is necessarily zero mean as the angular

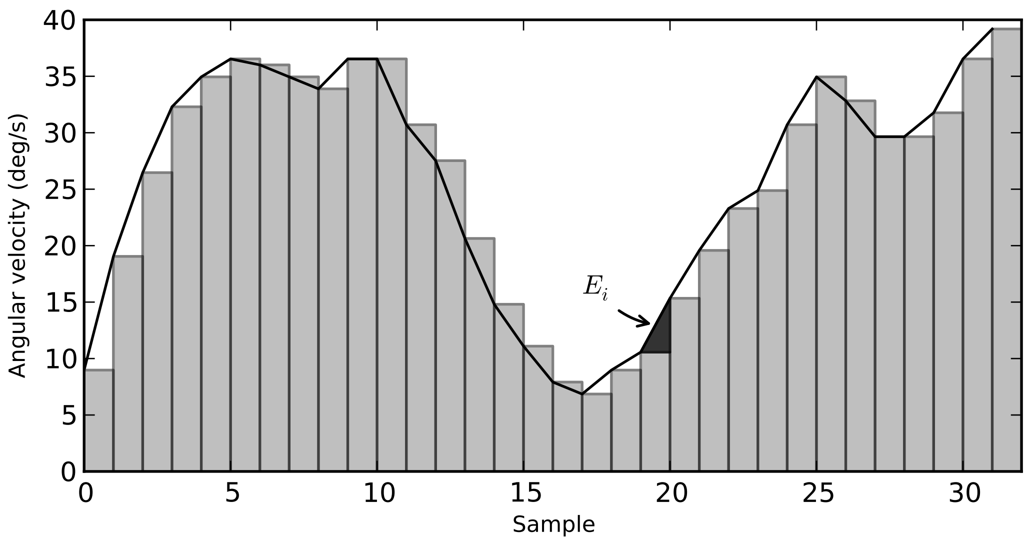


Figure 6.3: Integration error with rectangle rule for numerical integration

velocity at the start and end of calibration is zero. It is clear, therefore, that cancellation of errors will occur.

In order to model the gyroscope integration error an initial experiment was performed to capture rate gyroscope data during typical calibration rotations. Raw gyroscope data was gathered as a device, placed flat on a desk, was rotated about its z-axis. First the gyroscope offset was estimated by taking the mean gyroscope output from the device at rest. The gyroscope signal was converted to degrees per second by first subtracting the estimated offset and then multiplying the signal by the nominal scale factor of 0.53 degrees per ADC point. The angular acceleration was then estimated from the rotational rate signal by numerical differentiation using a first central difference approximation.

The angular acceleration distribution and autocorrelation, generated from a single experiment, are illustrated in Figure 6.4 and Figure 6.5 respectively. Figure 6.4 indicates that the angular acceleration during the calibration has an approximately normal distribution. The deviations from the reference line indicate that the actual distribution has heavier tails than the normal distribution indicating an increased probability of large values. The single high spike in the autocorrelation plot indicates that the angular acceleration has little dependency between samples. These properties were apparent for repetitions of the initial experiment. It was therefore concluded that the angular acceleration could be approximated by a gaussian white noise process. The mean standard deviation of the angular acceleration, over fifteen tests, was $1000^\circ/\mathrm{s}^2$.

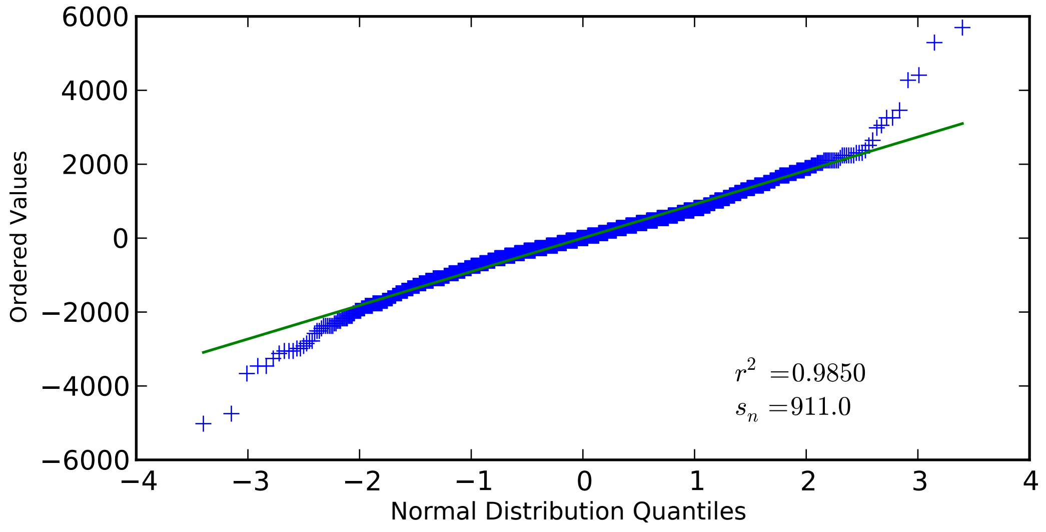


Figure 6.4: Normal quantile-quantile plot of angular acceleration during gyroscope calibration.

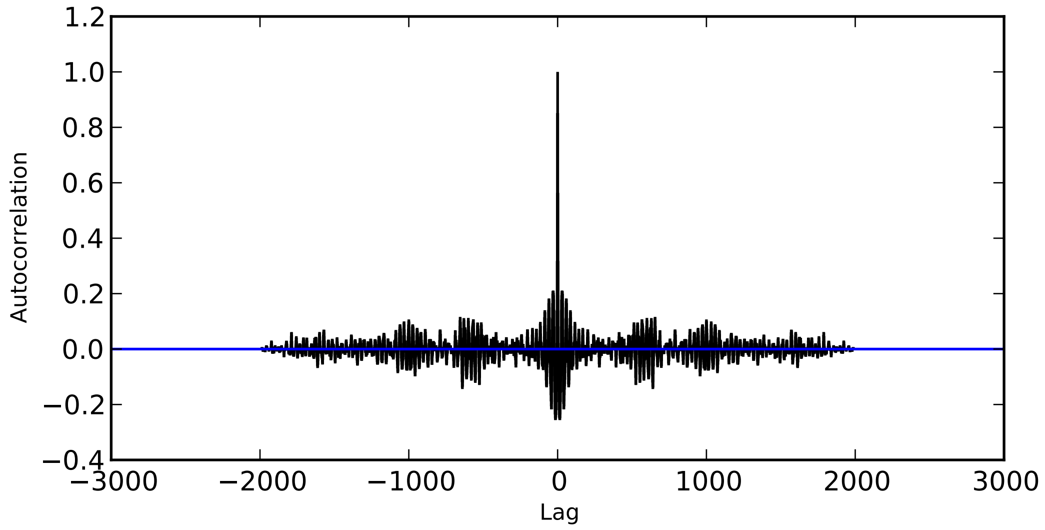


Figure 6.5: Autocorrelation of angular acceleration during gyroscope calibration.

The expected numerical integration error during the calibration, δI, is:

$$\begin{aligned}\delta I &= \sum_{i=0}^{N} \frac{1}{2} f'(x_i)\Delta t^2 \\ &= \frac{1}{2}\Delta t^2 \sum_{i=0}^{N} \alpha_i, \quad \alpha = \mathcal{N}(0, 1000) \\ &= 0 \pm \frac{1}{2}\Delta t^2 \sqrt{N \times 1000^2}, \end{aligned} \tag{6.29}$$

where N is the number of samples integrated.

The fractional error in S_i is therefore composed of the fractional error in the rotation by the user, $\delta\theta$, the integrated offset error, $T\delta O_i$, and the numerical integration error, δI:

$$\frac{\delta S_i}{S_i} = \sqrt{\left(\frac{\delta\theta}{\theta}\right)^2 + \left(\frac{T\delta O_i}{\theta}\right)^2 + \left(\frac{\delta I}{\theta}\right)^2}. \tag{6.30}$$

The rotation angle, θ, was chosen to be 360°, as this allows the device to be rotated until realigned with a starting mark. The error in rotation is assumed to have a standard deviation of 2° for a fractional accuracy of 0.6%. A 2° standard deviation in the error seems a reasonable figure for careful hand calibration where the device can be aligned with an obvious feature such as the edge of a table. The time given to perform the rotation was set at 8 seconds as this provided a reasonable time to carefully perform the rotation. The resultant scale error is therefore:

$$\begin{aligned}\delta S_i &= \sqrt{\left(\frac{2}{360}\right)^2 + \left(\frac{8 \times 0.05}{360}\right)^2 + \left(\frac{\frac{1}{2}\left(\frac{1}{256}\right)^2 \sqrt{2048 \times 1000^2}}{360}\right)^2} \\ &= \sqrt{\left(\frac{2}{360}\right)^2 + \left(\frac{0.4}{360}\right)^2 + \left(\frac{0.4}{360}\right)^2} \\ &= 0.6\%. \end{aligned} \tag{6.31}$$

It is evident from Equation 6.31 that the scale factor error is dominated by the accuracy of the user performing the calibration rotation.

6.2.2 Calibration Procedure

The complete calibration procedure involves twelve steps. First, the device is placed in each of six unique static orientations, described in Section 6.2.1.2, in order to obtain the minimum and maximum values for the accelerometers and magnetometers and the gyroscope offsets. Second, six 360° rotations are performed, one clockwise, one anti-clockwise, for each axis to estimate the gyroscope scale factors.

During each calibration step the device accumulates the output from each sensor axis. At the end of the step the resulting sums are returned to the host system in order to calculate the

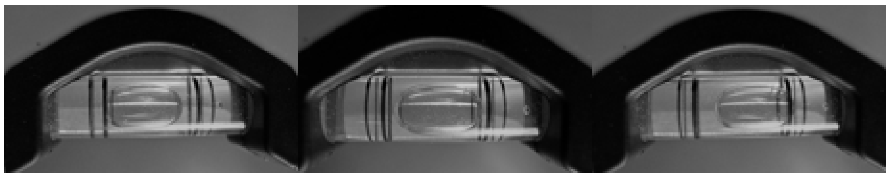


Figure 6.6: Alignment of PTU showing $-0.5°$, $0°$, and $0.5°$ rotations

offset and scale values. Accumulating the sensor outputs on the device means that reliability of the radio link is not an issue.

A GUI dialog in the MotionViewer application prompts the user to perform the necessary steps. The results of each of the calibrations are validated against minimum and maximum acceptable values derived from the sensor data-sheets. This validation helps to prevent invalid calibration of devices, for example by placing the device in an incorrect orientation. Validation also helps to quickly identify faults in the device.

After calibration, or at any time when not performing a capture, the calibration of the device can be checked by graphing the calibrated data outputs.

6.3 Static Accuracy

Static accuracy is a measure of the error between the device orientation estimate and the known true orientation when the device is held in fixed positions.

6.3.1 Experimental Setup

In order to test the static accuracy of the device it is necessary to be able to accurately position the device in several orientations and be able to do so in a repeatable manner. In order to achieve this a Pan and Tilt Unit (PTU) from Directed Perception [80] was used. The PTU is capable of rotating in two orthogonal axes with a precision of 0.05°. A Python module was written to provide simple scripting of the PTU behaviour. The module provides a PTUControl object with methods to set the target position, the rotational velocity, and get the current position of each of the two axes.

To assess the static accuracy of a test device, the PTU was mounted on its side to provide rotation in the standard pitch and roll axes.

The PTU axes were aligned using a spirit level. The spirit level used had three spirit levels at 0°, 45° and 90°. The markings on the spirit level allowed the PTU to be aligned to within ±0.5°. This was confirmed by commanding the PTU to perform 0.5° rotations away from the home position; in each case the misalignment was clearly measurable using the spirit level, as shown in Figure 6.6. A small machined aluminium block was mounted onto the PTU and the device taped on top. The device was attached so that two sides of the casing were flush to the

PTU and mounting block, allowing for repeatable mounting. The resulting setup is shown in Figure 6.7.

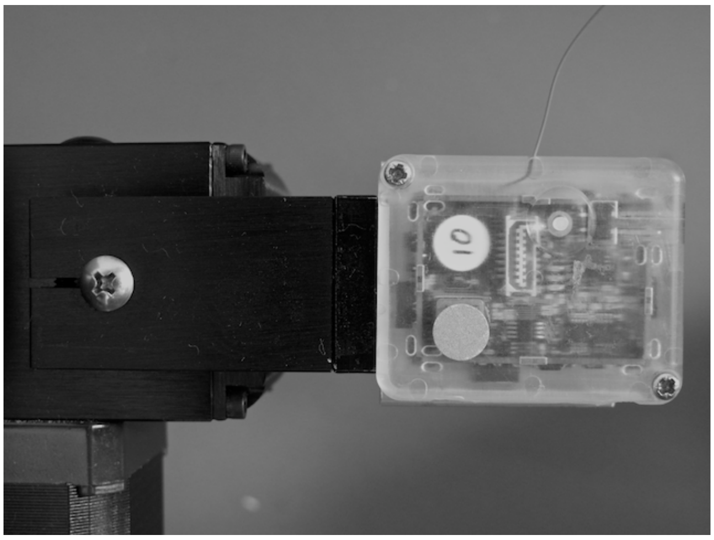


Figure 6.7: Static accuracy test device mounting

Ten *Orient* devices were tested. Each device was first fully charged and then calibrated using the procedure described in Section 6.2.2. The calibration was performed in the same room as the static accuracy test. Temperature measurements, conducted periodically during the experiments, indicated a steady temperature of $20 \pm 1^\circ$.

The PTU was used to rotate each device in to 77 unique orientations: 11 about the device x-axis, from 40° to 140° in 10° steps; 7 about the device z-axis, from -30° to 30° in 10° steps. The number of orientations was selected to provide a balance between the coverage of possible device orientations, the range limitations of the PTU axes, and the time taken to perform the experiment.

For each orientation 2048 samples were captured. The devices were configured to transmit computed orientations and calibrated data from the sensors. The number of samples was selected, based on the analysis presented in Section 6.2.1, in order to allow accurate mean values to be calculated for each sensor. The filter parameters used by the devices were: $k = 128$, $a_T = 0.1g$ and $g_T = 3.6^\circ/\text{s}$. These parameters were selected based on experience of the qualitative performance of the system over a number of years. As calibrated sensor data was also gathered it was possible to evaluate the performance of the device with varied filter parameters in simulation, rather than with repetitive experiments.

6.3.2 Experimental Results

To assess the accuracy of the device it is necessary to first select a suitable metric. The close proximity of the PTU motors to the device under test result in distortions to the magnetic field

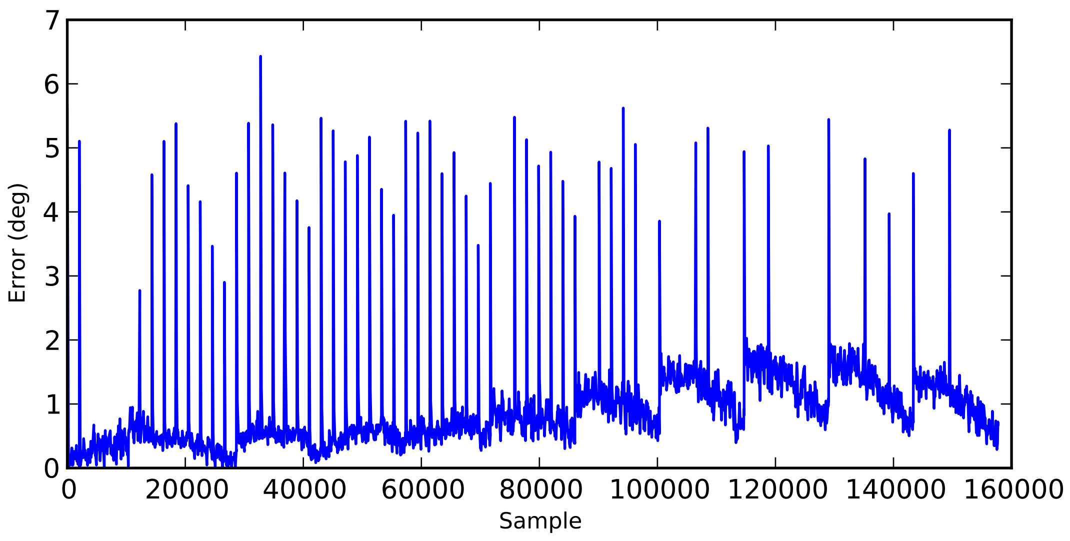


Figure 6.8: Example static accuracy trial

measured by the device. As such the heading of the device is highly unreliable. The static accuracy of the device is therefore measured by calculating the error angle, E, between the down vector, $\vec{d}$, in the device co-ordinate frame, $\vec{d}_{dev}$, as estimated from the device orientation, $\hat{q}$, and the expected down vector, $\vec{d}_{com}$, given the commanded orientation, q:

$$\vec{d} = (0,0,1) \tag{6.32}$$

$$\vec{d}_{dev} = \hat{q}\vec{d}\hat{q}^* \tag{6.33}$$

$$\vec{d}_{com} = q\vec{d}q^* \tag{6.34}$$

$$E = \cos^- 1\left(\frac{\vec{d}_{dev} \cdot \vec{d}_{com}}{\left\|\vec{d}_{dev}\right\| \left\|\vec{d}_{com}\right\|}\right). \tag{6.35}$$

Evaluation of the error in this manner eliminates the effects of magnetic field distortions, resulting in errors only due to accelerometer and rate gyroscope calibration and filter operation.

The result of a single static accuracy trial is shown in Figure 6.8. The resulting graph shows two notable features. Firstly, there are numerous large spikes in the error angle, corresponding to the convergence of the filter from an initial vector observation estimate. The noise in the accelerometer and magnetometer signals permit individual vector observations to vary significantly from the mean. Secondly, the baseline error changes as the device orientation changes.

The mean calculated errors from all devices for each of the 77 unique orientations is shown in Figure 6.9. The variation in the error angles is indicated by the error bars representing the tenth and ninetieth percentiles. The distribution of error for each orientation is distinctly non-gaussian due to the absolute nature of the error metric and the skew introduced by the initial

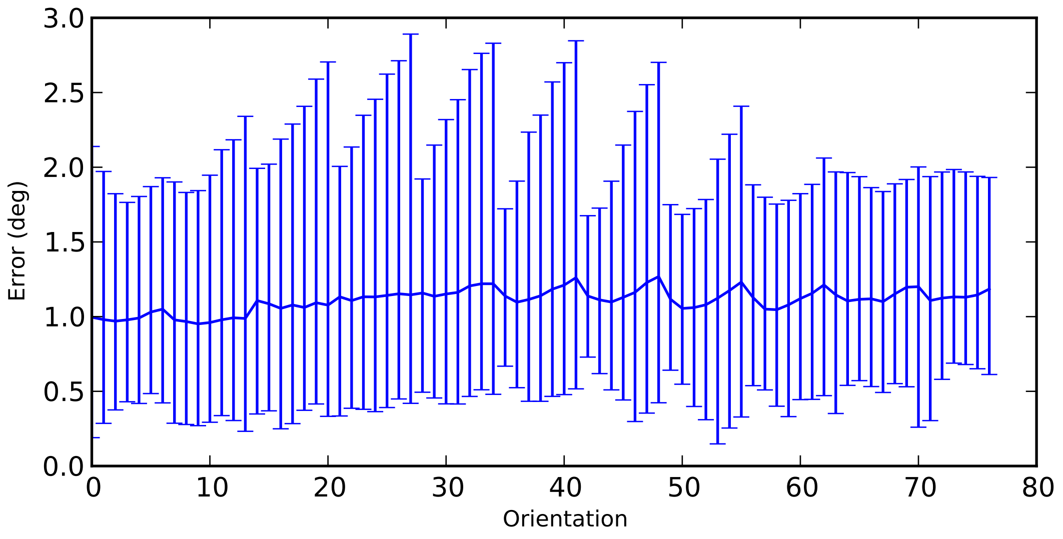


Figure 6.9: Static accuracy error for 10 devices

convergence of the filter.

To understand the errors in the static accuracy results, it is useful to consider the various sources of error in the experiment and the device.

The first error to consider is the error in the orientation filter estimate. In the static case the estimated gravity vector, $\vec{d}_{dev}$, should converge upon the mean acceleration vector measured by the device accelerometer. The error angle between the estimated gravity vector and the observed vector, calculated in a similar manner to Equation 6.35, is shown in Figure 6.10. As before error bars correspond to the tenth and ninetieth percentiles. The median presents a better estimate of the central tendency of the error as the mean value is distorted by outlying errors that occur during filter convergence, as illustrated by Figure 6.11.

The static orientation errors in the filter estimate, being much lower, are clearly not responsible for the errors seen in Figure 6.9.

The next source of error to be considered is the alignment of the PTU axes. As the PTU was aligned once before commencing the experiments, misalignment would produce a systematic error that would be the same for all devices. As such it does not explain the wide variation in observed errors. An example error trace, with PTU alignment errors of 0.5° in the pitch and roll axes, is shown in Figure 6.12. In contrast to the device errors the PTU alignment error is related to the commanded PTU orientation. Additionally, the magnitude of the possible errors is substantially lower than the observed error. To confirm this observation for other possible alignment errors a Monte Carlo simulation was performed with the pitch and roll errors modelled as independent normal distributions with standard deviation of $0.5/3 = 0.17^\circ$ to match the empirical accuracy of the spirit level. Again the magnitude of the error is substantially

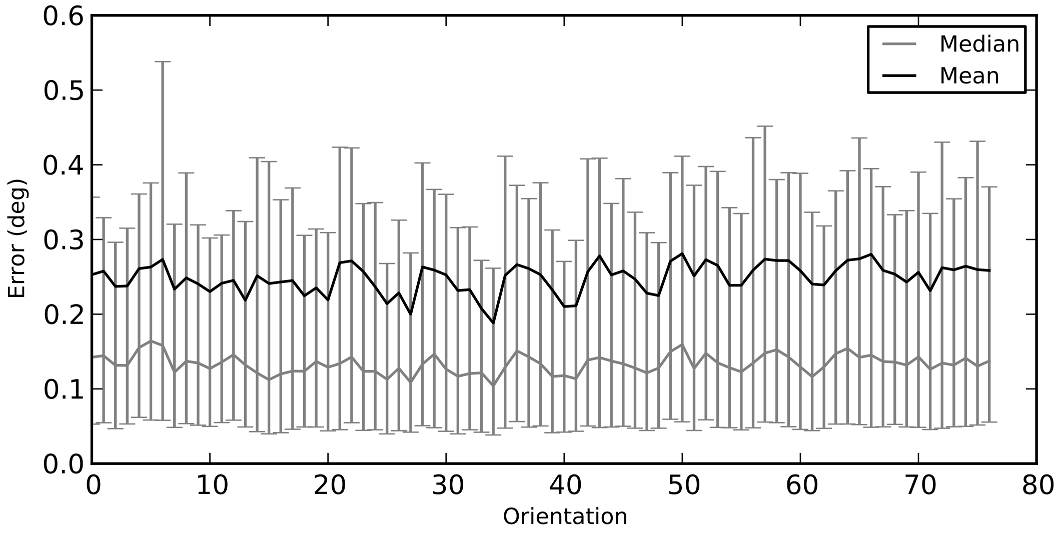


Figure 6.10: Error angle between estimated and observed gravity vectors for 10 devices

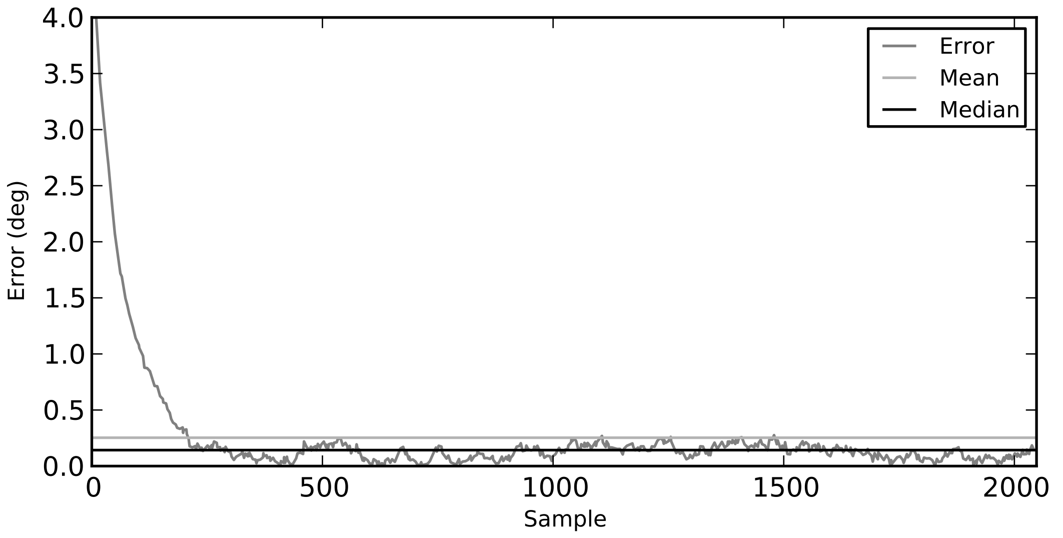


Figure 6.11: Convergence of orientation filter estimate for a single trial

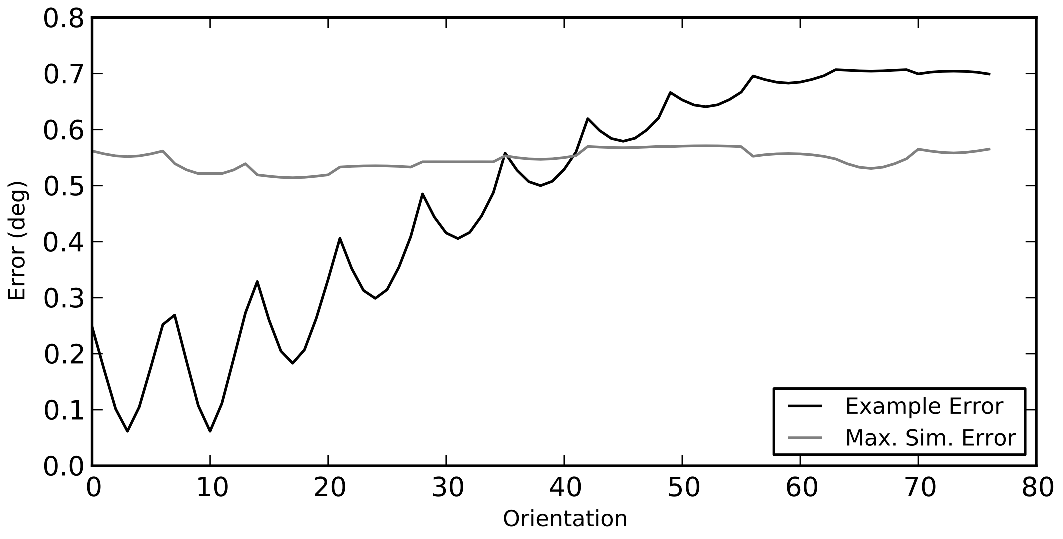


Figure 6.12: PTU alignment error

lower than the observed error in the device estimates.

Systematic error in the spirit level measurements would result in similar behaviour to that of Figure 6.12, only with potentially increased error magnitude. Again such an error cannot account for the observed device errors due to their large variation and uniformity across orientations.

The mounting of the device to the PTU is also subject to error. Unlike the alignment of the PTU, misalignment of the device would result in a constant error offset for all orientations. However, as the device casing is mounted with two edges flush to the PTU assembly, the expected alignment error is very small. With the equipment available no measurable error was observable.

The sources of error discussed so far cannot account for the degree of error seen in Figure 6.9. Furthermore, the error is known to lie in the calibration of the accelerometer rather than in the behaviour of the orientation filter. From Section 6.2.1.1 it is unlikely that the errors observed could be due only to errors in the scale and offset calibration of the accelerometer. In order to test this hypothesis the scale and offset errors were minimised by fitting the scale and offset parameters using the Levenberg-Marquardt least squares algorithm.

The result of applying least squares optimisation is illustrated in Figure 6.13. As expected, the unfitted accelerometer error matches closely the errors seen in Figure 6.9. The fitted accelerometer data shows a substantial reduction in error. However, there is still a substantial variation in the error between devices.

The scale and offset values estimated by the least squares optimisation indicate an error on the order 1%. Such errors are greater than predicted by the analysis of Section 6.2.1.1.

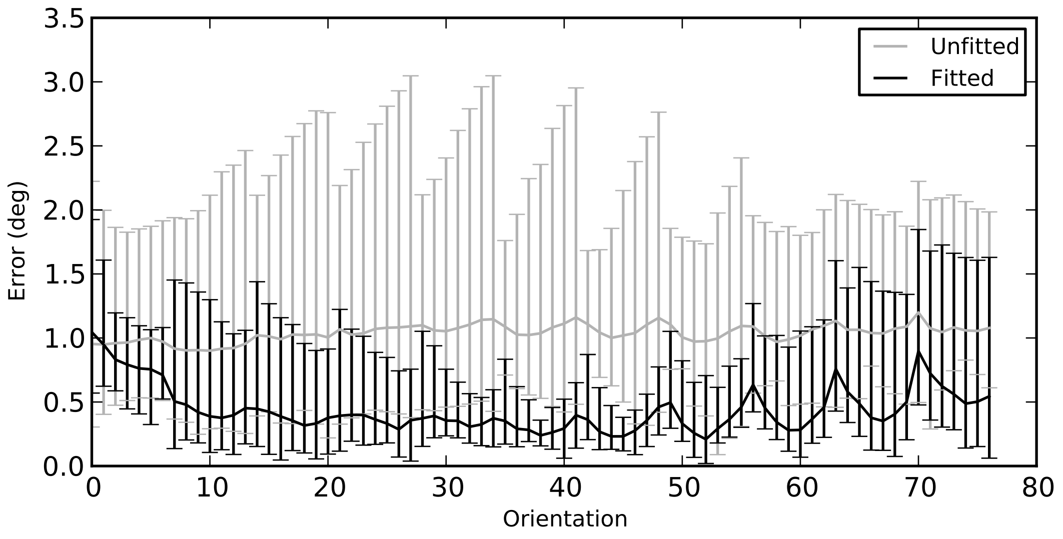


Figure 6.13: Effect of least squares optimisation of scale and offset error

The definitive cause of this increase in error has not been established. Possible sources of error include: increased accelerometer noise caused by PTU vibration due to micro-stepping; misalignments between accelerometer and device axes; and non-linearities in accelerometer sensitivity.

The simple model used so far for accelerometer calibration, modelling only scale and offset errors, is unable to account for all sources of accelerometer error. A more generic model is to replace the original diagonal scale factor matrix with a general 3×3 matrix thus allowing for any affine transform. Use of an affine transform allows the calibration to account for misalignments and cross-axis effects in addition to scale and offset errors. The result of applying a least squares optimised affine transform is shown in Figure 6.14. Applying the transform provides a further reduction over simple scale and offset correction and substantially reduces variation between devices. The remainder of the error may now be mainly attributed to PTU axis misalignment.

6.3.3 Filter Parameters

The accelerometer gating threshold, a_T, should be set such that it accepts the majority of the measured accelerations. To achieve this it should be set to approximately three times the standard deviation of the magnitude of the acceleration vector. As each component of the acceleration vector has a standard deviation of 15.3mg, the standard deviation of the vector magnitude is $\sqrt{3 \times 15.3^2} \approx 26.5mg$. The accelerometer gating threshold should therefore be set to at least $3 \times 26.5 = 79.5$mg.

The effect of varying the accelerometer gating threshold is shown in Figure 6.15. Lower

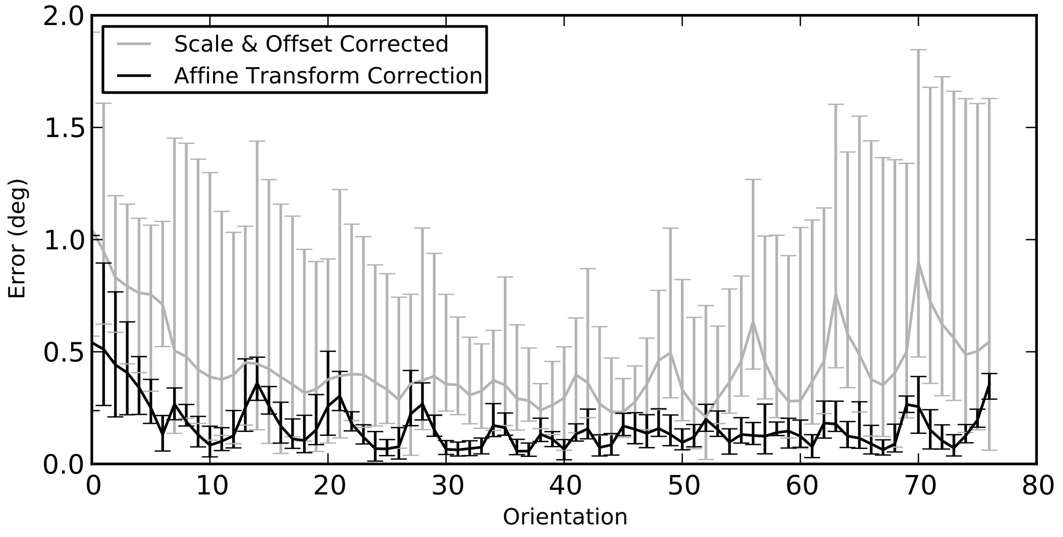


Figure 6.14: Effect of least squares optimisation of affine transform correction

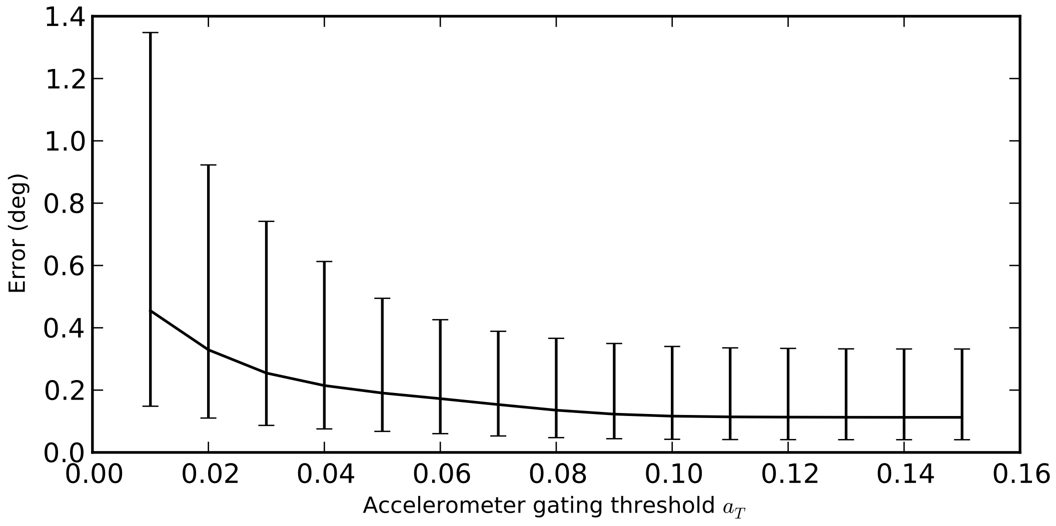


Figure 6.15: Effect of accelerometer gating threshold on static accuracy.
$k = 128, g_T = 1.79°/\mathrm{s}$

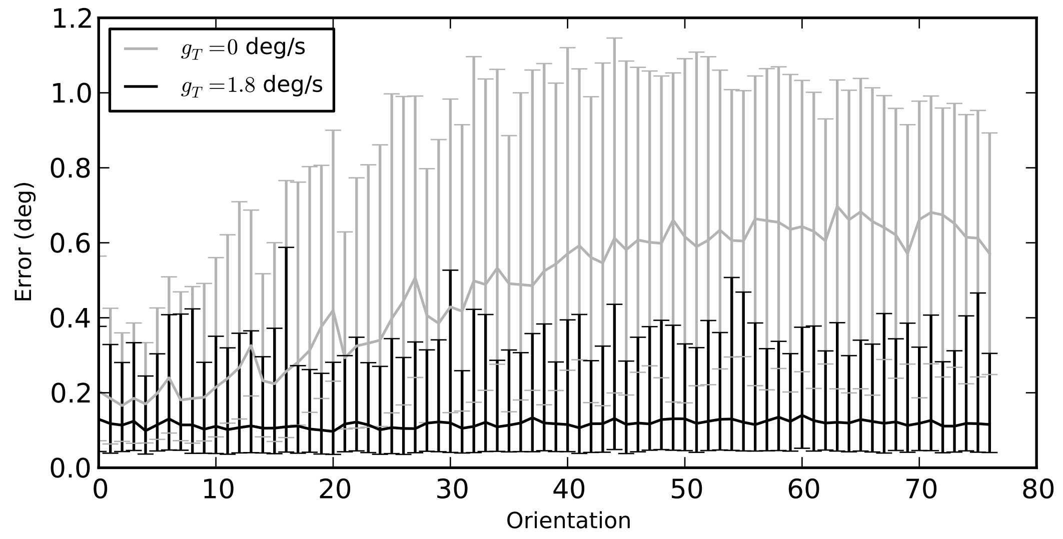


Figure 6.16: Effect of rate gyroscope gating on filter accuracy.
$k = 128, a_T = 0.1g$

values of a_T result in increased error due to increased convergence time and greater sensitivity to accelerometer scale errors. Scale errors in the accelerometer calibration result in valid accelerometer readings falling outside the gate threshold. The samples accepted by the gate are therefore skewed. Increasing the gating threshold results in reduced error as more samples are available. As expected, setting the gating threshold beyond 80mg does not provide further substantial error reduction.

The gyroscope gating threshold is used to reduce the effects of gyroscope noise and bias offset when the device is static. As discussed in Section 4.1, the orientation filter has an output bias proportional to the gyroscope bias and filter co-efficient. The gyroscope bias and noise after calibration and conversion to radians per timestep is less than one Least Significant Bit (LSB) in the fixed-point format.

The effect of applying gyroscope gating is illustrated in Figure 6.16 which shows the error angle between the estimated down vector, based on a simulated filter implementation, and the mean accelerometer measurement vector. With gyroscope gating the error is roughly uniform across the different orientations and is due to the filtered noise of the accelerometers. Further increases in gT provide no additional benefit. Without gyroscope gating the error angle is increased and varies with orientation as the additional error introduced by bias integration effects each orientation differently.

Variation of the filter co-efficient, k, has the greatest effect on the filter error. From Section 4.3, increasing k results in decreased noise from the accelerometers at the cost of increased convergence time and gyroscope bias integration error. Gyroscope gating, however, can com-

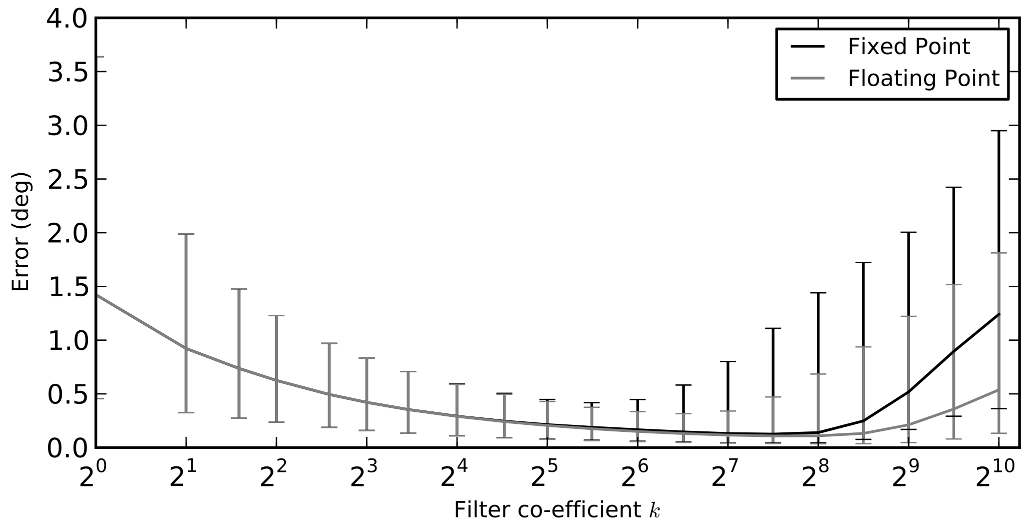


Figure 6.17: Effect of varying filter co-efficient k

pletely remove the integration error. Selecting a value for k, purely in terms of static accuracy error, is therefore a matter of selecting a balance between noise reduction and convergence time.

The effect of varying k is illustrated in Figure 6.17 for both a fixed-point and floating-point filter implementation. The graph shows the median error angle between the estimated down vector and the mean accelerometer measurement for each device and orientation against the value of k. As before the median value is selected as the estimate of the central tendency of the data as the mean is substantially skewed by outlying errors.

For small values of k the error is dominated by the noise of the accelerometers. As the value of k is increased the error level drops as the accelerometer noise is increasingly filtered out. For values of $k < 64$ the results from the floating-point and fixed-point filters are virtually identical. However, as k increases beyond 64 the fixed-point error variation starts to increase. This occurs due to the limited precision of the fixed-point format as the error signal divided by k becomes too small to represent. Figure 6.18 shows the error for each sample from a single run with three values of k using a fixed point filter implementation.

The median error for both filters continues to decrease until $k = 2^8 = 256$. At this point the fixed-point filter error begins to increase rapidly and the effects of limited precision dominate. The floating-point filter error also begins to increase at this point. However, in this case the error is due to the increased convergence time as illustrated by Figure 6.19.

The optimal value of k, in order to reduce static accuracy errors, is therefore between 64 and 512. The lowest error variation occurs at $k = 64$, however, the median error continues to decrease achieving a level state at approximately $k = 128$.

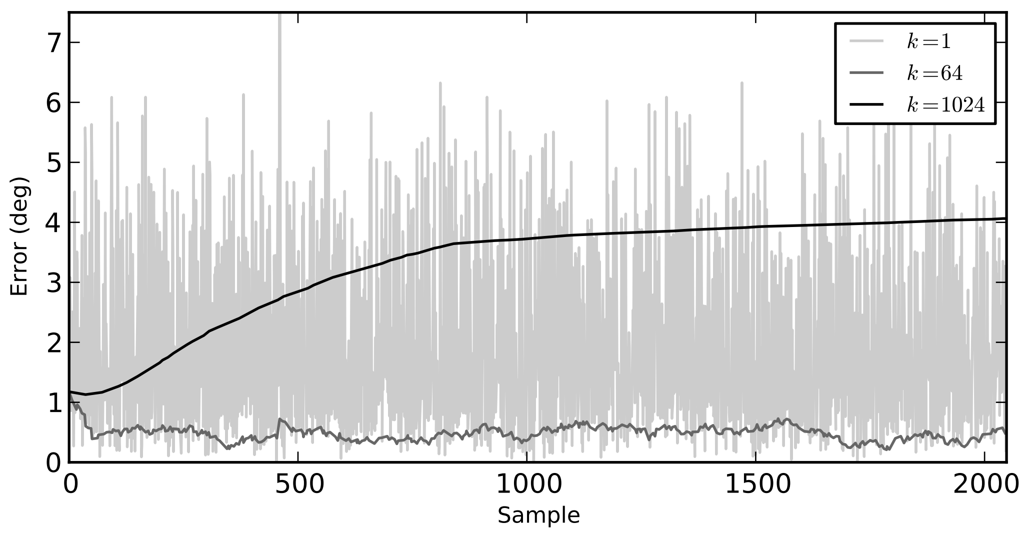


Figure 6.18: Example error trace for a single device and orientation with fixed-point filter

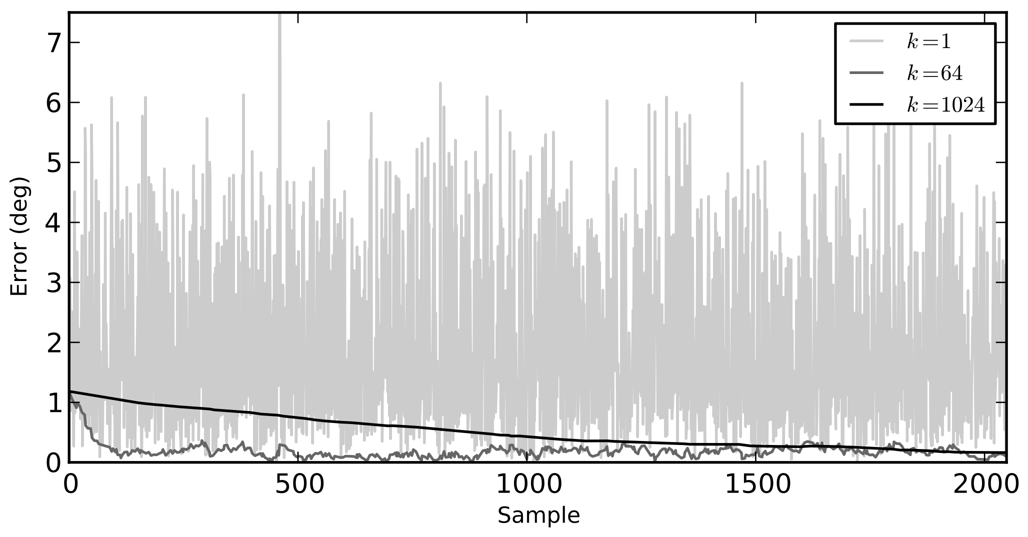


Figure 6.19: Example error trace for a single device and orientation with floating-point filter

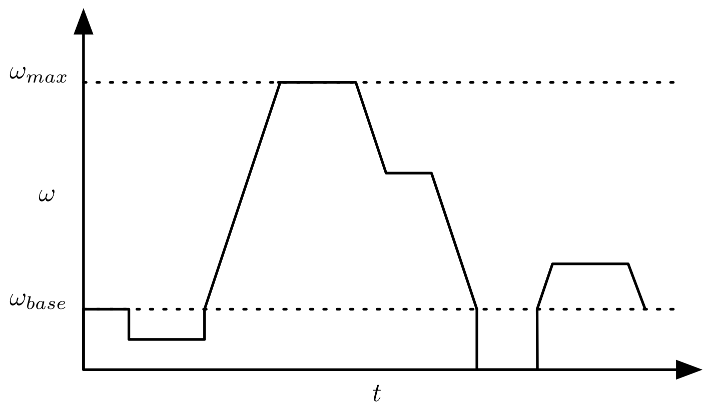


Figure 6.20: PTU angular velocity control

6.4 Dynamic Accuracy

Dynamic accuracy is a measure of the error between the estimated orientation of the device and the known true orientation whilst the device is in motion.

6.4.1 Experimental Setup

The PTU was again used to assess the dynamic accuracy of the device. The device was mounted on to the PTU in the same manner as with the static accuracy tests so that the base of the device was 45 ± 1mm from the axis of rotation. The sensor calibrations from the static accuracy test were reused.

The PTU was scripted to perform a sequence of ten 135° rotations, each rotation in the opposite direction to the last, about the device's x-axis. The rotations were performed at seven different angular rates: $50, 75, 100, 125, 150, 175$ and $200°/\text{s}$. The maximum rotational rate was selected as it was the maximum rate the PTU could achieve without the stepper motors losing synchronicity.

The PTU uses a symmetrical linear angular acceleration profile to achieve high speeds, as illustrated in Figure 6.20. The PTU supports instantaneous changes in angular velocities up to ω_{base}. Higher angular velocities are achieved by acceleration at a constant rate. The same constant rate is used for deceleration. The PTU was configured to use a base angular velocity of $10°/\text{s}$ and a constant angular acceleration of $1000°/\text{s}^2$.

The device under test was configured to transmit its estimated orientation, vector observation estimate and calibrated sensor data. The radio link between the device and basestation is not fast enough to support this amount of data at the native sampling frequency of 256Hz, so the device was configured to subsample its outputs to 128Hz. As device data was transmitted using the TDMA MAC layer it was possible to confirm that no packets were lost. The angular position of the PTU was also sampled by continuously polling the PTU controller.

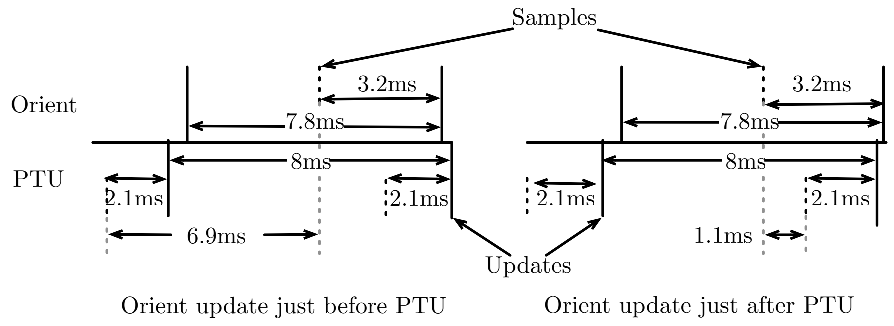


Figure 6.21: Extremes of timing jitter between *Orient* and PTU samples caused by difference in sampling rates.

Communication with the PTU is performed using ASCII commands over a 38400 baud USB-serial connection. PTU position is read by transmitting a 4-byte command and receiving a reply of between 5 and 8 bytes. The latency between of PTU position estimates is therefore at least between $\frac{5\times10}{38400} = 1.3\text{ms}$. The worst case transmission delay is $\frac{8\times10}{38400} = 2.1\text{ms}$. Additional latency due to USB scheduling, operating system and Python virtual machine overhead is exceedingly complex to estimate. The average update rate of polling the PTU was estimated by timing 1000 polls and taking the average. The average polling period was 8ms.

The latency of the data from the *Orient* device is composed of the sample acquisition time, filter processing and network transmission delay. The sampling and processing time, measured using an oscilloscope, add to approximately 600μs. When transmitting the estimated orientation and calibrated sensor values the device must transmit $4\times16+9\times16 = 208$ bits per update. An additional 72 bits of packet overhead must also be transmitted. The data is transmitted over two communications links: first, a 250kbps radio link to the basestation; second, a 230400 baud USB-serial connection to the host PC. The minimum communication delay is therefore:

$$\frac{208+72}{250000} + \frac{10}{8}\frac{208+72}{230400} = \frac{280}{250000} + \frac{350}{230400} = 2.64\text{ms}. \tag{6.36}$$

The total latency is $0.6+2.64 = 3.24\text{ms}$. Due to the much greater communications delay, the minor variations in processing time due to branches in the filter implementation are insignificant. The 128Hz sample rate results in an update period of 7.8ms.

As the update rates from the PTU and *Orient* are different there is a time varying delay between the two orientation estimates. The varying delay between estimates introduces an uncertainty in the error which is calculated as the difference between the estimates. As the update rates from the PTU and *Orient* are 125Hz and 128Hz respectively the phases of the two signals align at a rate of $128-125 = 3\text{Hz}$. The angular error introduced by the delay depends on the rotational rate of the PTU. The angular error is calculated as:

$$\theta_E = \Delta t \omega, \tag{6.37}$$

Table 6.2: Uncertainty in Angular Errors

Rotational Rate (°/s)	**Min. Error** (°)	**Max. Error** (°)
50	-0.35	0.06
75	-0.52	0.08
100	-0.69	0.11
125	-0.86	0.14
150	-1.04	0.17
175	-1.21	0.19
200	-1.38	0.22

where Δt is the time delay between the PTU and *Orient* measurements and ω is the rotational rate of the PTU. From Figure 6.21, the delay between the PTU and *Orient* varies uniformly between $-6.9 \rightarrow 1.1$ms. The uncertainties in the angular errors for each tested rotation rate are summarised in Table 6.2. It should be noted that when the rotational rate of the PTU is reversed the time delay error bounds are also reversed. Therefore, over the course of the experiment, the mean error due to sampling lag is zero.

6.4.2 Results

6.4.2.1 Gyroscope Calibration

The error in the hand calibration was assessed by performing linear regression between the rotational rate measured by the rate gyroscopes and the rotational rate of the PTU.

The rotational rate of the PTU was estimated by calculating the rate of change of the PTU output angle in the middle of the rotation. Linear regression was used to calculate the best fit straight line for each of the linear regions of the graph, as shown in Figure 6.22. The linear region of the graph was calculated based on the commanded angular rate and angular acceleration. Using the equations of angular motion the angle at which the peak velocity is achieved can be calculated:

$$t = \frac{\omega - \omega_0}{\alpha}, \tag{6.38}$$

$$\theta = \omega_0 t + \frac{1}{2}\alpha t^2. \tag{6.39}$$

The linear region can then be identified as when the PTU angle lies between $2 \cdot \theta^\circ$ and $135 - 2 \cdot \theta^\circ$. The additional scaling factor of two was introduced as examination of the gyroscope data, illustrated in Figure 6.23, indicated that the PTU was not performing symmetrical acceleration and deceleration. Examination of Figure 6.23 indicated that a factor of two ensured that sampling was performed while the device was rotating at constant angular velocity.

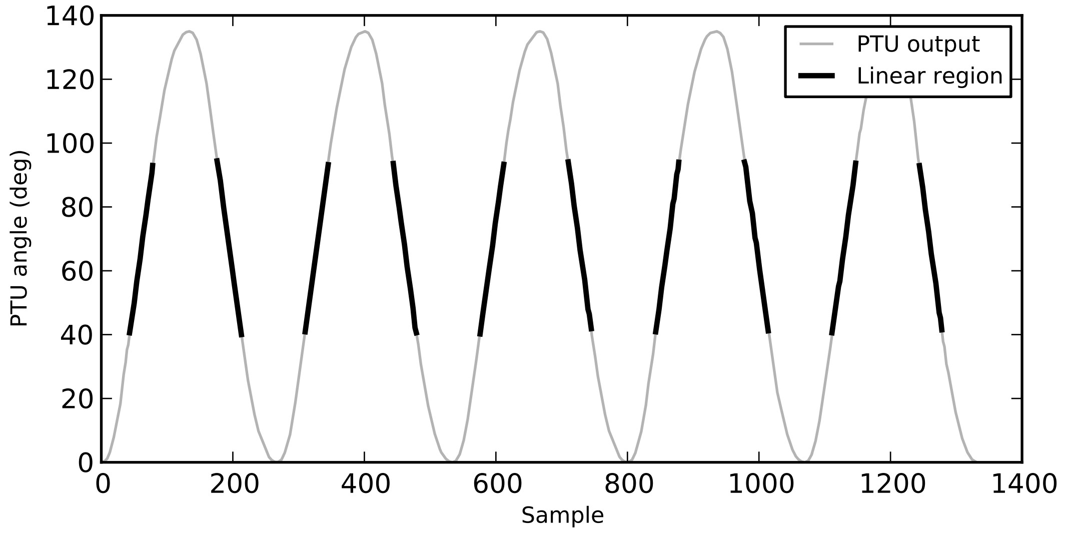


Figure 6.22: PTU angle trace showing linear angular velocity region.

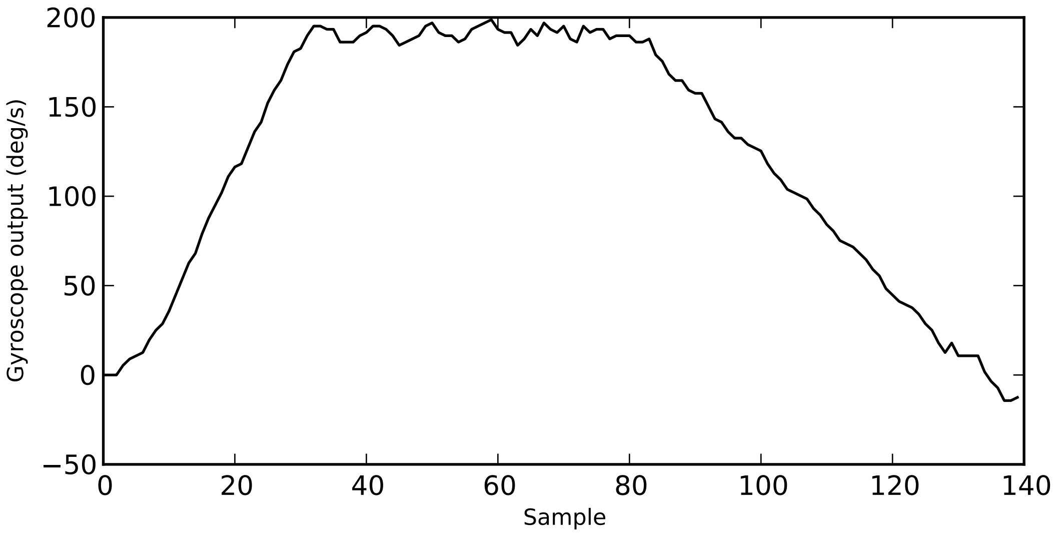


Figure 6.23: Illustration of non-symmetrical PTU acceleration during a single rotation.

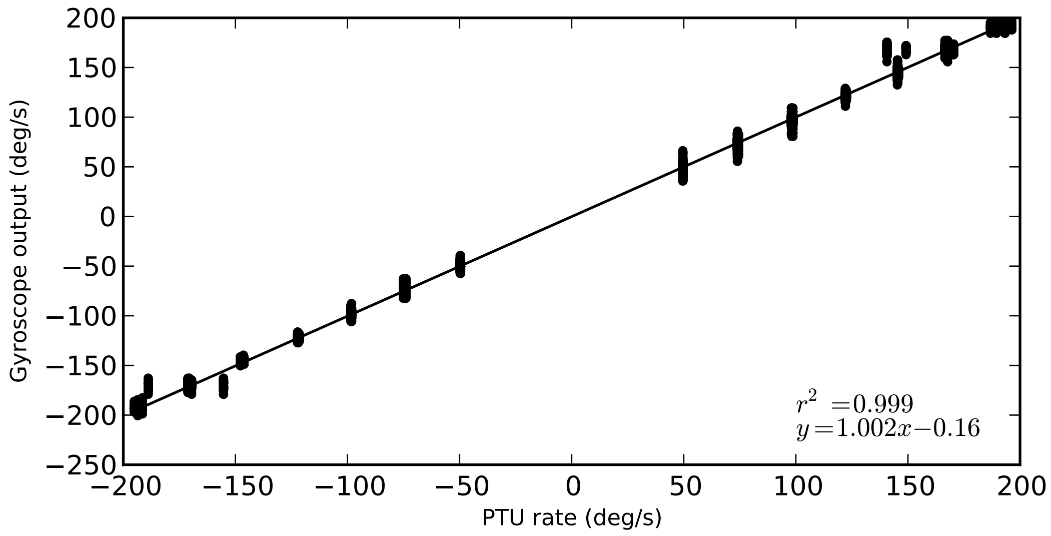


Figure 6.24: Example of linear regression between rate gyroscope output and PTU rate.

An example linear regression plot is shown in Figure 6.24. The results for each of the ten devices is shown in Table 6.3. The scale factor errors are consistent with the expected errors due to hand calibration, with the majority of errors within one standard deviation. The offset errors are significantly greater than the 0.05°/s error predicted. The explanation for this is the limited precision of the fixed point number format. The calibrated gyroscope signal is scaled such that it represents the rotational rate in half-radians per time-step. The result of this scaling is that the smallest representable value corresponds to 1.79°/s. The offsets observed therefore represent only fractions of a least-significant bit. The increase in offset error is therefore attributed to the dithering introduced by the gyroscope noise.

6.4.2.2 Orientation Estimation

As discussed in Section 4.1.2.1 orientation estimation based only on rate gyroscope integration leads to a constantly increasing error due to offset integration. This error is illustrated in Figure 6.25 which shows the PTU reference angle, pure gyroscope integration estimate, and device orientation estimation filter output angle for a single device rotating at 200°/s. The output angle for the orientation filter was produced by equating the estimated quaternion with its matrix equivalent and then calculating the x-axis rotation as:

$$\theta = \text{arctan2}\left(M_{2,1}, M_{2,2}\right). \tag{6.40}$$

It is clear from Figure 6.25 that, for a single run, the output of the orientation filter produces a more accurate estimate than pure gyroscope integration alone. This result is confirmed by Figure 6.26, which shows the aggregate results for all ten devices at each rotational rate. The

Table 6.3: Rate gyroscope calibration results

Scale Error (%)	**Offset Error** ($^\circ$/s)	r^2
1.4	-0.89	0.9996
0.6	0.09	0.9994
0.2	0.16	0.9991
0.1	-0.56	0.9984
0.6	0.12	0.9996
0.3	-0.50	0.9996
0.7	-0.30	0.9995
0.5	-0.53	0.9994
0.7	-0.42	0.9994
0.7	-0.97	0.9995

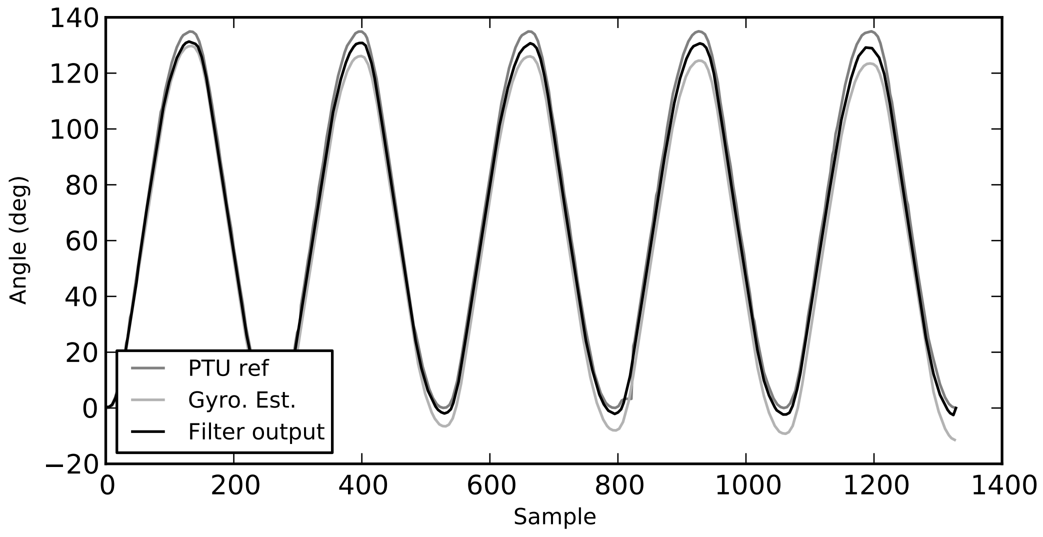


Figure 6.25: Effect of scale and offset errors on pure rate gyroscope integration.

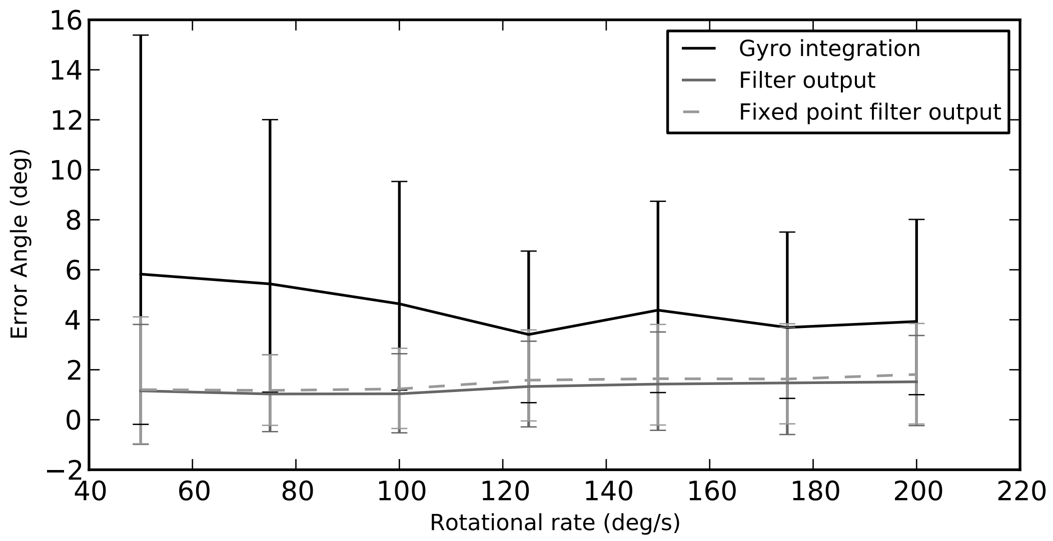


Figure 6.26: Comparison of orientation filter accuracy versus pure gyroscope estimation.

error angle is calculated as the PTU reference angle minus the estimated angle. As before, error bars are displayed at the tenth and ninetieth percentiles.

Both the pure gyroscope integration and orientation filter estimate display a positive bias in the error. From Table 6.3 the majority of the devices tested have a negative gyroscope offset. As the error is calculated as the difference between the PTU reference and the device estimate this negative bias results in the positive error seen in Figure 6.26.

The fixed-point filter implementation running on the device performs only slightly worse than the simulated filter implemented with double precision floating point values. The error between the two filter implementations is approximately 0.2°.

6.4.2.3 Filter Parameter Selection

The most important filter parameter for the dynamic accuracy test is the filter co-efficient k. The effect of varying k is shown in Figure 6.27. As with the static accuracy the selection of the value for k represents a tradeoff between different error sources. For low values the mean error is low indicating that the gyroscope offset is being mitigated. However, the variation is high due to the noise in the accelerometer data. As k is increased the variation decreases as the accelerometer noise is filtered out. Making k too large effectively reverts the orientation estimate to pure gyroscope integration resulting in increased susceptibility to offset integration error. It should be noted that, as data is subsampled by two in order to transmit it, the values of k should be doubled to select suitable values for the full sampling rate.

Gyroscope gating makes no difference to the dynamic accuracy in these tests as the rotational rate of the device is much greater than the gate threshold.

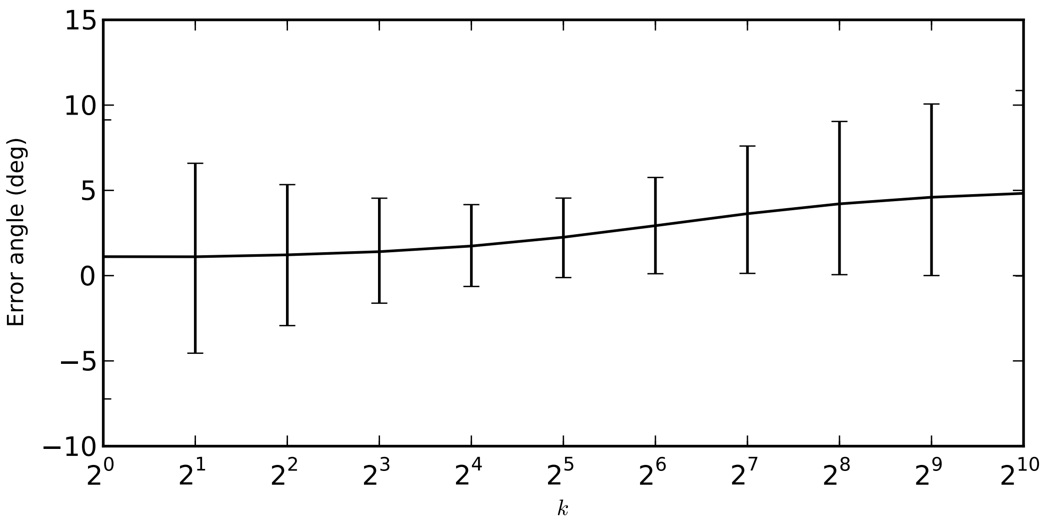


Figure 6.27: Effect of varying k on dynamic accuracy.
$a_T = 0.1g, g_T = 1.79°/\mathrm{s}$

The effect of varying the accelerometer gating threshold, a_T, is shown in Figure 6.28. As in the static accuracy case, having too low a value for a_T results in increased error as the accelerometer noise results in few accelerometer samples being accepted by the gate. The point at which the curve levels off is increased relative to the static case as the device will experience additional linear accelerations. The magnitude of these accelerations can be estimated using Equation 4.17.

The peak acceleration experienced by the device occurs during the angular acceleration of the PTU motion. Substituting the offset of 45mm and angular acceleration of $1000°/\mathrm{s} = 17.45\mathrm{rad/s}$ into Equation 4.17 results in an acceleration of $7.84\mathrm{m/s}^2 = 0.8g$. This acceleration is transitory, occurring only at the start and end of each rotation. During the rotation the acceleration is significantly less, with a maximum value, occurring at the maximal rotational rate of $200°/\mathrm{s} = 12.18\mathrm{rad/s}$, of $0.045 \times 12.18 = 0.55\mathrm{m/s}^2 = 0.055g$ during the majority of the rotation. This acceleration has a very limited effect on the accuracy of the device, corresponding to a worst case error of 3.2° calculated using Equation 3.71.

6.5 Comparison to Optical Capture

To provide an understanding of the accuracy of the *Orient* devices compared to the existing standard for motion capture, experiments were performed to compare the accuracy to an existing optical capture system [20]. The optical system used for the experiments was a six camera, passive marker, Qualisys system.

The two capture systems were synchronised by using two GPIO outputs on the *Orient*

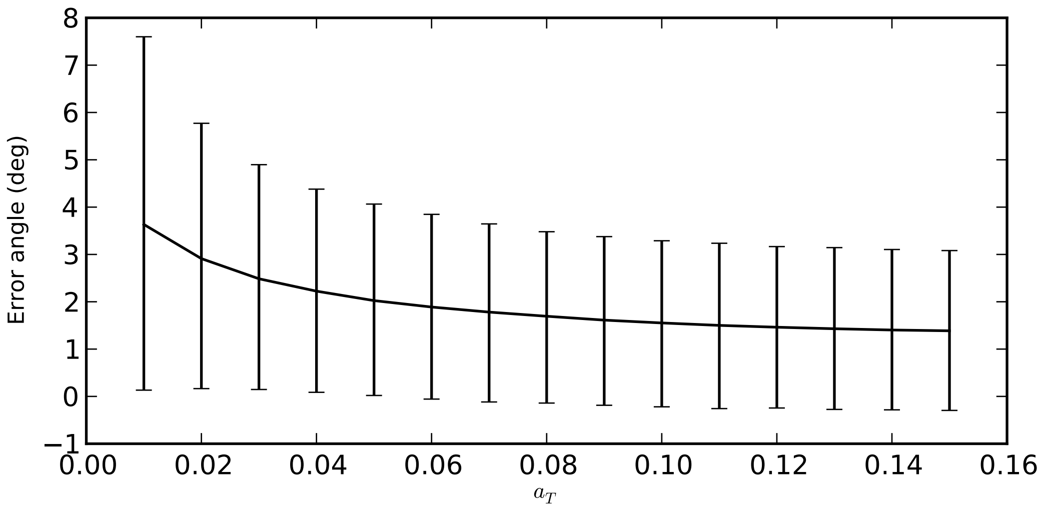


Figure 6.28: Effect of varying a_T on dynamic accuracy.
$k = 128, g_T = 1.79°$

basestation to provide start of capture and frame timing signals to the master Qualisys camera. The Qualisys software was configured to capture a frame at each pulse of the frame timing that was emitted at the same time as the TDMA synchronisation packet was transmitted to the capture devices. The latency between the two systems was therefore held constant throughout the capture. The sampling time of all devices in the network is synchronised guaranteeing that the data is gathered within 4ms of the frame signal.

In order to assess the suitability of the devices for gait analysis applications it was decided to capture the motion of a single leg of a subject walking on a treadmill. It was necessary to perform the capture using the treadmill as the capture volume supported by the Qualisys system is severely limited, only allowing 2-3 strides to be captured.

Data was captured from a single device mounted on the lower leg of a single subject. The lower leg was selected as it is subject to greater linear acceleration than the upper leg and is therefore more prone to error.

Five thirty-second captures were performed. Each capture starting with the subject standing at rest for approximately five seconds before starting to walk. The device was securely mounted to a rigid plastic set-square with optical reflective markers positioned at the corners. The three optical markers are sufficient to calculate the orientation of the device using a vector observation technique. The Gram-Schmidt process outlined in Section 3.3.3.2 was used to calculate rotations from observed marker positions. The sensor mounting can be seen in Figure 6.29

Calibrated sensor data was captured from the test device at the full sample rate of 256Hz.

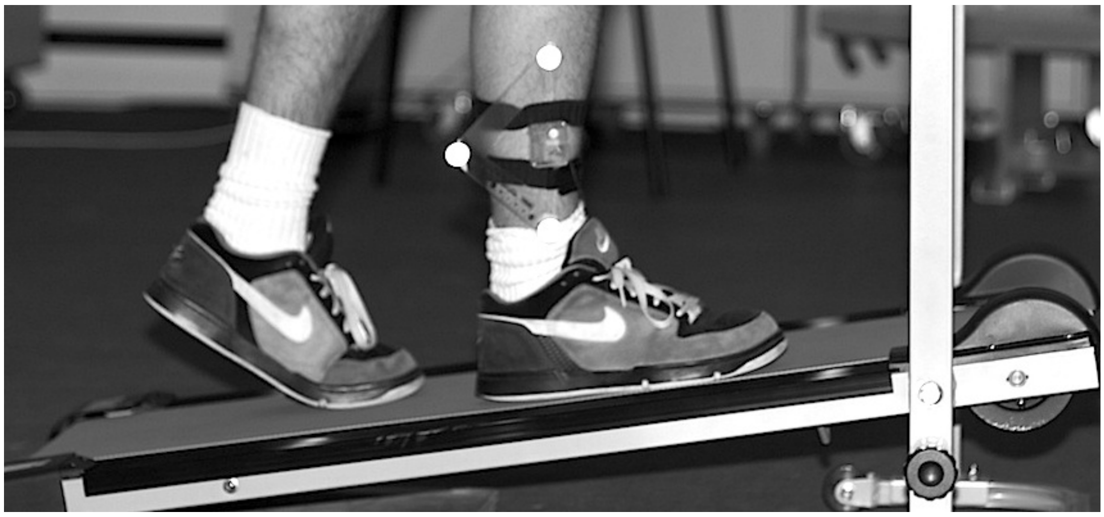


Figure 6.29: Experimental setup for comparison to optical motion capture.

Capture of the calibrated data allows for off-line filter simulation to explore the effects of variation in filter parameters. Only a single device was used during the experiments due to limited access time to the optical capture system.

The Qualisys system was calibrated prior to performing comparison captures. The calibration procedure involves placing a fixed L-shaped bracket holding four markers on the floor to define the co-ordinate frame, and then repeatedly moving a T-shaped wand of known width, with a marker at each branch of the T, through the tracking volume. After calibration the system reported an average residual error for each camera of under 2mm and a wand-length standard deviation of 2.4mm.

6.5.0.4 Problems in Performing Comparisons

Performing a comparison with optical capture is subject to several constraints and problems. The greatest problem is that the current generation *Orient* devices do not support synchronised capture of raw data from multiple devices. The *Orient-2* devices had a dedicated FLASH memory for data storage. However, this could only be accessed over a slow serial connection and was removed from subsequent designs to make space for improved charging and signal conditioning circuitry. Without the ability to store data locally on the device it is necessary to stream data over the radio. Due to the limited radio bandwidth raw data can only be gathered from a single device at the native sampling frequency. This precludes experiments involving capture from multiple devices for off-line processing.

Alignment of co-ordinate frames is another problem in performing a comparison. The optical co-ordinate frame is defined during system calibration by an L-shaped marker array. In order to align the two systems this must be carefully aligned with the local down and North vectors used by the *Orient* devices. Alignment in the pitch and roll axis is relatively straightforward as the marker array can be levelled using a spirit level. As in the PTU experiments this introduces a systematic error of approximately $\pm 0.5^\circ$. Alignment of the heading angle is more problematic as the marker array must be aligned with the local magnetic North. However, the marker array is itself ferrous and interferes with compass measurements. Additional magnetic interference was introduced by force plates in the floor of the capture laboratory. The heading alignment of the marker array was therefore performed by using a normal hill-walking compass held approximately one metre above the floor. The accuracy of the heading angle is therefore estimated at approximately $\pm 2^\circ$.

The final problem is specific to the requirement to use a treadmill to perform captures of multiple strides. It was decided to use a non-motorised treadmill so as to remove the effect of powerful motor magnets on the orientation estimation. This removes control of the walking speed from the experimental conditions. However, the treadmill does have a speedometer and this was monitored by the capture subject in order to maintain an approximate walking speed of four miles per hour. Although the treadmill contained no motors the steel frame still distorts the magnetic field. Additionally the frame uprights, seen in Figure 6.29, occluded the view of some of the cameras leading to reduced accuracy and the necessity to manually recombine small portions of data as markers passed behind the frame.

6.5.1 Results

A comparison of the simulated orientation filter output, using default filter parameters, and the orientations calculated from the optical markers for a single experiment is shown in Figure 6.30. The figure illustrates that the *Orient* is capable of tracking the general form of the optical data and the periodic nature of the gait cycle is clearly visible.

In order to examine the error between the orientation estimates the error angle was calculated by finding the quaternion rotation between each pair of orientations and converting this to an axis and angle representation. The resulting error angles for a single repetition are shown in Figure 6.31. The variation in error angle shows a correlation with the gait cycle period, increasing and decreasing for each cycle. However, the error is not directly repeated for each cycle.

The mean error angle for each repetition, with error bars at the tenth and ninetieth percentiles, are shown in Figure 6.32. The error angles factorised into Euler angle components are shown in Figure 6.33. It can be seen that the majority of the error lies in the yaw angle for all repetitions.

In order to investigate the source of these errors it is useful to examine the measured magnetic field and acceleration vectors. The magnitude of these vectors is shown in Figure 6.34. It can be seen that the magnitude of both vectors varies with each gait cycle. This variation

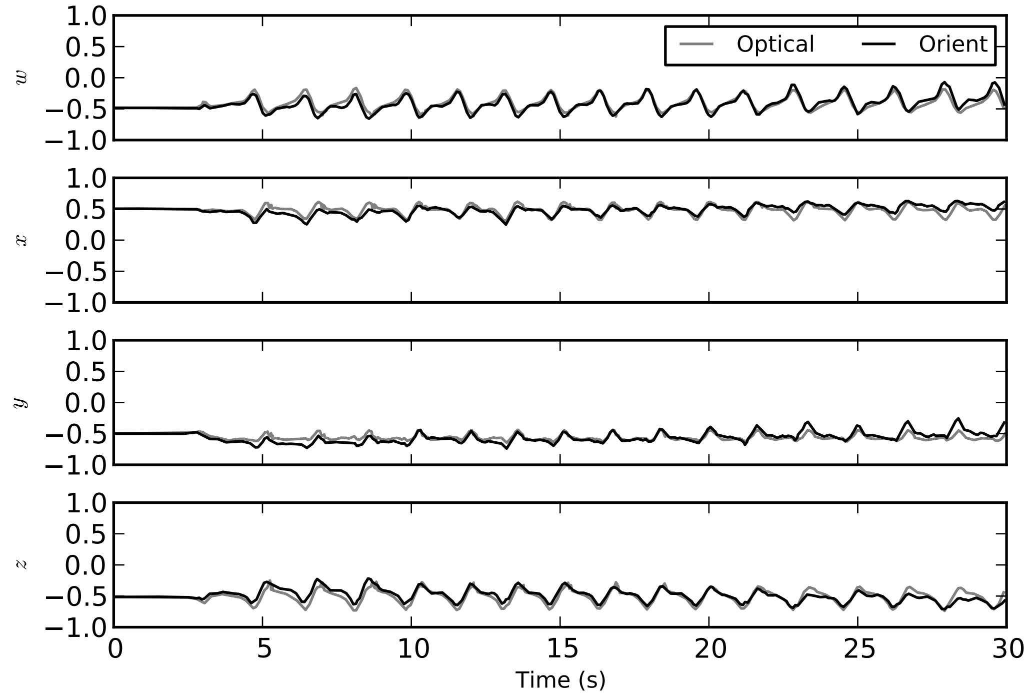


Figure 6.30: Comparison of quaternion components.
$k = 128, aT = 0.1g, g_T = 1.79°/\mathrm{s}$

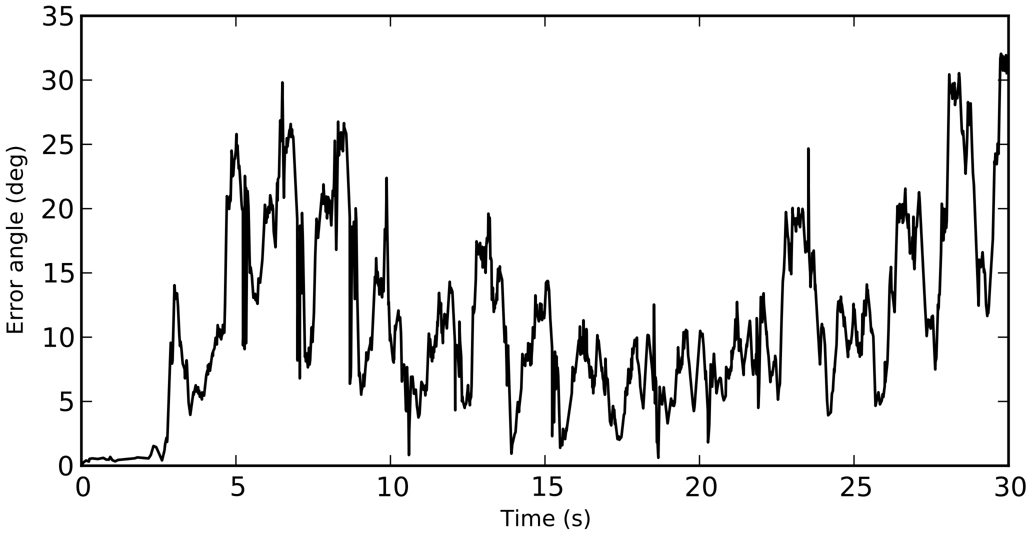


Figure 6.31: Error angle between estimates.
$k = 128, aT = 0.1g, gT = 1.79°/\mathrm{s}$

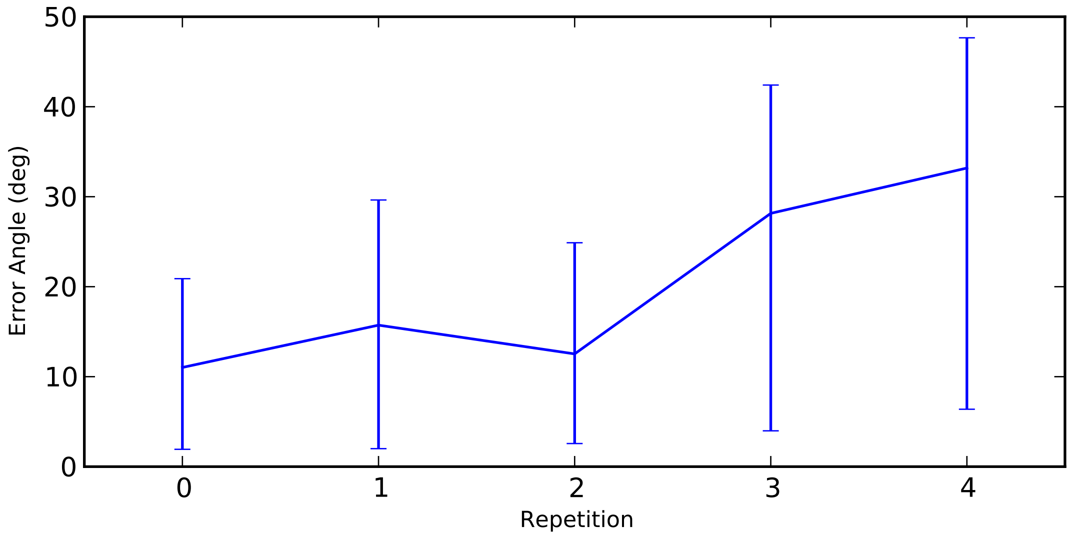


Figure 6.32: Error angle summaries for all repetitions.

$k = 128, aT = 0.1g, gT = 1.79°/s$

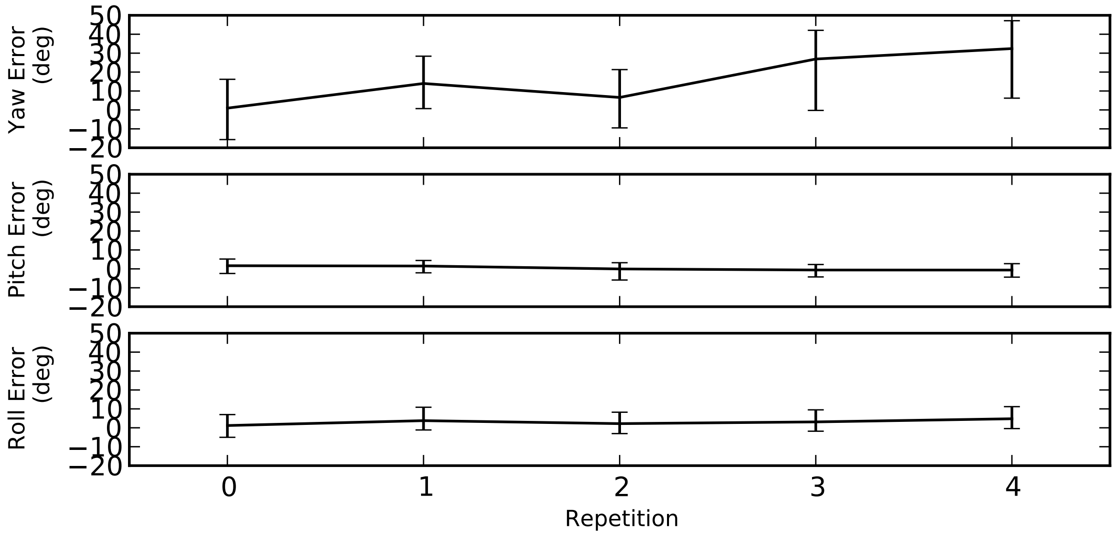


Figure 6.33: Euler error angles for all repetitions.

$k = 128, aT = 0.1g, gT = 1.79°/s$

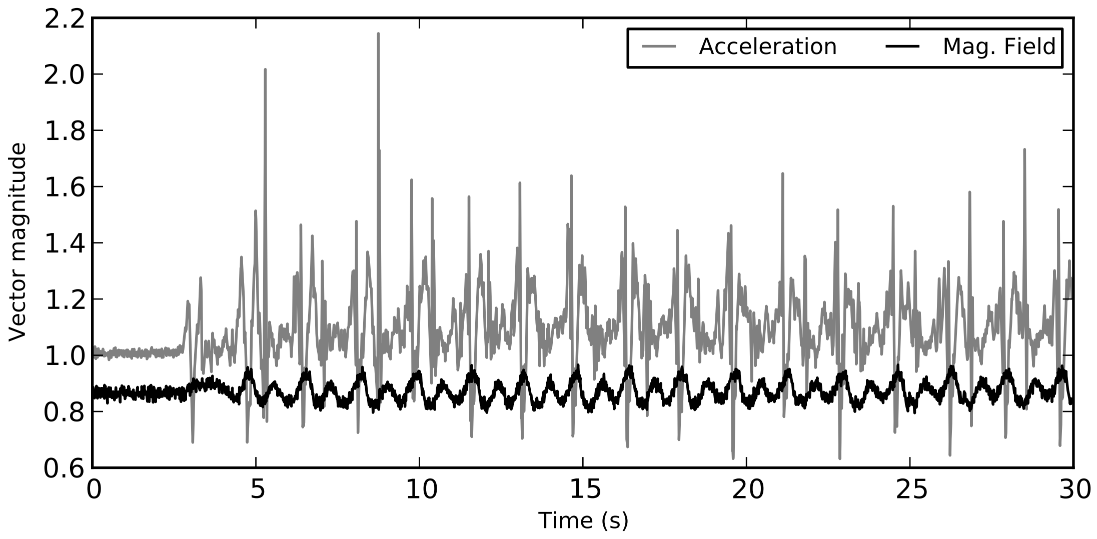


Figure 6.34: Magnitude of acceleration and magnetic field vector during a single repetition.

in magnitude indicates a disturbance in the measured quantity. In the case of the acceleration vector the variation is caused by linear accelerations due to motion. The variation in the magnetic field strength indicates a variation in the local magnetic field, most probably due to the proximity of the steel frame of the treadmill. Furthermore it can be seen that the magnitude of the magnetic field vector is uniformly less than unity indicating a variation in field strength between the laboratory where the device was calibrated and the motion capture studio.

The effects of these errors in observed vectors can be estimated by replacing the measured vector with a simulated version. Simulated vectors are calculated by rotating the Earth fixed reference vector into the co-ordinate frame defined by the optical estimate. The result of replacing the magnetic field and acceleration vectors with their simulated counterparts is shown in Figure 6.35. Additionally the effect of ignoring vector observations altogether, and relying solely on rate gyroscope integration, is shown.

The results indicate that replacing the magnetic field vector alone results in no real improvement in accuracy. This is initially surprising as the majority of the errors lie in the Euler yaw angle determined mainly by the magnetic field vector. The reason why replacing the magnetic field vector does not work is that the vector must be projected into the horizontal plane defined by the local gravity vector. An error in measuring the gravity vector results in an error when projecting the magnetic field into the horizontal plane.

The magnitude of the error angle between the reference gravity vector derived from the optical orientations and the acceleration vector measured by the device, for a single repetition, is shown in Figure 6.36. The graph shows the effect of applying gating based on the magnitude of the measured vector. The gating process reduces the RMS error between the vectors from

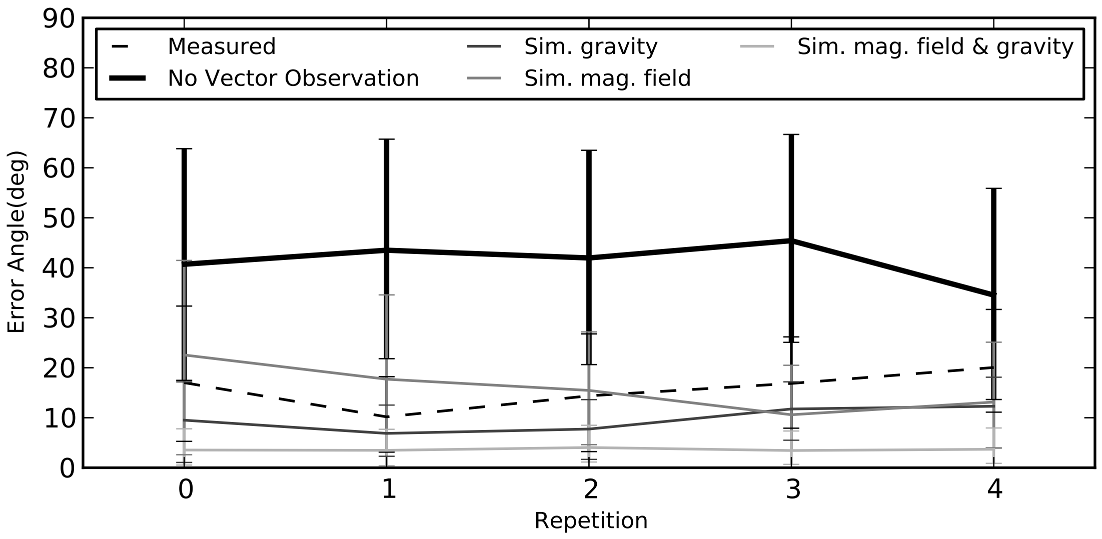


Figure 6.35: Comparison of errors when using simulated vector observations

23.2° to 16.1°. However, substantial errors are still accepted.

Substituting the estimate of the local gravity vector for the measured acceleration results in a substantial improvement in the error performance. This substitution removes the effects of linear accelerations on the orientation estimate. The errors introduced by linear acceleration are of two types: firstly, the errors examined in Section 4.1.2.3, which corrupt vector observations; secondly, when accelerometer readings are outside the gate threshold, gyroscope offset errors which go uncorrected.

The greatest reduction in error is achieved by replacing both the measured magnetic field and acceleration vectors by their simulated counterparts. The remaining error is attributed to gyroscope calibration errors. It should be noted that the use of vector observation, even with corrupted vector measurements, is significantly better than relying solely on rate gyroscopes as offset error quickly leads to large angular errors.

6.5.2 Filter Parameter Selection

As in the dynamic accuracy tests performed using the PTU the filter co-efficient k plays a primary role in controlling error. The effect of varying k is illustrated in Figure 6.37. As before, selecting too low a value results in large errors due to noise in the observed vectors. The accelerations in the treadmill experiment were much greater than those experienced using the PTU resulting in substantially increased errors. Due to the magnitude of the errors introduced by linear accelerations and magnetic field distortions the graph does not show an increase in error as k is increased. However, repeating the analysis with simulated vectors shows the expected increase as k is increased. This again illustrates the necessity to balance k between rejecting

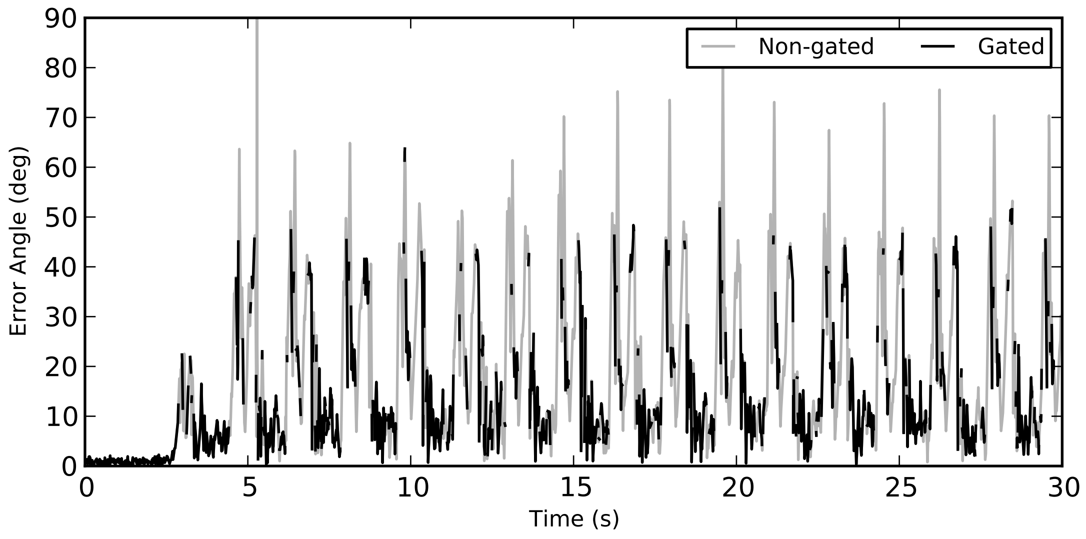


Figure 6.36: Error angle between true gravity and measured acceleration vector.

vector observation noise and compensating for gyroscope bias integration. As in the static and dynamic accuracy tests a filter co-efficient of 128 represents a reasonable compromise between convergence time and error rejection.

The effect of varying the accelerometer gating threshold is shown in Figure 6.38. As in previous cases very low values result in increased error due to accurate corrections being discarded due to accelerometer noise. Increasing the threshold beyond the noise level of $0.1g$ results in increased error as more inaccurate observations are accepted by the filter.

6.5.2.1 Comparison to Alternative Filters

In addition to the normal *Orient* vector observation method the orientation filter was simulated using the TRIAD [67], FQA [54] and QUEST [57] vector observation algorithms. The results of the simulations are shown in Figure 6.39. The three sub-optimal estimates, Orient, TRIAD and FQA, perform identically while the optimal QUEST algorithm performed marginally worse. The similarity in behaviour between the sub-optimal estimates is expected as all are based on the same principle of discarding the vertical component of the magnetic field. The QUEST algorithm incorporates knowledge of the expected magnetic field vector and is therefore additionally affected by magnetic field distortions.

As a final test of orientation filter behaviour it was compared to the two Kalman filter implementations introduced in Section 4.1.2.3. The results of the comparison are shown in Figure 6.40.

The worst performance is seen in the complementary Kalman filter. This filter, based on Euler angle representation, is unable to track pitch angles through $\pm 90°$. In the walking sce-

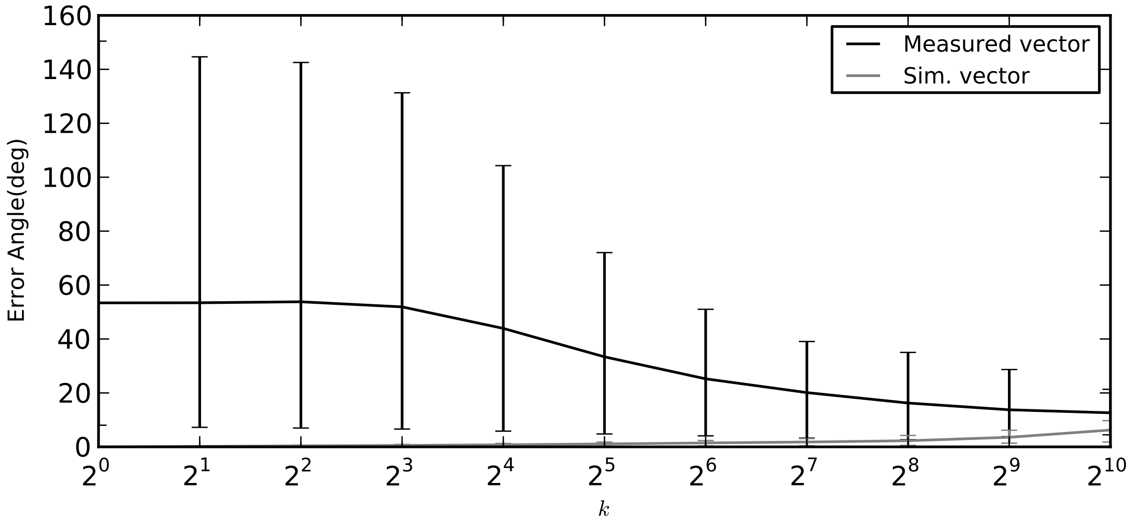


Figure 6.37: Effect of varying k on filter accuracy.

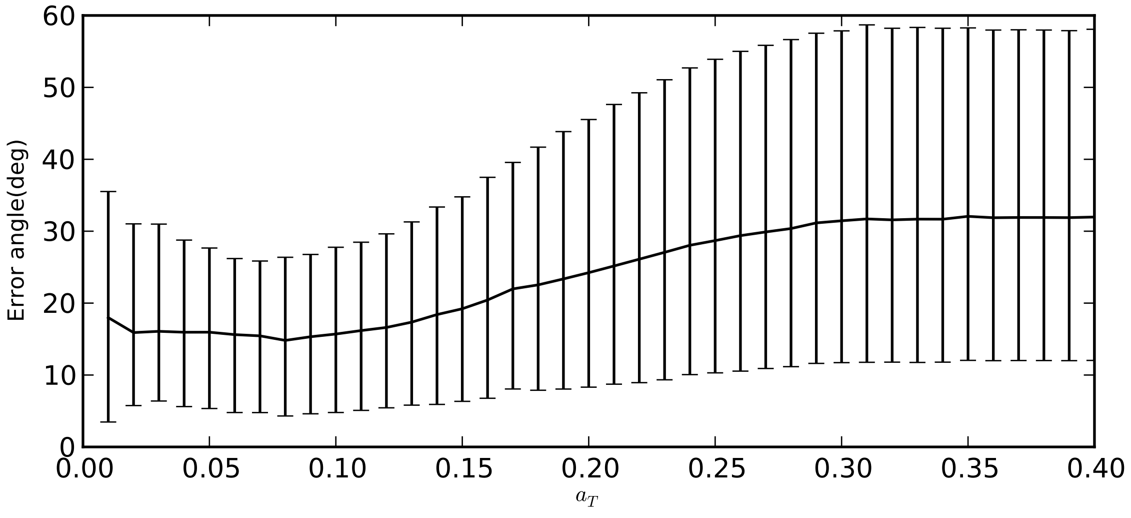


Figure 6.38: Effect of varying a_T on filter accuracy.

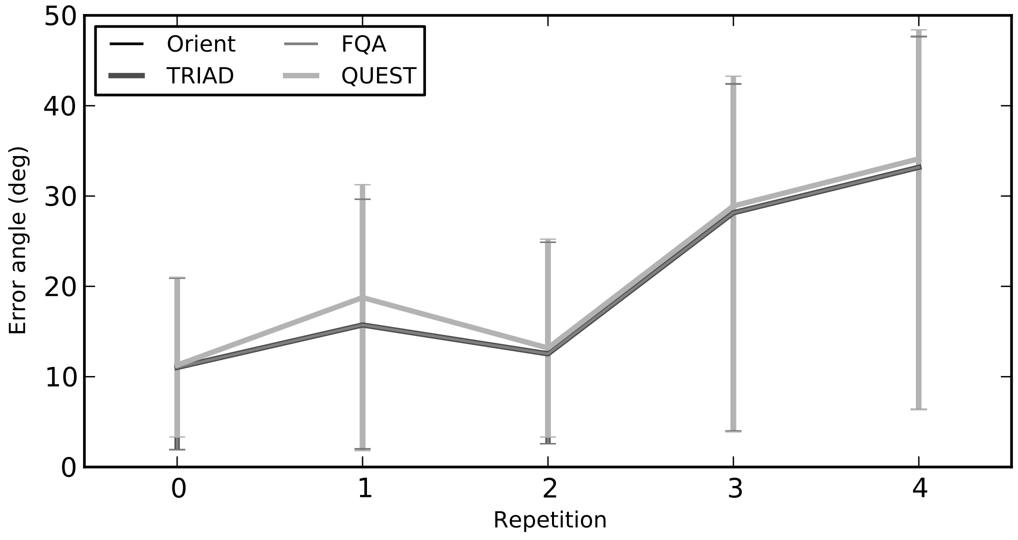


Figure 6.39: Comparison of alternative vector observation algorithms.

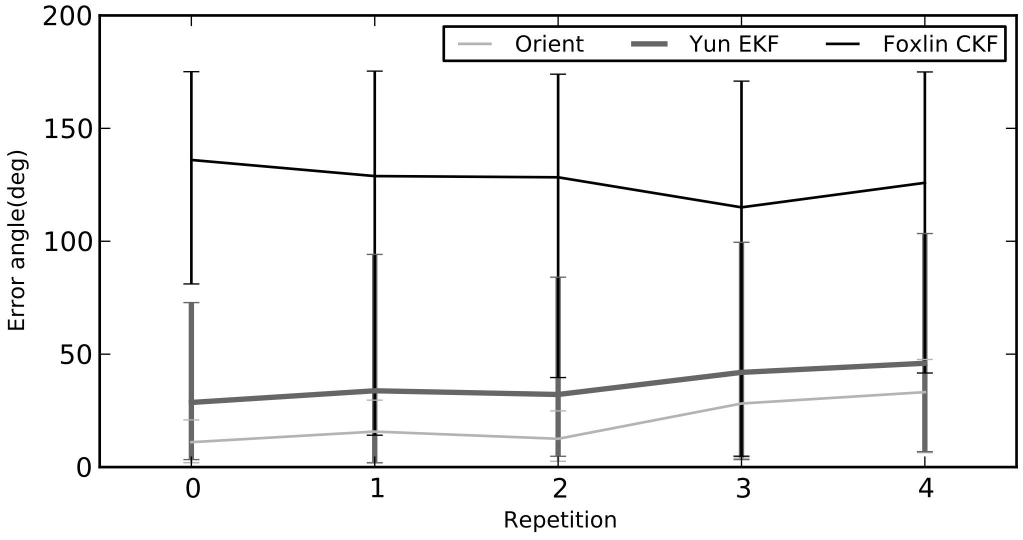


Figure 6.40: Comparison of orientation estimation filters

nario the device passes through the Euler pitch discontinuity with every gait cycle leading to complete failure of the orientation algorithm. This is consistent with the behaviour seen in the simulations of Section 4.1.2.3.

The extended Kalman filter also performs worse than the *Orient* filter. It is likely that this is due to the discrepancy between the expected dynamics, built into the filter, and the actual dynamics of the device. This illustrates the weakness of Kalman based algorithms for human motion tracking in that they require knowledge of the expected dynamics. In the case of human gait, with its regular periodic nature, it is highly likely that a better prediction model could achieve significantly improved results. However, such a model would perform poorly on unexpected motions and would not be generally applicable to the entire body.

6.6 Summary

The calibration and accuracy of individual devices have been investigated. A simple hand calibration procedure is used to allow quick calibration of devices without the requirement of specialised equipment. This calibration procedure is built in to the supporting MotionViewer software.

Testing of the ability of devices to estimate their pitch and roll orientations have been conducted using an automated test platform based on a computer controlled Pan and Tilt Unit. The results of these tests indicate that hand calibration of accelerometers is capable of achieving an average error of one degree over numerous orientations and multiple devices. Tests revealed scale and offset errors of the order of one percent, an order of magnitude greater than predicted. The source of this additional error has not been determined and is subject to continuing investigation.

Application of least squares optimisation to scale and offset parameters revealed that these two parameters are not sufficient to accurately calibrate the accelerometers. The calibration model must be extended to support a full affine transform allowing for accelerometer misalignment and cross-axis effects to be minimised. Support for affine transform calibration requires only minor modification to the existing firmware and the development of a suitable calibration procedure. For maximum accuracy an automated calibration procedure based on the PTU should be developed. These improvements are ongoing.

Testing of dynamic accuracy, again using the PTU, reveal that rate gyroscope calibration performs as predicted. The greatest error in the gyroscope calibration is experienced in the correction of offset errors. This offset error is related to rounding errors linked to the limited precision available in the fixed point number format used. Gating of gyroscope signals at very low rotational rates has been demonstrated to improve static accuracy without compromising dynamic accuracy. The ability of the orientation filter to correct for gyroscope offset has been demonstrated with the filtered orientations substantially more accurate than pure gyroscope integration.

An experiment to compare against the standard for motion capture, optical motion tracking, has been performed. Due to time constraints results were produced using only a single device and a single motion pattern. The ability to capture raw data from multiple devices is precluded by the low bandwidth of the radio communications link. Raw data is required in order to investigate variations in filter parameters and to provide insight in to the sources of error. For this reason further comparisons await hardware support for synchronised capture and storage of data from multiple devices.

The results of the optical motion capture comparison indicate that devices are able to track the general form of the walking motion, and that the orientation filter performs significantly better than gyroscope integration alone. However, substantial errors were experienced. The greatest reduction in error was achieved by substituting a simulated gravity vector that removes the effects of linear accelerations. Doing so allows for continuous drift correction. For this reason, estimation of linear acceleration is seen as a fundamental future development for improving system accuracy. Simulation of existing alternative orientation estimation algorithms has indicated that these are subject to the same linear acceleration errors as the proposed algorithm.

Finally, the selection of filter parameters has been explored. The use of gating to discard obviously erroneous updates from both gyroscope integration and vector observation has been validated. A gyroscope gate threshold of 1.79°, equivalent to 1 LSB, provides a marked improvement in static accuracy. An accelerometer gate threshold of $0.1g$, just greater than the combined accelerometer noise floor, results in optimal filter response. The filter co-efficient provides the greatest variation in error performance and its selection is open to debate as different values provide different levels of performance in varying motion scenarios. A value of $k = 128$ has been selected as a reasonable compromise between fast convergence and error rejection. Adaptive control of k, similar to the behaviour of the Kalman gain, remains an area of future research.

Chapter 7

Analysis of System Performance

This chapter presents an evaluation of the system as a whole. The semi-distributed implementation is shown to provide the optimal reduction in required data transmission. The latency of the system is broken into its constituent parts and the network latency is shown to dominate. The power budget of an individual device is examined in the various operation modes.

7.1 Analysis of Implementation Strategies

Three strategies can be adopted in the design of a wirelessly-networked motion tracking system: centralised, semi-distributed, and fully-distributed. These design strategies will be analysed in the following sections.

To provide a comparison of the number of supported devices and the radio activity, the packet format of Section 5.3.3.4 will be assumed with a radio bandwidth of 250kbps. A two-byte packet header is required in the case of framed data packets, in addition to the seven bytes added by the CC1100 packet engine.

7.1.1 Centralised Model

In a centralised model, the sensor devices transmit raw sensor data to a central host for orientation estimation and body modelling. As all orientation estimation techniques require integration of rate gyroscope data, it is necessary to transmit data at a relatively high rate. Existing inertial motion capture systems [31, 32] typically have update rates of 120-180Hz. The high update rate is required in order to accurately integrate data from rate gyroscopes.

Assuming 12-bit samples and nine sensor values per update, the minimum network data rate required, including packet overhead, is:

$$DR_{centralised} = 120 \times (9 \times 12 + 9 \times 8) = 21600\, bits/s/device, \tag{7.1}$$

resulting in a packet transmission time of:

$$PT_{centralised} = \frac{9 \times 12 + 9 \times 8}{250000} = 720\mu s \tag{7.2}$$

7.1.2 Semi-Distributed Model

In a semi-distributed model the sensor devices process sensor data locally to produce an orientation estimate. This orientation estimate is then transmitted to a central host for body model processing. As the gyroscope integration is performed locally, there is no need for the high update rate used in the centralised implementation. Instead the update rate must be merely high enough to capture human motion without aliasing. An update rate of 64Hz, more than twice the sampling rate used in television and cinema, is sufficient for most applications. Assuming the orientation estimate is represented as a quaternion, requiring only four 16-bit values to be transmitted, the following data rate requirements apply:

$$DR_{semi\_distributed} = 64 \times (4 \times 16 + 9 \times 8) = 8704 bits/s/device. \tag{7.3}$$

The packet transmission time is:

$$PT_{semi\_distributed} = \frac{4 \times 16 + 9 \times 8}{250000} = 544\mu s \tag{7.4}$$

7.1.3 Fully-Distributed Model

In a fully distributed model, each device would be required to transmit its orientation and the position of its associated body model joint. As in the case of the semi-distributed model, the update rate can be reduced thanks to the local processing of orientation data. Therefore, the network data rate required is:

$$DR_{fully\_distributed} = 64 \times (7 \times 16 + 9 \times 8) = 11776 bits/s/device. \tag{7.5}$$

The resulting packet transmission time is:

$$PT_{fully\_distributed} = \frac{7 \times 16 + 9 \times 8}{250000} = 736\mu s \tag{7.6}$$

In addition to transmitting data, devices are also required to receive data from their parent device. In the case of the CC1100 radio, packet reception is roughly equivalent in power consumption to transmission [70]. Therefore, the fully distributed implementation requires approximately double the communications power per device.

7.1.4 Sensitivity to Packet Loss

When data is transmitted over a radio network, some packet loss is practically inevitable. The ability of the system to handle packet loss is therefore of great importance. The semi-distributed

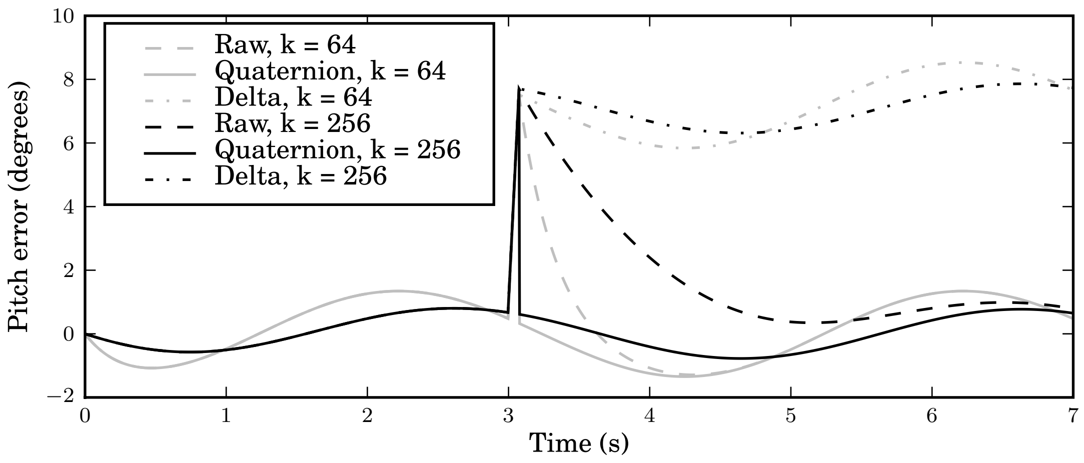


Figure 7.1: Error Angle Caused by Packet Loss

model provides additional benefits over the centralised model when faced with a lossy communications channel [21].

Three possible network implementations were considered: a centralised model, transmitting raw sensor data; a semi-distributed model, transmitting calculated orientations; and a semi-distributed model transmitting deltas between calculated orientations.

The three implementations were compared with the aid of a simple simulation. Sensor measurements were generated for a simulated device mounted on the end of a rotating arm. The arm was set to rotate at $90^\circ/\mathrm{s}$ around one axis. Two alternative values of the filter drift correction factor k were simulated for each implementation. The simulation was run for seven seconds. At three seconds into the simulation, a 100ms link failure occurs.

Figure 7.1 plots the error over time of the resulting orientation estimate at the receiver. Prior to the packet loss the results are identical for each approach. The errors present at this stage are due to linear accelerations of the sensor as described in Section 4.1.2.3. The error in the filters with $k = 256$ is less than that for $k = 64$, since the higher filter co-efficient is better at reducing the effects of linear acceleration.

During the link failure, the error increases as the true rotation of the arm diverges from the last successful update of the estimate. The resulting initial peak error is the same for all three approaches.

When communication resumes, the three approaches respond very differently. The locally produced estimate, transmitting complete orientations in each packet, immediately informs the receiver of its best estimate of the true orientation. In contrast, the centralised implementation takes a significant time to correct the error resulting from the missed gyroscope samples. The time taken to correct the error is directly proportional to the value of the filter co-efficient con-

Table 7.1: Comparison of Peak and RMS Error with Packet Loss

Data format	*k*	**Peak Error (°)**	**RMS Error (°)**
Raw	64	7.51	1.05
Quaternion	64	7.51	0.89
Delta	64	8.53	4.43
Raw	256	7.70	1.23
Quaternion	256	7.70	0.52
Delta	256	7.86	4.22

trolling drift correction. The $k = 256$ filter, which achieved smaller errors before the packet loss, now gives larger errors since the correction takes longer. Finally, the delta coded data maintains a constant offset from the true value, as it provides no way to detect the error. Table 7.1 shows the peak and RMS error figures for the different implementations, over the entire duration of the simulation.

The interaction between the value of k and the duration of packet loss induced error illustrates an additional constraint on selecting k in the centralised processing model. The properties of the radio communication error rate must be considered as, if errors are frequent, a lower value of k may need to be selected to combat the effects of packet loss. In the case of transmitting locally estimated quaternions no such constraint is imposed, allowing k to be selected for optimal filter performance regardless of packet loss rates.

The error introduced by data dependence is not simply theoretical. It has been observed in experiments using an InterSense Wireless InertiaCube-3 [31], as seen in Figure 7.2. The graph shows the output of the InertiaCube filter software during a dynamic accuracy experiment like that discussed in Section 6.4. During the test radio communications with the InertiaCube were lost for approximately one second. During the communications loss an error of 20° was introduced that took approximately ten seconds to correct.

7.1.5 Summary

The performance of the three models for the networked motion capture system is summarised in Table 7.2. The maximum number of devices is estimated by dividing the radio data rate by the data rate per device. The active radio time, either transmitting or receiving, is calculated by multiplying the packet transmission time by the required update rate. It can be seen that the semi-distributed model provides the best performance in terms of the number of devices supported and communications power. Full body posture tracking requires fifteen devices and the centralised implementation is unable to achieve this even under ideal conditions. This demonstrates that local processing of data is not only desirable but crucial in enabling full

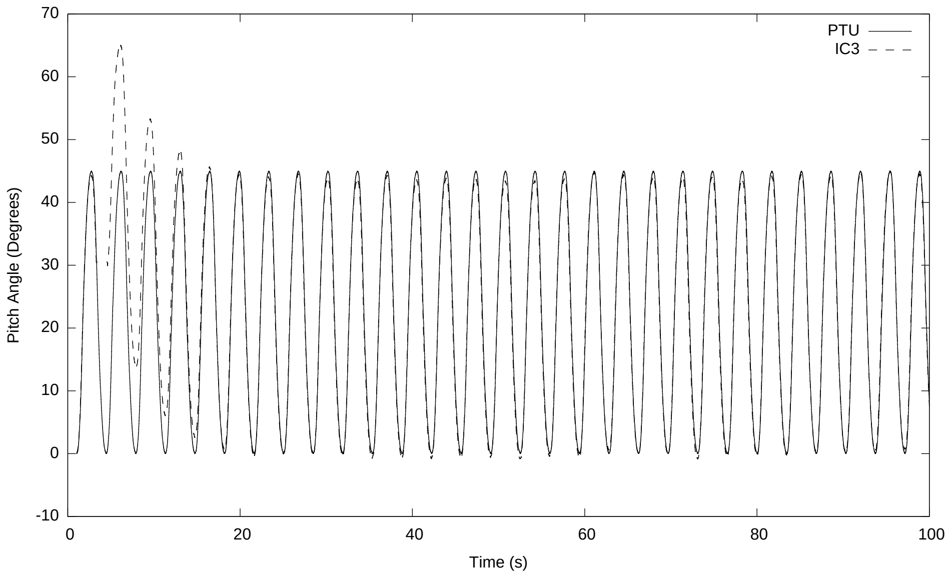


Figure 7.2: Dynamic error due to packet loss with InterSense InertiaCube-3.

	Model		
	Centralised	Semi-Distributed	Fully-Distributed
Transmission Data Rate Per Device (bps)	21600	8704	11776
Percentage of Centralised Data Rate	100	40.3	54.5
Maximum Number of Devices (ignoring synchronisation and guard times)	11	28	21
Packet Transmission Time (µs)	720	544	736
Update Rate	120	64	64
Active Radio Time (ignoring synchronisation) (ms/s)	86.4	34.8	94.2
Percentage of Centralised Communications Power	100	40.3	109

Table 7.2: Comparison of Three Models of Network Implementations

body wireless motion capture.

The semi-distributed model also has an advantage when faced with packet losses in the implementation of the network. In the centralised model, missing packets result in incorrect integration of rate gyroscope data resulting in errors. Although this error will be corrected by the estimation filter, there will be some delay in doing so depending on the size of the error. In the semi-distributed model, there is no dependence between orientation updates, and therefore correct orientation will be achieved as soon as a new packet is received.

7.2 Tracking Latency

The tracking latency of the system is composed of four principal components: sample acquisition, orientation processing, network delays, and body model processing.

7.2.1 Sample Acquisition

The first step in computing the estimated orientation of a device is to acquire data from each of the nine sensor inputs.

The dsPIC 30F3014 supports up to 13 ADC channels. However, the device has only a single ADC unit. Automatic sampling of multiple channels is provided by scanning the input channels sequentially. The dsPIC ADC is a sample-and-hold device, which requires a capacitor to be charged to the input voltage and then held while a successive approximation algorithm is used to calculate the correct sample value. In order to sample correctly each input, it is necessary that the acquisition time be long enough to charge the sampling capacitor to the correct level. Channel sampling and input multiplexing is automatically performed with a complete sample set gathered in approximately $320\mu s$.

7.2.2 Orientation Processing

Once a new sample set has been acquired the orientation can be estimated by the orientation estimation filter. The filter implementation has been highly optimised to perform well on the dsPIC. The implementation was optimised through extensive use of Microchip's fixed-point vector math library, designed especially for the dsPIC. Additional vector operations, such as cross product and normalisation, and quaternion operations are written in a mixture of C and assembler to best utilise the features of the dsPIC's DSP engine. Functions make use of the ability to perform multiple operations per clock cycle, such as multiply-accumulate operations with data pre-fetches.

A summary of the necessary functions, and their approximate cycle timings, is given in Table 7.3. The most costly functions are those that involve square root and divisions, such as the normalise functions and converting between matrix and quaternion forms.

Table 7.3: dsPIC math function instruction timings

Vector function	Cycles	Quaternion function	Cycles
Cross product	28	Multiply	49
Dot product	17	Dot product	20
Scale	18	Scale	20
Add	26	Add	29
Subtract	29	Subtract	33
Normalise	188	Normalise	214
Length	110	Create from matrix	162

The orientation filter contains no iteration and few branches resulting in a processing time which is approximately 280μs. The majority of the filter processing time is dedicated to the direct calculation of the estimated orientation derived from the acceleration and magnetic field vectors. For comparison, the time required for a single 4×4 matrix inversion, as required by Bachmann's filter, implemented using the standard dsPIC DSP library and simulated using the dsPIC simulator, requires 2.8ms, excluding the conversion to, and from, floating-point format.

As seen in the simulations of Section 4.2, the proposed orientation filter requires significantly lower processing requirements than existing filters. The primary reason for this is the simplicity of the algorithm which does not require any matrix inversions or transcendental functions.

7.2.3 Network Latency

The network latency in the semi-distributed version of the motion tracking system is the time taken to buffer and transmit orientation updates from all the devices to a host PC for body model processing.

In the *Orient* system the sampling and TDMA network operations are synchronised so that all the devices transmit their most recent orientation estimate as of the start of the TDMA frame. Devices then transmit their orientation data in their uniquely assigned transmission slots. At the end of the frame the basestation signals the PC to indicate the end of the frame and the body model is updated. This guarantees that consistent body postures are produced as both orientation data and body model updates are carried out at a single instant.

In order to have an orientation estimate ready to load at the start of the TDMA frame, it is necessary to buffer the last estimate produced in the previous frame. For a fixed sampling frequency of 256Hz, a new estimate is produced approximately every 4ms, resulting in a worst-case buffering time of 4ms.

As orientation data is loaded at the start of the TDMA frame and processed at the end of

Table 7.4: Possible TDMA frame rates for quaternion transmission

Frame rate (Hz)	256	128	85.33	64	51.2	42.67	36.57	32
Max. devices	3	7	11	15	19	23	27	31

the frame, the network latency is equivalent to the TDMA frame time. The TDMA frame time can be calculated as follows:

$$FrameTime = SlotTime \times (Slots + 1) \tag{7.7}$$

where one transmission slot is required per device, plus one extra one for synchronisation. For orientation-only tracking, each device is only required to transmit quaternion updates. As shown in Table 7.2, this results in packet transmission times of approximately 0.55ms. However, system testing demonstrated that the time required to transmit, receive and forward a packet was actually closer to 1ms. The extra time, required for transmission over the relatively slow serial connection, resulted in a 1ms slot time to be chosen.

In order to maintain synchronisation between the TDMA and sampling schedules it is necessary that the TDMA frame length be a multiple of the sampling period. The possible frame timings are summarised in Table 7.4. The MotionViewer software automatically selects the highest frame rate possible for the number of devices available and the requested data types. The calculation takes into account the number of bytes of packet header and payload; the rate of the slowest communication link, in the current system the serial connection to the PC; and an empirically determined guard time. The guard time was determined by monitoring the basestation serial lines and radio SPI bus with a digital oscilloscope. The guard time was adjusted until no packet collisions occurred as the number of devices and their data outputs were varied. A maximum network of 31 devices, enough for two full body sets, was tested.

Full body tracking requires fifteen devices to be tracked resulting in a network latency of 16ms. An oscilloscope trace of a basestation in operation is shown in Figure 7.3. The first four traces are the SPI signals between the processor and the radio. The ‘PKTRX’ trace shows the interrupt line between the processor and radio which is low during active packet reception. The final trace is the serial output forwarding the data received to the host system for body model processing. The oscilloscope cursors are set to the transmission of the TDMA frame synchronisation packets which confirm a TDMA frame time of 16ms. It can be seen from the ‘PKTRX’ trace that much of the available radio bandwidth is unused. The limit on TDMA slot length is determined by the time taken to transfer the packet over the SPI link to the processor and forward the data to the host system over the serial link.

The limiting factor in the current implementation is the serial link between the basestation and the host PC. This limit is set by the clock speed of the basestation baud generator. In the future a dedicated basestation, with a faster serial connection could be used to increase

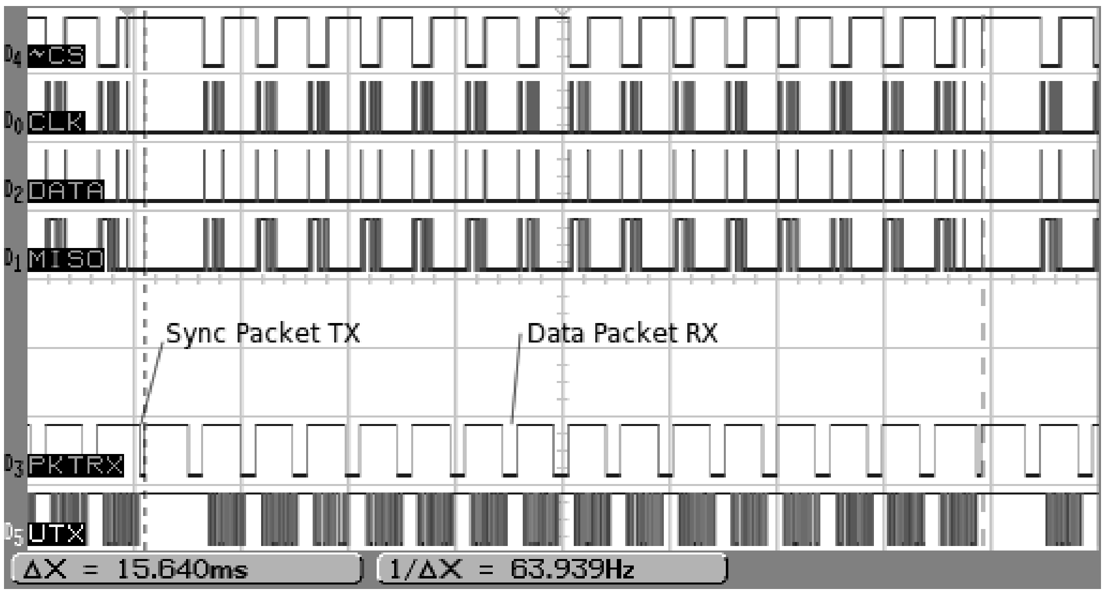


Figure 7.3: Oscilloscope Trace of TDMA Timing

network throughput, thus increasing the amount of data that each device could transmit per packet. However, due to the time spent transferring data between the radio and processor, limited by the SPI data clock, it is unlikely that more devices could be accommodated within the TDMA schedule.

7.2.4 Body Model Processing

Body model processing is performed at the end of each TDMA frame. Basic processing involves calculating the relative position of joints based on the orientation data received from the *Orient* devices.

No figures are available for body model processing as this is highly machine dependent. However, the relative position tracking code, minus 3D visualisation, has been demonstrated running on a Hewlett-Packard iPAQ rx1950 PDA with a 300MHz processor [26]. The body model processing was performed using the same *Java* code as is used on the full PC based *MotionViewer* software. This illustrates an additional benefit of the local orientation processing approach as it greatly reduces the processing load on the host system. It is highly likely that an optimised implementation of the body model processing could be implemented on a low power device such as the dsPIC.

7.2.5 Summary of Tracking Latency

A breakdown of tracking latency is given in Table 7.5. It can be seen that network latency accounts for the majority of the tracking latency because channel access must be serialised

Task	Time (ms)	Percentage
Sample Acquisition	0.32	1.55
Orientation Estimation	0.28	1.36
Network Buffering	< 4	19.4
Network Latency	16	77.69
Total	20.6	100

Table 7.5: Latency for Full Body Tracking

between devices for data transmission to the basestation. In the future a dedicated basestation device, supporting higher speed communications to the host machine, could be used to significantly reduce the latency.

As discussed in Section 7.2 the use of local orientation estimation greatly reduces the required network bandwidth and hence also helps to reduce the system latency. The network delay could be further reduced by the use of a higher bandwidth radio, at the expense of higher power consumption. Alternative network architectures will be discussed in Chapter 8.

7.3 Power Consumption

The power budget for the complete system was estimated from the power data of individual components gathered from data sheets. The estimated average currents for the different system states are illustrated in Figure 7.4. Total current draw for each mode was verified by measurement of the current using an ammeter in series with the battery.

With the standard 120mAh battery the system has been tested to run for more than 140 minutes in the TDMA mode. It is estimated that the device could remain in 'Wake on Radio' mode for approximately four days between charges.

It can be seen that the majority, approximately 70%, of the power budget is dedicated to the gyroscopes. This high power requirement is due to the inability to duty cycle the gyroscopes because of their substantial, 35ms, start-up time. This problem is compounded by the inefficiency of the 5V power supply on the current generation *Orient* devices. The efficiency of the single chip Texas Instruments Reg711 [81] was estimated to be about 67% for a nominal battery voltage of 3.7V.

Additional power savings may be achieved by use of dynamic clocking of the processor. The low processing cost of the orientation filter requires that the processor is only active for approximately 300µs per sample. When combined with the other maintenance tasks, such as initiating sampling and maintaining network time, this is equivalent to a duty cycle of approximately 12%. A hardware fault in the current generation of the dsPIC [82] prevents the use

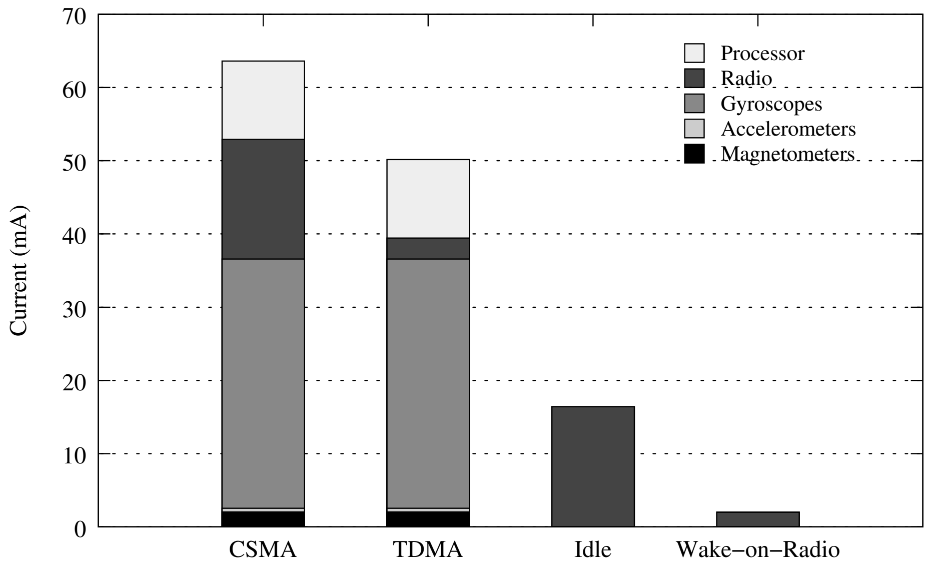


Figure 7.4: Power Budget for *Orient-2*

of the lowest power sleep mode while maintaining the RTC. As the RTC must be active to maintain sampling and TDMA timings, this increases average current draw for the processor by 10mA. It is hoped that a new revision of the dsPIC will correct this fault. In the mean time dynamic clock reduction while the processor is idle may help to reduce power consumption.

7.4 Comparison To Existing Motion Capture Solutions

7.4.1 Inertial Motion Trackers

A comparison to the commercial inertial tracking solutions discussed in Chapter 2 is shown in Table 7.6. It can be seen that the Orient-2 system is the only one to support full body tracking with a single receiver. The InertiaCube supports up to 32 devices managed by a single host but requires multiple receiver devices, increasing the complexity and the cost.

Comparison of accuracy to existing devices is complicated by the dependence of accuracy on the specific motion experienced by the device. For example the InertiaCube3 has a stated RMS accuracy of 0.25° in the pitch and roll axes, however, no indication of the motion scenario is given. The full body Xsens Moven system has a stated accuracy of approximately 2° RMS, but this requires a full bio-mechanical constraint system utilising redundant data from wired sensors.

The static accuracy of the *Orient* system, based on the results of Chapter 6, is comparable to the commercial systems. The greatest sources of error in the current design are the hand calibration of sensors and the effects of sensor misalignment and cross-axis coupling. Utilising automated calibration and least squares optimisation to calculate full affine transform sensor

Property	Device		
	InterSense Wireless InertiaCube3	MicroStrain Inertia-Link	Orient-2
Wireless	Yes	Yes	Yes
Local Processing	No	Yes	Yes
Multiple Device Support	4 per receiver, 32 total	No	15 (tested)
Maximum Update Rate (Hz)	180 (2 devices) 120 (4 devices)	100 (wireless) 250 (wired)	256 (1 device) 64 (15 devices)
Maximum Rotational Rate ($^\circ$/s)	1200		
Power (mW)	240	450	210
Dimensions (mm)	$31.3 \times 43.2 \times 14.8$ (Excluding battery)	$41 \times 63 \times 24$	$35 \times 43 \times 16$
OS Support	Windows Linux MacOS X	Windows Linux	Windows Linux MacOS X

Table 7.6: Comparison to Commercial Motion Trackers

correction will allow for further reductions in static accuracy errors.

The dynamic accuracy of the system is dependent on the motion being captured. Linear accelerations are seen as the primary source of error, resulting in either uncorrected gyroscope integration error or erroneous drift corrections. Work is ongoing to minimise these errors through linear acceleration estimation and is discussed in Chapter 8.

7.4.2 Device Cost

The cost to manufacture *Orient* devices, for a small scale production run of seventy-five devices, and amortising one-off costs such as case tooling and pick-and-place machine programming, was approximately £200 per device. Half of this is the cost of the components, with the majority, 60%, being the rate gyroscopes, and 11% the processor, radio and other sensors. The USB basestation, being substantially simpler, costs only a fraction of the device cost, approximately £15. A complete, full-body, set therefore has a manufacturing cost of approximately £3000.

The manufacturing cost of the *Orient* devices cannot be compared directly to commercially available devices as it is based on very low volumes and does not include any profit margin or the costs of research and development. For a rough comparison the Animazoo IGS-190, based on nineteen wired InertiaCube3s, costs over £40000. Optical systems have a cost ranging from just over £10000, for a low end four camera system, to over £50000[1].

The cost of the rate gyroscopes is the greatest single fixed cost in manufacturing devices. This cost is likely to go down significantly as gyroscopes become more widely available. This process has already begun with a three-axis rate gyroscope chipset available from InvenSense for $18[2].

7.4.3 Alternative Motion Tracking Technologies

In comparison to alternative motion tracking solutions the *Orient* system presents both advantages and disadvantages.

Spatial tracking systems, such as optical and ultrasonic, require dedicated infrastructure to support tracking. This requirement limits such systems to relatively small tracking volumes. While it is possible to move the installation it requires extensive calibration and planning to produce the best results. In contrast the inertial based tracking solutions, including the *Orient* system, require no external infrastructure to operate. Such systems therefore provide almost unlimited tracking.

While spatial systems suffer from limited tracking coverage they do have a significant advantage in that they can directly provide information about position as well as posture of a tracked subject. Inertial systems can estimate position using dead reckoning but are prone to drift errors when used in isolation. The problem of providing accurate spatial tracking without infrastructure is an open research question and will be discussed further in Chapter 8.

7.4.3.1 Positional Accuracy

The greatest advantage of spatial tracking systems is that they are subject to significantly lower positional errors than inertial tracking. In typical optical tracking systems, such as the Qualisys system used in Section 6.5, the positions of markers are measured independently and with great accuracy. Therefore the position of all marked points on the subject are known with equal uncertainty. In contrast, inertial systems tracking limb orientations, and using forward kinematics to estimate positions, suffer from increasing uncertainty as more limb segments are considered.

Assessing the uncertainty of position information is a non-trivial problem as the uncertainty in the orientations during motion is poorly defined. Even with accurately defined uncertainty in orientation estimates the positional error at the end of a kinematic chain is complicated as it is dependent on the orientations of each proceeding segment.

The problem of variable spatial uncertainty in kinematic chains is illustrated in Figure 7.5. The figure illustrates the result of Monte Carlo simulation of two possible three segment chains,

[1]Source:`http://www.inition.co.uk/` Prices correct as of 10/12/2009
[2]Source:`http://www.invensense.com/` Price correct as of 10/12/2009

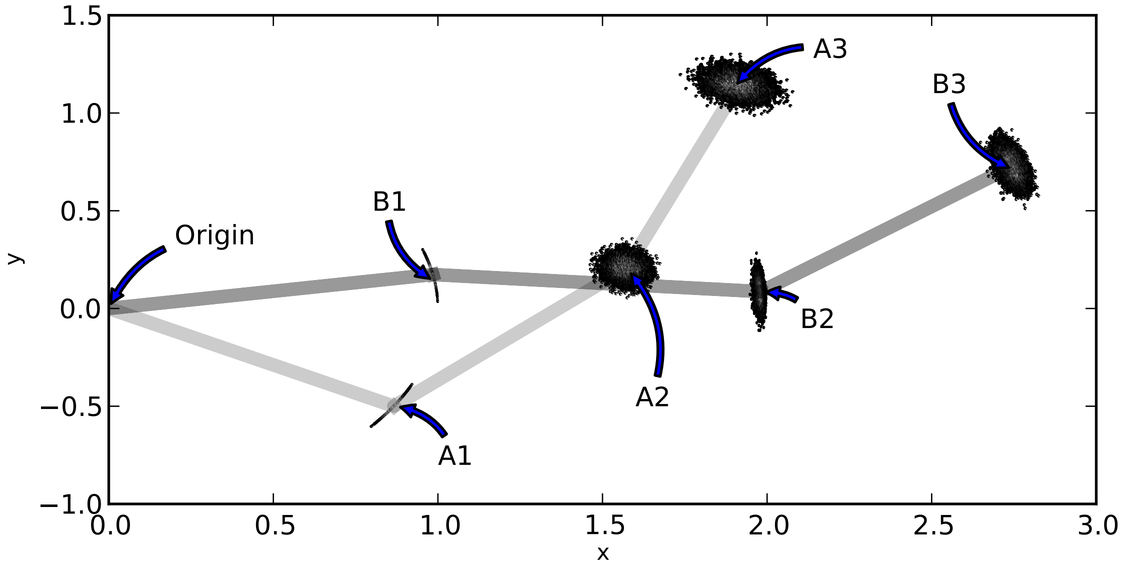


Figure 7.5: Simulated variation in position uncertainty with increasing kinematic chain length.

$(A1,A2,A3)$ and $(B1,B2,B3)$, in a two dimensional space. Each segment is of unit length and is subject to a Gaussian error with standard deviation of 2°. The error distribution at the end point of each segment is illustrated based on 10000 random error combinations. Lighter regions, in the centre of the estimated position distributions, indicate higher probabilities. As can been seen from the figure, the spatial distribution of error is dependent on both the length of the kinematic chain and the orientations of the individual joints. In the case where segments have variable length, the spatial error distribution is also dependent on the segment lengths.

The uncertainty in the orientation of the initial segment results in the end point lying on a circular arc. The subsequent segment then has a initial position error that combines with its own angular error to produce the end point error distribution. This process continues as additional segments are added. In the full three dimensional case of the *Orient* system the end point of the each joint segment has a probability distribution defined on the surface of a spherical dome which must be combined with the probability distribution of the start point through superposition.

The tendency of spatial error distributions to disperse as kinematic chain length increases poses a problem for applications that require accurate estimates of positions of body model end-effectors. In order to accurately estimate positions, the orientations of model joints must be known to a very high accuracy.

7.4.3.2 Joint Angle Accuracy

As with optical capture, the ability to measure joint angles between adjacent body segments is not directly subject to variation as kinematic chain length increases. This is because the orientation of each body segment is measured independently, relative to the Earth-fixed coordinate frame. The uncertainty in the joint angle is therefore only affected by the uncertainty in each of the adjacent body segment orientations.

While the uncertainty in joint angle is not directly related to the length of kinematic chains, it is likely that joints further from the body centre will be subject to greater error. The extremities of the body are generally subject to greater linear accelerations and rotational rates. As seen in Chapters 4 and 6, the accuracy of inertial orientation estimates depend on the magnitude of linear accelerations, and on the magnitude of rotational rate due to rate gyroscope scale factor errors, and therefore the uncertainty in the orientation of more distal joints will likely be increased.

Further experiments are required in order to assess the accuracy of joint angle estimation during motion. The error introduced by motion of the limbs is expected to correlate to some degree between adjacent limb segments. As the joint angle is calculated as the difference in orientation between two adjacent segments, any correlated error will be cancelled out as a common factor. Experiments to determine if this is the case await new hardware with support for synchronised logging.

7.4.3.3 Accuracy of Rotational Rate and Acceleration Data

Inertial sensors present an advantage over spatial tracking solutions where derivatives of spatial data are required. As seen in Chapter 6, inertial measurement devices, including the *Orient*, are capable of tracking rotational rate and linear acceleration with a high degree of accuracy. In contrast, data from optical motion capture, while normally having a much higher spatial accuracy, often results in poor estimates of derivative functions. The reason for this is that optical position data must be numerically differentiated to calculate derivatives. The presence of even a small amount of noise in the position estimates results in significant noise in the derivative. Higher order derivatives, such as calculating acceleration from position data, are particularly prone to error in this manner.

Figure 7.6 illustrates the problem of differentiation of noisy data. The figure shows the effect of numerical differentiation of a cosine function with and without an additive Gaussian noise signal with standard deviation of 0.001. The top plot shows that the two original signals are virtually identical. The lower plot shows the result of applying a first order difference numerical differentiation function. It is clear that the level of noise in the derivative is greatly increased during the differentiation. While low-pass filtering of derivative signals can help to reduce the effects of noise it comes at a cost as high frequency information in the original signal may be lost.

The ability of inertial sensors to accurately sense the derivatives of motion is the basis for ongoing collaboration between the Speckled Computing Applications Centre and the SMART

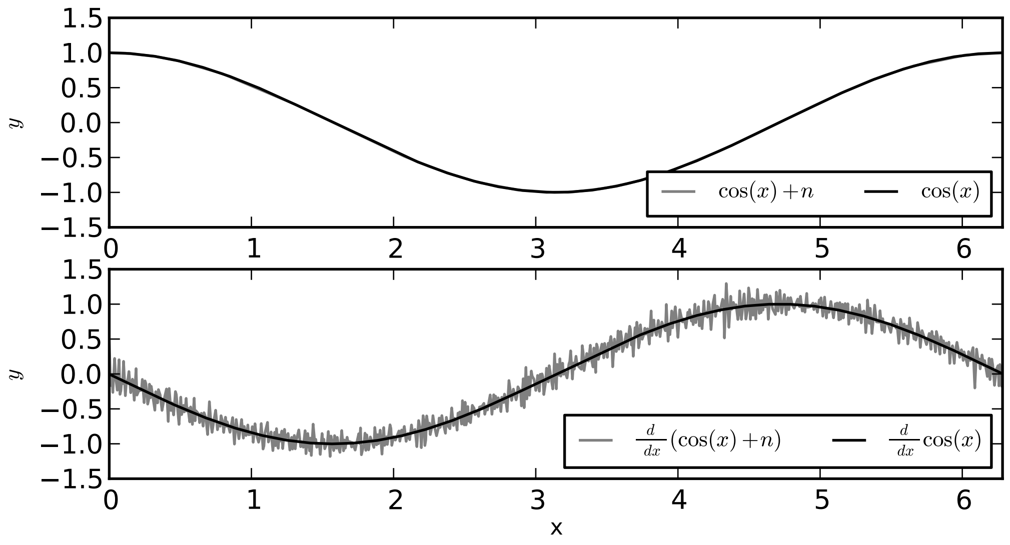


Figure 7.6: Increase in signal noise after numerical differentiation.

centre gait analysis laboratory based at the Astley Ainslie hospital in Edinburgh. Initial experiments on patients with prosthetic limbs have suggested that inertial sensors are capable of detecting subtle variations in gait, due to adjustment of prosthetic limb length, that were not detected by an optical Vicon system. Further work is required to validate these early results with additional patients and to generate a set of normal data for comparison purposes.

7.5 Experience of the *Orient* System in Use

The *Orient* system has been demonstrated publicly at two Wireless Sensing Showcase events, organised by the Wireless Sensing Interest Group, winning the award for best research and development demonstrator in 2008. Additional demonstrations have been performed at: the 2009 workshop on Implantable and Wearable Body Sensor Networks at the University of California, Berkeley; the official opening of the University of Edinburgh Informatics Forum in 2008; and the 2007 Edinburgh international science festival. Additionally the system has been used by students and other members of the Speckled Computing Consortium [26, 24, 27, 25, 28], as well as in school demonstrations throughout Scotland.

This section presents the subjective experience of many hours of use, with networks varying from a single device, to multi-device capture of full body motion, and even capture of multiple subjects.

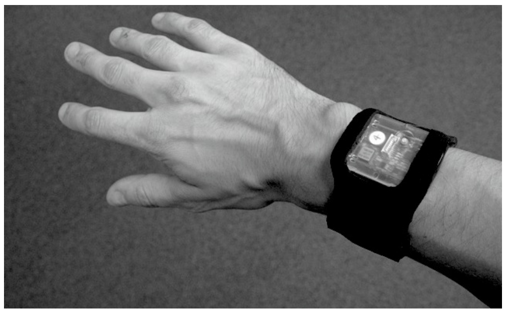


Figure 7.7: Attachment of *Orient* sensor to forearm.

7.5.1 System Setup and Calibration

To begin a new capture project the MotionViewer software is loaded and a new project created. When creating a project a BVH file containing a joint hierarchy must be provided. This file provides the information about the length of limb segments, their connectivity, and their default alignment. In the case where previous motion capture of a subject is available a BVH model can be exported and used directly in MotionViewer.

In most circumstances a generic human model can be used to give approximate results without specific measurement of the subject. Use of the generic model can lead to problems when the proportions of the subject and model differ substantially. Such errors in the model result in undesirable behaviour, such as the limbs penetrating the body or being unable to bring two joints together. A graphical user interface plugin allows modification of limb lengths to be performed when necessary. Modifications to the body model affect both existing and subsequent captures, allowing for corrections to be made after the capture session.

To begin using the *Orient* sensors with MotionViewer it is necessary to register them with the software. This is achieved by first waking the devices, using the wake-on-radio protocol, and then enumerating the available device identifiers. These are presented to the user in a table allowing device identifiers to be mapped to body segments. The device mapping is stored in the project file and so only needs to be performed once. If a mistake in the mapping is discovered, for instance swapping the left and right arms, it can be rectified after the capture.

Sensors on the arms and legs are attached using neoprene straps, as illustrated in Figure 7.7. Ideally the straps are worn underneath clothing so as to reduce movement of the sensor relative to the associated limb segment. The small dimensions and low weight of the sensor and strap allow the device to be worn comfortably and discretely under everyday clothing. Sensors for the hips and chest are attached to a belt and chest harness respectively. For coarse-grained capture of the subject's motion ten devices, attached to the hips, chest, legs and arms, are sufficient. For finer detail additional devices may added to the hands, feet and head.

Once devices are attached to the subject the alignment between sensor and joint co-ordinate frames must be configured. Alignment of the co-ordinate frames is achieved by the subject holding a reference stance while data is captured. The reference stance is defined by the joint hierarchy in the project BVH file. As such, any stance that the user is comfortable to hold for a short period can be used. The calibration capture takes thirty seconds, allowing a subject to initialise the calibration and then attain the reference stance without assistance.

The alignment between sensors and limbs is stored as a quaternion and is applied to subsequent data by post-multiplication:

$$q_{aligned} = q_{device} \otimes q_{alignment}. \tag{7.8}$$

During the alignment calibration procedure the alignment is set to the quaternion conjugate of the estimated orientation of the device. This results in the aligned orientation being the identity rotation, which, provided the subject is in the correct stance, is by definition the orientation of the joint. The accuracy of this alignment procedure is primarily determined by the ability of the subject to accurately attain the reference stance.

The total time taken to apply the sensors and perform alignment calibration is generally under ten minutes. This time may be further reduced in cases where an existing motion capture project is used and sensor mapping has already been done.

7.5.2 Subjective Impressions of System Performance

The original stance chosen for calibration was a T-stance that involved the subject standing with feet together, facing North, and with arms extended horizontally to the sides. This stance was selected as it is commonly used in optical capture where it provides clear separation between optical markers. This stance was also beneficial in integration with existing animation tools, such as Alias MotionBuilder, that expect it as the default stance.

Use of the T-stance for calibration was found to be problematic as subjects found it tiring to hold their arms horizontally. The T-stance also resulted in inaccurate alignment of the upper arm sensors due to the movement of the skin over the bone. Changing the calibration stance so that the arms were held vertically down by the body, with palms facing in, resulted in much better looking calibrations.

The problem of movement of the devices relative to the underlying bone presents a constant challenge. Wearing devices directly next to the skin, rather than on top of clothing, results in noticeable improvements in tracking accuracy. Further research into the design of straps is required as the current ones are difficult to tighten single-handedly. Loose straps, particularly on the arms, result in the devices slipping around the associated limb.

Attachment of the chest device proved a particular problem, especially for female subjects. Ideally the sensor would be placed near the top of the spine, between the shoulder blades. However, no good solution has been identified as the motion of the shoulders make it hard to secure the device.

Motions captured by the system are fluid and generally perceived as natural. Low acceleration motions are captured with good visual accuracy even during prolonged motions. Short sudden motions are also captured clearly, although generally a small amount of drift correction is noticeable following rapid motions such as punches or kicks. Prolonged capture of fast motion, without periods of rest, result in increasing loss of accuracy, as seen in the optical comparison experiment of Section 6.5. Errors are corrected once the subject returns to rest.

The choice of the filter co-efficient plays a major role in the subjective quality of the motion capture. Choosing too small a value results in jittery position estimates and unnatural looking jerks as accumulated errors are corrected. This is particularly noticeable following rapid movements where the system can appear to snap back into position. Using a larger value, while it might increase the overall error due to longer convergence times, produces an aesthetically more pleasing result as sudden changes in orientation, without obvious subject motion, are suppressed.

The most obvious error that the system is prone to are unnatural yaw rotations related to magnetic field distortions. These errors are particularly noticeable in the direction of the feet, which, being closest to the floor, are often confronted with significant distortions due to underfloor features such as electrical conduits and ventilation ducts. The effects of these distortions can result in errors of up to 180°. However, magnetic distortions are generally limited in range of effect with sensors mounted high on the lower leg often being only minimally effected. Other ferrous structures within the environment, such as steel girders, metal filing cabinets, chairs, and tables, all result in distortions to nearby sensors.

Predicting the exact effect of metal within the environment is extremely complicated, varying dramatically based on the size, shape, and orientation of the ferrous structures. Experience indicates that subjects should try to maintain a distance of at least one metre from large metal objects. Calibration of the magnetometers within the capture volume can help to improve accuracy by reducing the scale errors introduced by ferrous structures within the environment. Calibration of the magnetometers, using a subset of the calibration steps outlined in Section 6.2.2, takes under a minute per device and the user is guided through the process by the MotionViewer software.

7.5.3 Radio Network Performance

Radio interference and packet loss have not presented serious problems. While no formal range tests have been performed the system has been used with subjects up to fifteen metres from the basestation without noticeable degradation of packet reception rate. The TDMA network protocol makes detecting missing packets within frames a simple task and this data can be inspected using a GUI plugin. Additionally the devices and basestation can be configured to provide Received Signal Strength Indicator (RSSI) data for both the reception of synchronisation packets by the devices and reception of data packets by the basestation. The system is able to maintain a packet delivery rate of better than 95% with RSSI levels between -20dBm and -80dBm

Networks of between one and thirty-one devices have been tested. Testing of packet reception rates indicated that every fourth device in the TDMA schedule suffered from significant (greater than 25%) packet loss. The cause of this packet loss was traced to interaction between the TDMA and sampling schedules. After resolving this timing issue all devices achieved the same high level of packet reception. The problem of packet loss was not noticed until real time packet monitoring was added to the software as the effects of sporadic packet loss, discussed in Section 7.1.4, are not obviously discernible.

Interference from other radio systems was not experienced. Use of the 433MHz ISM band is much less common than the 2.4GHz band. However, only two orthogonal radio channels are available, limiting the maximum number of subjects that can be tracked. As the current system implementation transmits data to a fixed basestation the radios are all configured to use a maximum transmit power of 10dBm. Interference is therefore possible between networks over a considerable distance. In the future it is planned to remove the requirement for a fixed basestation, replacing it with a subject-mounted relay device bridging between the low-power body area network and more powerful, higher bandwidth, uplink such as Bluetooth or WiFi. This would allow reduced transmission power between local devices, in turn reducing the interference range of the network.

7.5.4 Integration with Other Systems

Integration with other software involves writing a plugin for the MotionViewer software. Plugins have access to both the kinematic body model and the raw data coming from the *Orient* sensors. Two sorts of plugins are supported: live plugins and export plugins.

Live plugins operate in real time as data is received from devices. Examples of live plugins developed to date include live export to Alias MotionBuilder for real time character animation, illustrated in Figure 7.8, network monitoring of packet reception rate and RSSI levels, body model adjustment, and footstep based position tracking.

Export plugins allow data to be rapidly exported to files for import into other programs. Existing export plugins include BVH, TSV, and C3D formats. Export plugins are not constrained to operate in real time, instead they simply process frames as fast as possible. As such they are more suitable for bulk export of a capture project to alternative software then real time live plugins. During export any live plugins are also called allowing extensions, such as foot tracking, to modify the data as if during normal capture or playback.

Integration with other systems at a hardware level is more limited. Devices have five GPIO pins available, including two analogue inputs. The GPIOs can be configured to output a variety of signals, such as frame synchronisation and process state, or to act as an input. Inputs are

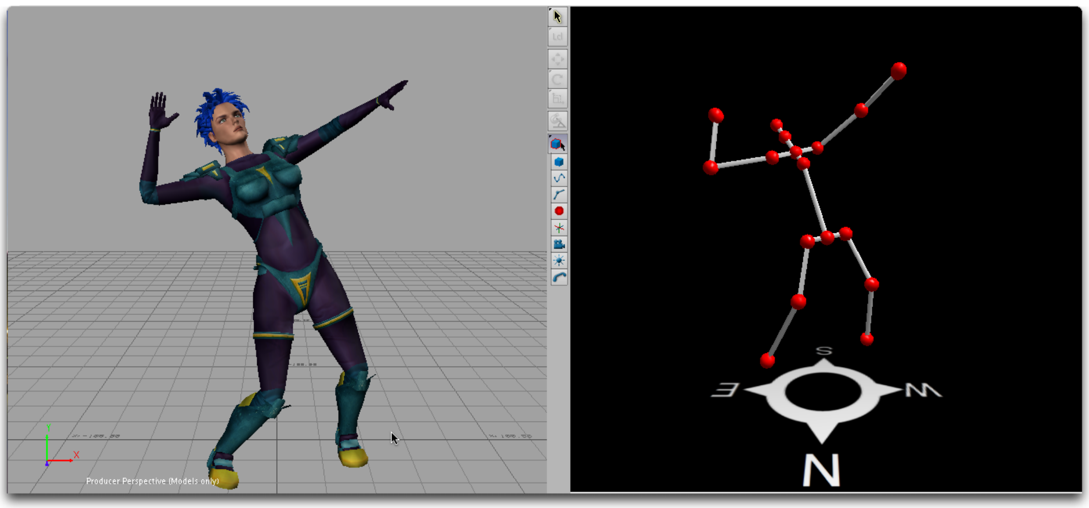


Figure 7.8: Real time mapping of body model motion to animated character in Alias Motion-Builder

sampled along with the main sensors allowing a variety of simple sensors to be attached. The basestation supports output of frame timing and capture start/stop signals for synchronisation of external systems. Support for synchronisation to external systems is under active development.

7.5.5 Suitability for Use in Motivating Applications

7.5.5.1 Animation

The ability to use the system for real-time animation has been explored in collaboration with two local animation companies and other groups within the University of Edinburgh School of Informatics. In general feedback has been encouraging with witnesses impressed by the quality of motion produced.

The most useful features of the system are the ability to quickly set the system up in almost any environment, and to see, in real time, the result of an actor's motion on the animated character. Combined with immediate playback this allows a fast workflow of rapid takes. The two biggest issues have been related to the inability of the system to directly track the position of a subject within a space and the errors introduced by magnetic field distortions.

For the short capture sessions performed to date the limitations of battery life have not been evident. However, in a production system it would be desirable to allow for all-day capture. With the development of new gyroscopes, allowing single board inertial measurement units to be fabricated, it is proposed to separate the battery from the device. This would allow a lower profile device to be built and allow rapid substitution of freshly charged batteries. Batteries could be integrated in to the existing straps with little modification.

The system as it stands is usable for generating three dimensional character animation in a far faster, and more natural manner, than key-frame animation. Further development, to support tracking of multiple subjects, would allow the system to be used for capturing subject interactions, a problem that is extremely difficult with today's technologies.

7.5.5.2 Gait Analysis

The suitability of the system for use in medical applications has not been demonstrated. The inability to accurately measure, with a high degree of confidence, the joint angles precludes use of the system as it stands. Further work is required in order to minimise the deleterious effects of linear acceleration and magnetic field distortion on orientation estimation. On going work to estimate linear acceleration based on physical modelling of the body is discussed in Section 8.2. No solutions to the problem of non-transient magnetic field distortions have been identified at the current time.

Preliminary work using raw rate gyroscope data on patients with prosthetic limbs has indicated that looking at rotational rate may be useful in assessing the optimal adjustment of prostheses. These results agree with the existing trend within the body sensor network community of using inertial measurement units to classify motions and to distinguish pathological conditions.

Real time gathering of results, with latencies measured in milliseconds, do not provide a significant advantage for the majority of diagnostic applications. However, real time visualisation is seen as being of use in applications such as physiotherapy where a patient is expected to perform a repeated motion. As these motions are typically slow, the major errors introduced by linear acceleration are less prevalent. However, usability of the system, for subjects with limited dexterity or range of motion, would need to be improved. One option would be to mount devices in clothing, simplifying the attachment of devices and preventing sensor transposition errors. Wireless devices in clothes present a potential reliability improvement over wired devices that may fail due to mechanical or electrical failure of interconnects. Battery recharging would, however, remain a serious issue.

While research into motion classification, based on limited numbers of devices, may present the most fertile direction for short term work, the ability to perform full motion tracking, allowing the use of existing bodies of knowledge about bio-mechanics, is an attractive goal. Discussions with clinicians at the Astley Ainslie hospital indicate that the concept of performing motion capture outside the confinement of laboratory remains an extremely exciting prospect.

7.5.5.3 Suitability for Other Applications

In addition to the motivating applications introduced in Section 1.3, the system has been considered for Human Computer Interaction (HCI) applications including games and development of virtual musical instruments [24, 25]. The low latency of the system has been sufficient to allow subjects to interact without feeling disconnected from the response.

Musical applications, using the positions of the limbs to modulate sound production, have proved quite successful, as, in general, the motions were quite slow and without the linear accelerations that result in large orientation estimation errors. More dynamic motions have also been investigated using raw sensor data from fewer devices, along with pattern recognition algorithms, to trigger sounds based on gesture recognition.

Use of the system for game interaction has been demonstrated by two simple games developed as group projects by honours year students. These games have demonstrated that the *Orient* devices can be used for HCI applications. One game involved throwing a virtual ball at a moving target, the other a two player boxing game. The boxing game demonstrated a difficulty of using the system for interaction with virtual objects, as the player had no form of haptic feedback on contact. The use of vibrotactile feedback is being considered to overcome this limitation.

Finally the suitability of the *Orient* devices to track non-human motion has been considered. In homage to the work of Eadweard Muybridge, a network of 10 devices was attached to a horse to attempt to track its spine and hind legs. Two significant problems were encountered: device alignment and observability of the skeletal structure. Firstly, as a non-human subject cannot be relied upon to hold a specific stance on command, devices must be manually aligned with the underlying bone structure. This complicates use of the system as sensors must be placed exactly in predetermined reference positions rather than attached freely. Secondly, as the *Orient* system uses a forward kinematic model for reconstructing the posture of the subject, it is necessary to observe the rotations of all major joints. In the case of horses, use of the system as it stands was concluded to be impossible, as the femur does not extend outside the main body. As the orientation of the femur could not be accurately observed it was not possible to accurately track the positions of the remaining joints.

7.6 Summary

Three possible approaches have been proposed for wirelessly-networked motion capture systems. The semi-distributed model has been shown to require the lowest data transmission rate per device. This translates to lower system latency and therefore a greater number of devices can be supported by the network.

The use of local processing to reduce data transmission is vital to achieving the goal of full body wireless motion capture. Local processing also results in increased tolerance of intermittent packet loss within the radio network as all dependencies between consecutive updates are removed.

The majority of the system's latency is due to the delay in gathering of orientation data from all the devices over the wireless network. The system allows a trade-off to be made between the latency and the number of devices. The overall system latency for full-body tracking of approximately 20ms is more than adequate for applications ranging from gait analysis to animation. Applications requiring lower latency can make use of multiple radio channels to reduce network delay. Future work in radio network architecture, discussed in Chapter 8, may allow for simultaneous multi-channel radio using a single receiver.

The ability to form networks of multiple devices and perform real time posture tracking has been demonstrated over many hours of system use. In subjective tests the system is able to capture the motion of a subject in a natural manner. While motion artefacts are sometimes noticeable, especially after prolonged and rapid motion, the system is useable for character animation. However, the inability to quantify the uncertainty of the angular and positional estimates precludes use in medical applications at the current time.

Part IV

Ongoing Work & Conclusions

Chapter 8

Ongoing and Future Work

This chapter presents areas identified as requiring future work. Short term work concentrating on system accuracy will be covered as well as more ambitious plans for the future.

8.1 *Orient* Development

8.1.1 Calibration procedure development

As seen in Chapter 6 the orientation accuracy is limited by the calibration of the scaling and offset of the individual sensors. The current method of hand calibration is limited by the ability of the calibrator to accurately perform the required calibration motions. It is proposed that using the PTU to perform automated calibration would provide improvements.

Automated calibration would allow for estimation of the full affine transform mapping individual sensor axes to device axes.

8.1.2 Hardware Development

The experience of using the *Orient* system has identified several areas in which the design could be improved. An updated device has been developed in conjunction with Martin Ling, a fellow member of the Speckled Computing Group at University of Edinburgh. The updated device features numerous improvements to the design to help improve accuracy, increase dynamic range, and increase ease of use.

To improve the accuracy of the system the power supplies were reworked to provide better noise performance. Temperature monitoring support, built into the AXRS300 rate gyroscopes, was made available to the processor to aid in bias correction. PCB layout was modified to increase separation between analogue and digital components.

The *Orient-2* device was limited to a maximum rotational rate of 1200° per second by the ADXR300 gyroscopes. An alteration to the power supply circuitry, powering the internal

MEMS resonator, allows for potentially much greater dynamic range to be achieved [78].

Alterations were made to the device to improve the general ease of use. The battery charger, originally requiring an external PCB, was moved onto the mainboard. This allowed for simpler recharging requiring only a single 5V supply. Additionally, reverse polarity protection was added to the design to prevent damage to the device caused by accidental reversal of the power supply. Finally, the design was modified to allow it to be more easily encased to reduce damage due to physical wear and tear sustained through repeated use.

The major flaw in the current design is the three-dimensional jigsaw nature of the device. This prevents fully automated assembly as the final construction must be done by hand. The solder joints, providing both electrical and mechanical connections between boards, have proved to be the greatest cause of device failure. Even when the device is protected by the external casing the joints are subject to stress.

New sensors have become available since the original *Orient* was designed that may allow for significant further improvements to be made. In particular new rate gyroscopes from InvenSense Inc. [83] and ST Microelectronics [84] allow for increased range in rotational rate sensing and multiple axis sensing on a single board. Incorporating these new sensors into a subsequent design would allow for longer device lifetimes, simpler PCB layout and improved reliability.

It is expected that further developments in MEMS sensor technology will continue reduce costs as MEMS sensors enter more products. This effect can already be seen since the introduction of the Nintendo Wii.

8.2 Linear Acceleration Estimation

The greatest limit on the accuracy of the *Orient* system is the effect of linear accelerations corrupting the orientation estimate. As was demonstrated in Section 4.1.2.3, linear accelerations result in errors even with perfect sensor calibrations. Current work is therefore focussed on estimating and cancelling linear accelerations within the body model.

8.2.1 Body Model Structure

The model currently under investigation is illustrated in Figure 8.1. The model consists of joints, J_i, and sensors, S_i. Each joint is a perfect spherical joint, in other words it can rotate in any direction about a single point. Furthermore, each joint has a local co-ordinate frame attached to it, the rotation of the joint therefore being the rotation of the local co-ordinate frame. Sensors are assumed to exist at a fixed point in a joint's co-ordinate frame, their position given by a constant offset vector, $\vec{o}_{S_i}$, from the joint centre. Joints form a tree structure with the position of a child joint given by an offset, $\vec{o}_{J_i}$, from the parent joint centre.

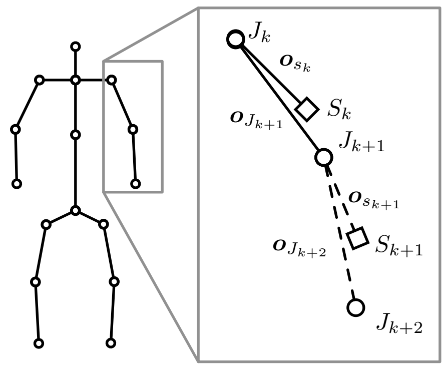


Figure 8.1: Model for Estimating Linear Accelerations

The equation for the linear acceleration of a point at an offset $\vec{o}$ from a fixed joint centre was derived in Section 4.1.2.3 as:

$$\vec{l} = \vec{l}_t + \vec{l}_r = \vec{o} \times \vec{\alpha} + (\vec{\omega} \cdot \vec{o})\,\vec{\omega} - \vec{o}\,\|\vec{\omega}\|^2 . \tag{8.1}$$

The resulting acceleration can be converted to the global world co-ordinate frame by rotating the joint local frame into the world frame using the estimated joint orientation q_{J_k}:

$$\begin{aligned} \vec{l}^W &= q_{J_k} \vec{l} q_{J_k}^* \\ &= q_{J_k} \left(\vec{\alpha} \times \vec{o} + (\vec{\omega} \cdot \vec{o})\,\vec{\omega} - \vec{o}\,\|\vec{\omega}\|^2 \right) q_{J_k}^*, \end{aligned} \tag{8.2}$$

where a “W” superscript indicates world co-ordinates.

Extending this result to the model of Figure 8.1 requires considering the effect of the accelerations due to the motion of ancestor joints. The acceleration of the joint J_{k+1} can be calculated from Equation 8.1. This acceleration applies to the entire co-ordinate frame of the joint. The accelerations of points in the J_k co-ordinate frame are therefore the superposition of the linear acceleration and the rotational kinematics of the frame:

$$\vec{l}^W = \vec{l}^W_{J_k} + q_{J_k} \left(\vec{\alpha} \times \vec{o} + (\vec{\omega} \cdot \vec{o})\,\vec{\omega} - \vec{o}\,\|\vec{\omega}\|^2 \right) q_{J_k}^*. \tag{8.3}$$

The value of l_{J_k} can be pre-calculated using Equation 8.3 for joint J_{k-1} where $\vec{o}$ is taken as the offset of J_k from the parent joint J_{k-1} in the parent co-ordinate frame. The resulting linear acceleration estimate must be rotated into the local co-ordinate frame of the device using the latest orientation estimate in order to subtract it from the measured acceleration.

8.2.2 Linear Acceleration Estimation Algorithm

From Equation 8.3 the linear accelerations of all sensors in the model can be estimated given the linear acceleration of the model root and the rotational kinematics of each model joint. This process can be applied recursively, working from the root of the body model hierarchy.

Equation 8.3 defined the linear acceleration, in world co-ordinates, of a point in terms of its offset from the point of rotation, $\vec{o}^J$, where the "J" superscript indicates joint co-ordinates; the angular velocity, $\vec{\omega}^J$; the angular acceleration, $\vec{\alpha}^J$; the orientation of the joint co-ordinate frame, q; and the linear acceleration of the joint, $\vec{l}_J^W$.

The orientation of each joint can be estimated based on the estimated orientation of an attached sensor, $\hat{q}_s$, and an alignment quaternion, q_a, specifying the alignment between the joint and sensor co-ordinate frames:

$$\hat{q} = \hat{q}_s q_a. \tag{8.4}$$

The angular rate and angular acceleration in the joint co-ordinate frame can be derived from the angular rate and angular acceleration measured by the sensor in the sensor local co-ordinate frame:

$$\vec{\omega}^J = q_a^* \vec{\omega}^S q_a, \tag{8.5}$$

$$\vec{\alpha}^J = q_a^* \vec{\alpha}^S q_a. \tag{8.6}$$

The estimated linear acceleration of the sensor, $\hat{\vec{l}}_s$, is calculated by applying Equation 8.3. Rotating the result into the sensor co-ordinate frame, and simplifying:

$$\hat{\vec{l}}_s^S = \hat{q}_s^* \vec{l}_J^W \hat{q}_s + \vec{\alpha}^S \times \vec{o}_s^S + \left(\vec{\omega}^S \cdot \vec{o}_s^S\right) \vec{\omega}^S - \vec{o}_s^S \left\|\vec{\omega}\right\|^2, \tag{8.7}$$

where $\vec{o}_s^S$, the offset of the sensor from the joint in the sensor co-ordinate frame, can be pre-calculated as:

$$\vec{o}_s^S = q_a \vec{o}_s^J q_a^*. \tag{8.8}$$

It should be noted that the linear acceleration of the joint, l_J^W, is received in the shared world co-ordinate frame and converted to the sensor local frame by applying the estimated sensor orientation, $\hat{q}_s$.

Similarly, the linear accelerations of child joints can be calculated directly from the sensor frame:

$$\vec{l}_{J_i}^W = \vec{l}_J^W + \hat{q}_s \left(\vec{\alpha}^S \times \vec{o}_{J_i}^S + \left(\vec{\omega}^S \cdot \vec{o}_{J_i}^S\right) \vec{\omega}^S - \vec{o}_{J_i}^S \left\|\vec{\omega}\right\|^2\right) \hat{q}_s^*. \tag{8.9}$$

In this case, the linear acceleration of the child joints is converted to the world co-ordinate frame before transmission.

A typical IMU, equipped with rate gyroscopes, is unable to directly measure the angular acceleration. The angular acceleration can be calculated using a first central difference approximation:

$$\vec{\alpha}(t) = \frac{\vec{\omega}(t+1) - \vec{\omega}(t-1)}{2\Delta t}. \tag{8.10}$$

Use of the first central difference results in lower error than the first backward difference approximation, and allows evaluation of the derivative at each sample, rather than between samples. However, central difference approximation requires knowledge of future samples.

The solution to this problem is to delay the calculation of the angular acceleration until the required data is available. This means that the estimate of $\vec{\alpha}$, and hence the linear acceleration, is delayed by at least one sample.

In order to update the linear accelerations of both the sensor and child joints, using Equations 8.7 and 8.9, it is necessary to know the linear acceleration of the joint. This information must have been previously calculated by the parent joint's sensor. Estimates of linear acceleration are therefore performed by a pre-order traversal of the body model structure. Each sensor estimates its own linear acceleration and the linear accelerations of all child joints. This data is transmitted so that subsequent calculations can be performed.

The complete filter algorithm, running on each sensor in the network, is presented as Algorithm 3. The algorithm is initialised by gathering two sample sets from each of the device sensors. These results are used to initialise buffers employed later in the algorithm. The initial orientation of the sensor is estimated directly from the observed gravitational and magnetic field vectors.

Once running, the filter is updated for each new sample set. First the estimated orientation of the device is updated using only the rate gyroscope data. This estimated orientation is transmitted immediately to minimise the latency in reconstructing the model posture.

The sensor then waits to receive an estimated linear acceleration for its associated joint from its parent in the body model tree. This linear acceleration is delayed by one sample period in order to calculate the value of $\vec{\alpha}$. Having received a linear acceleration estimate the sensor can then estimate its own linear acceleration using Equation 8.7. The estimated linear acceleration is then used to produce a vector observation estimate of the orientation as it was one sample period previous to the current sample. This vector observation is used to perform drift correction of the estimated orientation. Finally the rate gyroscope data is re-applied to produce the drift-corrected orientation estimate for the current sample.

After correcting its estimated orientation the sensor sends an updated linear acceleration estimate to each of its children in the body model. The estimated linear accelerations are calculated using Equation 8.9.

8.2.3 Discussion

Algorithm 3 enables each sensor to estimate its linear acceleration based on its measured rotational kinematics, and a linear acceleration received from its parent joint. The requirement, of receiving a linear acceleration estimate from the parent joint before transmitting linear accelerations of child joints, imposes a constraint on network scheduling. To minimise latency the devices should transmit in the same order in which they would appear in a pre-order traversal of the body model.

All linear accelerations passed between sensors are taken to be in the shared world co-

Algorithm 3: Acceleration estimation filter

```
begin Initialise
    $\vec{a}_0, \vec{m}_0, \vec{\omega}_0 =$ GetNextSample();
    $\vec{a}_1, \vec{m}_1, \vec{\omega}_1 =$ GetNextSample();
    $\hat{q}_0 =$ VectorObs($\vec{a}_1, \vec{m}_1$);
end
while running do
    $\vec{a}_t, \vec{m}_t, \vec{\omega}_t =$ GetNextSample();
    begin Inertial orientation update
        $\dot{q} = \frac{1}{2}\omega\hat{q}_{t-1}$;
        $\hat{q}_t = \hat{q}_{t-1} + \dot{q}\Delta t$;
        Send($\hat{q}_t$);
    end
    if Receive($\hat{\vec{l}}^W_{J_{t-1}}$) then
        begin Estimate previous linear acceleration
            $\vec{\alpha}_{t-1} = (\vec{\omega}_t - \vec{\omega}_{t-2})/2\Delta t$;
            $\hat{\vec{l}}_t = \vec{\alpha}_{t-1} \times \vec{o}_s$;
            $\hat{\vec{l}}_r = (\vec{\omega}_{t-1} \cdot \vec{o}_s)\,\vec{\omega}_{t-1} - \vec{o}_s \,\|\vec{\omega}_{t-1}\|^2$;
            $\hat{\vec{l}}_{t-1} = \hat{q}^*_{t-1}\hat{\vec{l}}^W_{J_{t-1}}\hat{q}_{t-1} + \hat{\vec{l}}_t + \hat{\vec{l}}_r$;
        end
        begin Apply drift correction
            $q_{vo} =$ VectorObs($\vec{a}_{t-1} - \hat{\vec{l}}_{t-1}, \vec{m}_{t-1}$);
            $\hat{q}_{t-1} = \hat{q}_{t-1} + \frac{1}{k}(q_{vo} - \hat{q}_{t-1})$;
            $\hat{q}_t = \hat{q}_{t-1} + \dot{q}\Delta t$;
            $\hat{q}_t = \hat{q}_t / \|\hat{q}_t\|$;
        end
        foreach child $J_i$ do Update child joints
            $\hat{\vec{l}}_t = \vec{\alpha}_{t-1} \times \vec{o}_{J_i}$;
            $\hat{\vec{l}}_r = (\vec{\omega}_{t-1} \cdot \vec{o}_{J_i})\,\vec{\omega}_{t-1} - \vec{o}_{J_i} \,\|\vec{\omega}_{t-1}\|^2$;
            $\hat{\vec{l}}^W_{J_{i_{t-1}}} = \hat{\vec{l}}^W_{J_{t-1}} + \hat{q}_{t-1}(\hat{\vec{l}}_t + \hat{\vec{l}}_r)\hat{q}^*_{t-1}$;
            Send($\hat{\vec{l}}^W_{J_{i_{t-1}}}$);
        end
    end
end
```

Table 8.1: Filter implementation costs

Filter	Estimated cycles	Static memory
Existing *Orient*	1066	8B
Linear acceleration estimation	$1470+231N$	$(36+6N)$B

ordinate frame. This allows sensors to interpret linear accelerations without knowledge of the orientations of neighbouring joints in the body model. This reduces the amount of data that must be shared between sensors. However, the new algorithm does require a 75% increase in transmitted data between devices. This additional data is not required for posture reconstruction, so one area of investigation is to make use of a second low-power radio channel for transmission of linear acceleration estimates. Using the rapid channel selection of the CC1100 the network traffic could be split into a two channel system with only relatively minor alteration to the existing TDMA networking code.

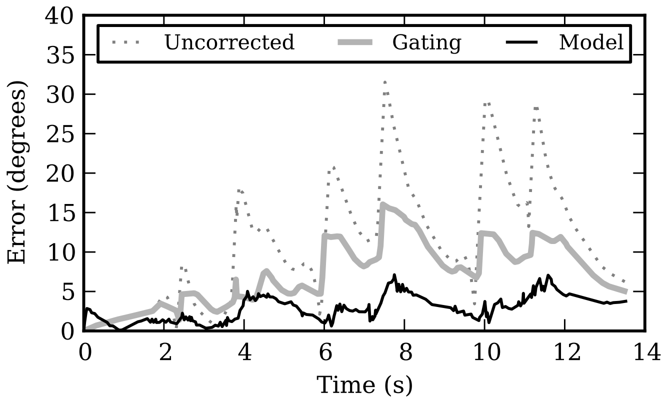
The algorithm has been analysed to estimate the required increases in processing and memory. The result of this analysis is presented in Table 8.1. The number of processing cycles is dependent on the number of child joints, N. In the existing human body model N is never greater than 3. The new algorithm therefore requires approximately double the processing resources.

The root of the body model, which obviously does not have a parent joint, represents a special case. The linear acceleration of the root joint must be estimated in an alternative manner. The appropriate choice of root joint should allow linear accelerations to be minimised. For example, the head undergoes less acceleration than the torso during walking [85]. Alternatively the linear acceleration of the root may be estimated in combination with other data sources, such as ultrasound, or step-tracking, or through local estimation based on previous values [86].

The decision to transmit estimated orientations based on limited integration of rate gyroscope data represents a tradeoff between latency and accuracy. Pure rate gyroscope integration leads to increasing uncertainty in the estimated orientation due to accumulated gyroscope bias and scale factor errors. However, the potential for error is small given the limited number of samples accumulated.

8.2.4 Proof of Concept Experiment

To demonstrate the concept of the new algorithm, calibrated sensor data was captured from two devices, one on the upper-arm and one on the forearm. As in the optical motion capture experiment of Section 6.5, the devices had optical markers attached to allow estimation of the true orientation. Additional markers were placed at the shoulder, elbow and wrist, to allow estimation of body model proportions. The motion captured was of a sitting subject raising



	Uncorrected	Gating	Model
RMS error (degrees)	13.56	8.19	3.25

Figure 8.2: Error Angle Comparison for Non-gated, Gated, and Linear Acceleration Estimation Algorithms

their arm from being held vertically by their side to having the upper-arm held horizontally in front of the body with the forearm vertical. The subject held this position for approximately two seconds before returning to the initial position. This motion was repeated three times.

Data from the experiment was used to drive three simulated orientation estimation algorithm implementations:

Uncorrected Direct conversion algorithm with accelerometer gating disabled. $(k = 128, a_T = \infty)$

Gating Direct conversion algorithm with accelerometer gating enabled. $(k = 128, a_T = 0.1g)$

Model Linear acceleration estimation algorithm. Limb segment lengths and sensor offsets estimated from initial optical frame. $(k = 128)$

The error angles between the reference quaternion calculated from the optical data and the output of each of the orientation filters for the forearm sensor are shown in Figure 8.2. The new algorithm can be seen to perform substantially better than the current gated orientation algorithm. Both the gated and linear acceleration estimation perform better than the non-gated algorithm.

8.2.5 Topics for Future Work

The result from the proof of concept experiment demonstrates that the new algorithm has potential, however, substantial further work is required to characterise its operation and stability. In particular investigation into the following questions is required:

- Is the algorithm convergent when the orientation estimate is perturbed?

  As the algorithm requires the use of local orientation estimates to convert to and from world co-ordinates, will it continue to converge when the local estimate is perturbed.

- What is the effect of packet loss?

  Without linear acceleration updates the orientation estimate will diverge from the true value due to integration of gyroscope bias.

- What is the required update rate of linear acceleration estimates?

  Currently the system performs drift correction on every filter update. However, reduced rate drift-correction has been previously demonstrated to not substantially increase orientation error [7].

- How to accurately calibrate the body model to the subject?

  The current system can operate with an approximate body model as the lengths of segments are only used to reconstruct the posture. In the new algorithm, knowledge of the offsets of joints and sensors is vital to accurate estimation of linear accelerations.

- Can the uncertainty in orientation accuracy be derived?

  The orientation accuracy of the current orientation estimation algorithm is not well defined as it depends on the specific motion of the subject. As the new algorithm attempts to compensate for this motion induced error, can its uncertainty be accurately estimated?

- What is the effect of rate gyroscope noise on estimation of angular acceleration?

  As demonstrated in Section 7.4.3.3, numerical differentiation results in amplification of noise. It will be necessary to investigate the effects of this noise on the accuracy of linear acceleration estimates.

8.3 Spatial Tracking

The *Orient* system as it stands has no knowledge of the spatial relationships between sensors. All spatial information is generated by applying rotation data to a rigid body model. The system is limited to estimating the appearance of the subject but cannot provide information about the location of the subject within an environment. While this is acceptable for many applications, there is scope for enhancement. The next logical step would be to make the system aware of spatial relationships.

Making the system capable of tracking both the posture and position of a subject would open up many more applications but also presents many interesting research topics. In particular, the addition of spatial relationships would be of great benefit to the estimation of linear accelerations. This section presents an overview of the identified problems and suggests possible methods for their solution.

As with posture tracking, spatial tracking can be achieved with or without external infrastructure. In keeping with the philosophy of producing an inexpensive and simple-to-use system it will be desirable to reduce the external infrastructure to a minimum.

8.3.1 Location Tracking in Wireless Sensor Networks

Significant work within the field of Wireless Sensor Networks has concentrated on the problem of location estimation [87]. Approaches include basic radio connectivity [88, 89], Received Signal Strength Indicator (RSSI) based ranging [90], acoustic time of flight [91, 92], and optical methods. It is beyond the scope of this thesis to examine these in any great detail. However, most WSN location schemes assume large numbers of nodes, which may not be desirable for a motion tracking system.

8.3.2 Traditional Motion Capture

Two common spatial motion capture systems, suitable for use both indoors and outdoors, are optical triangulation and ultrasonic multi-lateration.

Use of an optical system is based on the same principle as in the case of full optical motion capture. However, as the posture of the subject is tracked using the *Orient* devices the optical system only needs to track a single point on the subject. This should reduce the number of required markers and cameras, which reduces the cost and the complexity of the system. Additionally, the resolution of the optical system could be reduced, allowing for larger markers which increases the tracking area.

A proof of concept system, combining a single overhead camera with the *Orient* network, has been developed as part of an honours year project by Aris Valtazanos.

Ultrasonic systems directly provide distance measurements that can be used for multi-lateration to locate three-dimensional points. As with optical tracking, the ultrasonic system only needs to track a single point on the subject.

Both optical and ultrasonic tracking systems require at least two line-of-sight links with the subject at all times to maintain an accurate track. The combination of inertial posture tracking with infrastructure-based location tracking presents an interesting possibility. As the location system only needs to track a single point, the positions, and number, of cameras or ultrasonic modules can be altered. Moving cameras to a position above the tracking area may provide a less obstructed view in many cases, while still supplying sufficient information about the subject's location. This allows the possibility of tracking multiple subjects that would normally pose a significant problem with full optical motion capture due to occlusions as subjects interact.

An alternative tracking system would be the use of Global Positioning System (GPS) data. However, this is limited to open areas and may not provide sufficient resolution for some applications.

8.3.3 Dead Reckoning

An alternative approach, requiring no external infrastructure, is to use only data available within the existing inertial system. This approach has the advantage that it reduces the cost of setting up the system but has a significant disadvantage that there is no common global reference frame. This in turns means that drift error minimisation is the key to accurate tracking.

Simple dead reckoning systems are based on counting the number of steps that a subject has taken and tracking their direction [93]. This gives an estimate of the distance travelled provided the subject's stride length is regular. This technique may be improved by adding an estimate of the stride length [94].

Dead reckoning based on stride length is of limited use when tracking a subject over rough terrain, or other varying conditions, where the stride length is unlikely to be uniform. Inertial dead reckoning may overcome this, as it makes no assumptions about stride lengths, but estimates position based on double integration of accelerometer data. Bias errors in acceleration data produce errors that grow quadratically in the output when doubly integrated. When using three-dimensional acceleration data, it is necessary to remove the static acceleration due to gravity. This requires an accurate orientation estimate as even small errors in orientation will lead to large errors in position estimation. A 1 milliradian bias error in orientation corresponds to a 1 milli-g acceleration which when integrated twice to produce a displacement will quickly add up to a sizeable error [95]. Reduction in inertial estimation error can be achieved by detecting heel strikes and zeroing velocity when the foot hits the ground [96].

8.3.4 Complementary Approaches

As with the orientation estimation problem a complementary approach could be taken to combine data from different position tracking methods. Once again the Kalman filter provides a good framework with which to build a position estimation filter combining data from multiple different sources. The earlier objections to the Kalman filter based on its complexity are somewhat mitigated in the case of position tracking. As we require only one filter and may allow ourselves a somewhat larger device to perform processing we allow the possibility of implementing a more sophisticated filter.

8.4 Multiple Subject Interaction

Tracking the motion of multiple subjects interacting is an interesting area for future development, which presents many challenges with the opportunity to improve on existing solutions. Optical systems suffer from occlusions making tracking of interacting characters difficult in even ideal situations. More complex environments, such as trying to track the motion of the members of a crowd, are difficult to solve with existing optical methods. This problem is illustrated in Figure 8.3 where the subject in the middle of the diagram has no line-of-sight links to the cameras mounted in the corners. Adding more cameras provides only a temporary relief until more obstructions are added.

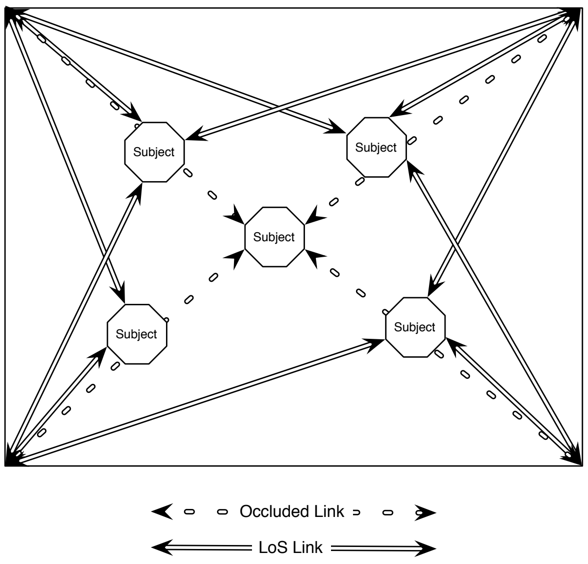


Figure 8.3: Occlusion Problem in Multi-subject Tracking

As inertial tracking does not rely on any infrastructure or line-of-sight links it presents a possible solution to tracking people interacting in complex environments. However, there are still substantial difficulties that must be overcome.

8.4.1 Shared Co-ordinate Frame

The first major problem is how to establish a shared co-ordinate frame between all the subjects. This is required in order to provide information and spatial relationships between the subjects. The *Orient* system provides orientation in a shared frame, ignoring standard provisos about magnetic field distortion, but cannot currently provide any spatial data. Infrastructure-less spatial methods, such as step-counting and inertial tracking, provide only relative position to a known starting location. Other methods rely on multiple line-of-sight links.

One interesting approach would be to form a mesh between subjects using a standard ranging method such as differential time of flight between radio and ultrasound transmissions. Given multiple receivers, with known displacements, it would be possible to localise the transmitting device. The *Orient* system already requires the subject to wear multiple sensor devices to provide posture tracking. Adding ultrasonic transceivers to the platform would provide the basis for inter-subject distance measurements. As the posture tracking algorithm provides information about sensor displacement and the TDMA networking layer provides accurate timing, it would be possible to calculate the translation of a transmitter. This requires only that there be line-of-sight links between the subjects rather than to external infrastructure, such as cameras. This would greatly reduce the incidence of subject occlusion. Figure 8.4 illustrates

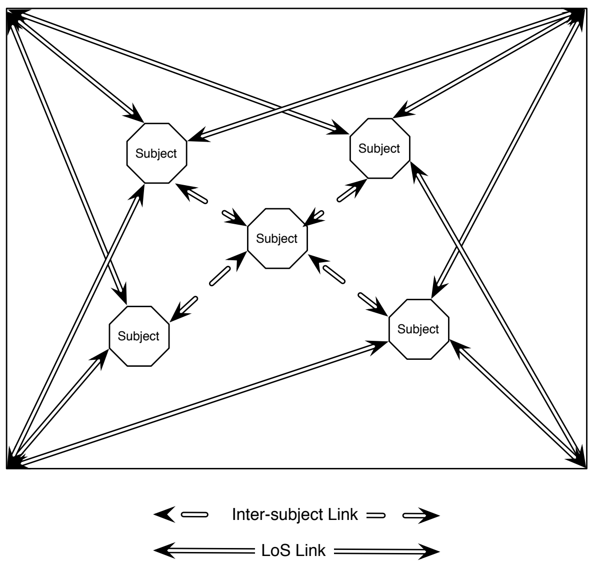


Figure 8.4: Overcoming Occlusions using Inter-subject Meshing

the resulting tracking solution where the central subject is localised using links to other subjects in the system. When combined with self-contained tracking solutions it may not even be necessary to maintain constant mesh links. Self-contained solutions, such as foot-step tracking, can maintain reasonably accurate tracking over short periods of time when ranging information is not available. This approach is commonly applied in complementary inertial/GPS systems, where the inertial track fills in for missing data from the GPS track.

8.4.2 Network Architecture

As seen in Chapter 7, the current system latency and update rate is limited by the network delay involved in gathering data from all the *Orient* devices. This delay is caused by the necessity to serialise transmission of data over a single radio channel. Reducing this delay would substantially decrease the overall system latency. This would be particularly important when tracking multiple interacting subjects.

This section will briefly outline a possible future radio network architecture, based on discussions with members of the Digital Signal Processing Enabled Communications (DSPeC) group at Strathclyde University, partners in the Speckled Computing Consortium, that could help alleviate the transmission delay. Two possible interaction scenarios, and their effect on network architecture, will be considered.

8.4.2.1 Future Radio Network Architecture

Two obvious approaches exist for reducing the network transmission delay. The first approach, of increasing the bandwidth of the shared channel has an associated cost in terms of transmission and reception power. Additionally, as the slot time assigned to individual devices is reduced the tolerance for synchronisation also decreases. This approach increases the cost for both the sensor devices and central node.

An alternative approach, adopted by current commercial systems such as the InterSense InertiaCube3, is to use multiple frequency channels to multiplex devices. This takes advantage of the asymmetrical nature of the network to allow greater resources to be utilised at the central point where richer resources will typically be available. Currently this requires multiple receiver devices, which increases the cost as much of the hardware must be duplicated in each receiver device. Developments in Digital Signal Processing (DSP) can allow for simultaneous reception of multiple time-synchronised narrowband signals using a single wide-band radio front-end and processor.

The technique of Orthogonal Frequency Division Multiple Access (OFDMA) potentially allows for low-cost narrowband transmitter designs to be used in the sensor devices helping to maintain low-power requirements. The central receiver cost increases, in terms of signal processing, but requires fewer components than using multiple narrowband receivers.

Using OFDMA presents challenges as devices must still maintain synchronisation, both for network performance and for consistency of captured posture updates. This requires that devices, while they transmit on uniquely-assigned channels, must be able to receive on a common channel. Additionally, the radio front end-must be designed to support multiple transmission channels, potentially requiring relatively wide-band antennas and output filters.

8.4.2.2 Interaction Scenarios

There are two possible interaction scenarios to consider when attempting to track multiple subjects. These can be described as internal and external interaction.

In internal interaction, one is interested in the interactions directly between subjects. An example of this type of interaction would be tracking the movements of a couple dancing. In this scenario one would be interested in contact between the users as they move together. In order to track this accurately snapshots are taken of all the tracked subjects. Using the

Interaction	Intra-user	Inter-user	Number of Channels	Features
Internal	TDMA	FDMA	Number of users	Users synchronised, medium latency
External	FDMA	TDMA	Number of sensors per user	Users unsynchronised low latency

Table 8.2: Summary of Network Architectures

current single channel network implementation this would lead to very large latencies and low update rates as data must be gathered from each sensor from each subject in turn. Using a unique channel per subject offers improved performance by allowing data to be gathered from subjects in parallel. In this case the latency of tracking multiple subjects would be the same as the current *Orient* system tracking an individual, provided that enough orthogonal channels were available.

In external interactions one is interested in the interaction of multiple subjects with a single external source of events. An example of such an interaction would be musicians playing virtual instruments. In this scenario, the lowest latency is desired for each user in relation to the external source. Therefore, a single snapshot of all the subjects is not required, but instead discrete snapshots should be available. In this case data for each subject is gathered with as little latency as possible. To achieve this one could use multiple channels per subject, which could be time-multiplexed as a complete snapshot of all subjects is not required.

8.4.2.3 Summary

The possibility of using multiple radio channels to gather orientation data simultaneously enables motion capture of multiple subjects while maintaining reasonable latency and update rate performance. Two possible interaction scenarios have been identified based on the characteristics of the motion tracking application. Both scenarios make use of time and frequency division multiplexing to achieve optimal performance. Table 8.2 presents a summary of the possible network architectures and their intra- and inter-user communication patterns.

8.5 Summary of Directions for Further Research

The current research on wireless inertial motion tracking represents only the tip of an iceberg. Short term work should continue to work on platform design and filter implementation in order to improve the current accuracy. The linear acceleration estimation algorithm has been demonstrated to have potential in significantly decreasing errors. Further experimentation and simulation need to be performed to validate the new algorithm's effectiveness.

In the longer term the system should be expanded to deal with tracking multiple users in space. This presents many challenges in how to localise users in a global co-ordinate frame and how to scale communication as more devices and subjects are added.

To assess the performance of the system it should be subjected to formal application studies involving end users. These will help to further identify system performance issues and areas for future investigation.

Chapter 9

Conclusions

9.1 Summary

This research has developed an integrated, wireless, motion tracking system for capturing the movement of articulated structures in real time. The system supports tracking the orientation of multiple wireless sensors using a single, low-bandwidth, radio channel. The orientation of multiple devices attached to an articulated structure, such as the human body, allows realtime reconstruction of the three dimensional body within a virtual environment.

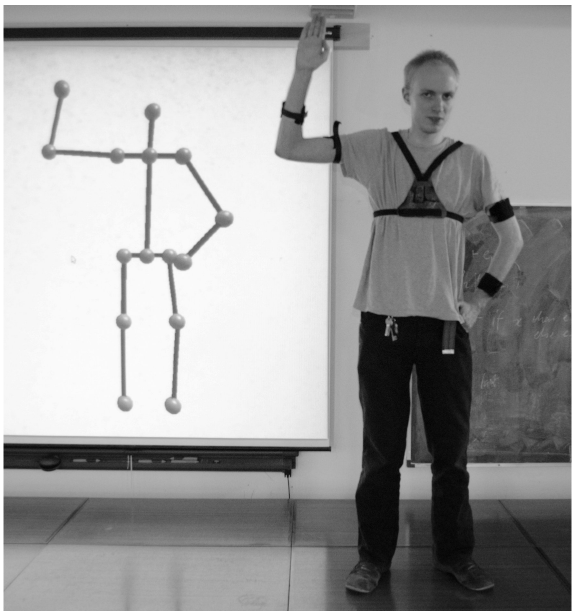


Figure 9.1: *Orient* Motion Capture System in Action

The *Orient* motion tracking system, illustrated in Figure 9.1, is the only known fully wireless, inertial, motion capture system capable of tracking the posture of a full human body in realtime.

The accuracy of the *Orient* devices have been tested for both static and dynamic conditions. Static accuracy is limited by the calibration accuracy of the accelerometers and magnetometers. Dynamic accuracy is limited by the effects of linear acceleration corrupting vector observations.

The correlation between subject motion and linear acceleration means that the dynamic accuracy cannot be usefully constrained for generalised motions. A new algorithm for estimating linear acceleration, and thereby mitigating its effect on dynamic accuracy, has been presented and demonstrated to have potential benefit.

9.2 Relation to Existing Works

The *Orient* system builds on work in the wireless sensor network community as well as work in traditional motion tracking solutions.

Within the wireless sensor network community many sensor devices have been developed that are capable of detecting motion. However, in general these devices do not have the required sensing capabilities to perform drift-corrected orientation estimation. Typical applications are limited to detecting and classifying motion through the use of devices equipped with accelerometers. Such devices are unable to capture motion with the fluidity and level of detail as the *Orient* system.

Advanced devices, including full inertial measurement functionality, have been developed by other groups. However, these devices rely on host processing to perform orientation sensing and have not been demonstrated to operate as an integrated full-body motion tracking solution.

Commercial inertial motion tracking solutions are available. However, existing implementation require sensors to be wired together in a body suit. The orientation estimation algorithms employed by such systems require significant processing power to implement which in turn requires a powerful host system to provide processing.

9.3 Key Developments

9.3.1 Low-Complexity Orientation Estimation

The most common estimation process used for inertial orientation calculations are derivatives of the Kalman filter structure. This filter structure is chosen as it can be shown to be optimal in terms of mean-square-error performance. However, the Kalman structure is relatively complex and requires significant processing power to implement.

The Kalman filter structure relies on a model of the system dynamics to predict accurately future states. Producing such a model for human motion would be a complex problem, as there are many possible motions that can be performed, and each part of the body would require a different dynamic model.

The complexity of the Kalman filter implementations advocated previously has required that orientation processing be performed by a host system. As sensor updates include rate gyroscope data that must be integrated to perform orientation estimation, there is a dependence between updates. This in turn requires high data update rates between the sensor devices and the host, with a consequent demand for network bandwidth. Packet loss, a common problem in wireless networks, results in errors in orientation estimation that can take significant time to rectify.

The approach taken in this research is to return to the basics of the orientation estimation problem. The resulting filter, described in Section 3.3.6, is the complementary fusion of two very simple approaches. The low complexity of the filter allows it to be implemented locally on the sensing device. No assumptions are made about the expected dynamics which allows the same filter to be used in many applications without alteration.

9.3.2 Integrated Wireless Motion Tracking

The *Orient* system developed in this thesis provides a completely integrated motion tracking system.

The *Orient* sensor device is a state-of-the-art compact wireless inertial/magnetic measurement device capable of tracking its three-dimensional orientation in real-time. An individual device comprises processing, wireless networking, rechargeable battery and rich inertial sensing capabilities in a very compact package measuring just $35 \times 43 \times 16$mm, including casing.

Each *Orient* device is capable of tracking its own orientation relative to a shared global co-ordinate frame. Orientation updates are performed locally on each device at a rate of 256 updates per second. Empirical measurements of the device accuracy have demonstrated it to be capable of estimating its orientation to within one degree while stationary.

Synchronised time division networking allows multiple *Orient* sensors to be tracked in real-time over a single low bandwidth radio link. The data from multiple sensors mounted on the subject's body is passed to a forward kinematic body model to produce an estimated posture derived from individual orientation estimates.

The implementation of the body model allows the system to be used in a modular manner. As sensor devices are interchangeable they can be used in any position on the body. By altering the number of sensors, more or less complex models can be constructed to meet the demands of application requirements. The latency of the system, dominated by the networking latency, scales linearly with the number of devices being tracked.

Unlike existing inertial motion tracking solutions, as orientation estimation processing is distributed throughout the network, there is no need for a powerful central host. The ability to perform motion tracking without requiring a powerful host opens the possibility of truly autonomous motion tracking.

As the orientation estimation process is performed locally the network bandwidth can be reduced by sub-sampling the filter output. In addition, as network updates are independent of each other, the system provides greater resilience to wireless network packet losses.

9.3.3 Analysis of Orientation Algorithm Errors

The behaviour of the proposed orientation estimation algorithm has been examined. The effects of linear accelerations due to subject motion, are demonstrated to be the primary source of errors. The effects of linear accelerations, leading to corrupted vector observations and thereby increased orientation errors, are common to other existing filtering algorithms. These errors occur regardless of the accuracy of sensor calibration and performance.

The use of vector observation gating to reduce linear acceleration errors has been demonstrated to provide a minor improvement in accuracy. However, the simple gating process used is unable to reliably detect all inaccurate vector observations.

9.3.4 Linear Acceleration Estimation Algorithm

As linear accelerations due to subject motion are concluded to be the primary source of orientation errors, the focus of ongoing work has been to estimate linear accelerations. By subtracting a linear acceleration estimate from the measured acceleration vector a more accurate vector observation may be achieved. Furthermore, if linear accelerations can be accurately estimated then gyroscope drift correction can be performed even during rapid movement.

A novel distributed orientation estimation algorithm incorporating linear acceleration estimation has been developed. While this algorithm requires substantial further investigation its potential to reduce errors has been demonstrated.

9.4 Future Directions

The *Orient* motion tracking system provides the potential for realtime posture tracking of subjects over an almost unlimited area compared to the restricted range of optical methods. However, in its current implementation it is unable to provide accurate spatial tracking, which camera-based approaches can. Preliminary investigations, presented in Chapter 8, indicate several possible directions for further research to overcome this limitation.

Achieving accurate spatial tracking, with a minimum of external infrastructure, along with developments in network architecture, offer a vision of a fully wireless, distributed, motion tracking solution capable of tracking multiple users over large areas. Such a system would go far beyond the technical limitations presented by current state-of-the-art optical motion trackers.

Bibliography

[1] National Geophysical Data Center, Last accessed: 15 Feb 2010. URL `http://www.ngdc.noaa.gov/wist/magfield.jsp`.

[2] RB Martin. A Genealogy of Biomechanics. *23rd Annual Conference of the American Society of Biomechnanics.*, October 1999.

[3] Reinhard Klette and Garry Tee. Understanding Human Motion: A Historic Review. Technical Report 192, Communication and Information Technology Reasearch group, University of Auckland, 2007.

[4] Simon Bouisset. Etienne-Jules Marey : When the Study of Motion Becomes an Experimental Science, June 2005.

[5] D. Roetenberg. *Inertial and Magnetic Sensing of Human Motion*. PhD thesis, Universiteit Twente, 2006.

[6] E. Foxlin, M. Harrington, and Y. Altshuler. Miniature 6-DOF inertial system for tracking HMDs. *Proc. SPIE Helmet and Head-Mounted Displays III*, 3362, April 1998.

[7] Eric Robert Bachmann. *Inertial and Magnetic Tracking of Limb Segment Orientation for Inserting Humans into Synthetic Environments*. PhD thesis, Naval Postgraduate School, Monterey, California, 2000.

[8] G.J. Pottie. Wireless sensor networks. *Information Theory Workshop*, pages 139–140, June 1998.

[9] D. Estrin, L. Girod, G. Pottie, and M. Srivastava. Instrumenting the world with wireless sensor networks. *Acoustics, Speech, and Signal Processing, Proceedings of the IEEE International Conference on*, 4:2033–2036, 2001.

[10] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[11] H. Karl and A. Willig. A short survey of wireless sensor networks. *Telecommunication Networks Group Technische University at Berlin, Dec*, 2002.

[12] E.M. Tapia, N. Marmasse, S.S. Intille, and K. Larson. MITes: Wireless portable sensors for studying behavior. *Proceedings of Extended Abstracts Ubicomp 2004: Ubiquitous Computing*, 2004.

[13] E. Jovanov, A. Milenkovic, C. Otto, and P.C. de Groen. A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 2(6), 2005.

[14] R. Aylward, S. D. Lovell, and J. A. Paradiso. A compact, wireless, wearable sensor network for interactive dance ensembles. In *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, pages 4 pp.–, 2006.

[15] C. Park, P. H. Chou, and Y. Sun. A wearable wireless sensor platform for interactive dance performances. In *Pervasive Computing and Communications, Fourth Annual IEEE International Conference on*, pages 52–59, 2006.

[16] R.C. King, L. Atallah, A. Darzi, and G.Z. Yang. An HMM Framework for Optimal Sensor Selection with Applications to BSN Sensor Glove Design. In *Proceedings of the 4'th Workshop on Embedded Networked Sensors*, pages 58–62, June 2007.

[17] G. Ebersbach, M. Heijmenberg, L. Kindermann, T. Trottenberg, J. Wissel, and W. Poewe. Interference of rhythmic constraint on gait in healthy subjects and patients with early parkinson's disease: Evidence for impaired locomotor pattern generation in early Parkinson's disease. *Movement Disorders*, 14(4):619–625, 1999.

[18] Jacquelin Perry. *Gait Analysis - Normal and Pathological Function.* SLACK Incorporated, 1992.

[19] A. D. Young, Martin Ling, and D. K. Arvind. Orient-2: A realtime wireless posture tracking system using local orientation estimation. In *Embedded networked sensors, Proceedings of the 4th workshop on*, pages 53–57, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-694-3.

[20] A.D. Young. Comparison of orientation filter algorithms for realtime wireless inertial posture tracking. *Wearable and Implantable Body Sensor Networks, International Workshop on*, 0:59–64, 2009.

[21] A. D. Young and M.J. Ling. Minimising loss-induced errors in real time wireless sensing by avoiding data dependency. *Wearable and Implantable Body Sensor Networks, International Workshop on*, 0:327–332, 2009.

[22] A. D. Young, M. J. Ling, and D. K. Arvind. Distributed estimation of linear acceleration for improved accuracy in wireless inertial motion capture. In *Information Processing in*

Sensor Networks, Proceedings of the 9th The International Conference on, pages 256–267, 2010.

[23] A. D. Young. Use of body model constraints to improve accuracy of inertial motion capture. In *Wearable and Implantable Body Sensor Networks, Proceedings of the 1st International Conference on*, 2010.

[24] V. Lympouridis, M. Parker, A. D. Young, and DK Arvind. Sonification of Gestures Using Specknets. In *Sound and Music Computing Conference, Proceedings of the 4th conference on*, volume 7, pages 382–85, 2007.

[25] V. Lympourides, D. K. Arvind, and Martin Parker. Fully wireless, full body 3-d motion capture for improvisational performances. In *Whole Body Interaction, Proceedings of the workshop on*. SIGCHI, 2009.

[26] DK Arvind and A. Bates. The speckled golfer. In *Body Area Networks, Proceedings of the ICST 3rd international conference on*, page 28. Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 2008.

[27] DK Arvind and A. Valtazanos. Speckled Tango Dancers: Real-Time Motion Capture of Two-Body Interactions Using On-body Wireless Sensor Networks. In *Wearable and Implantable Body Sensor Networks, Proceedings of the International Workshop on*, pages 312–317. IEEE Computer Society, 2009.

[28] D.K. Arvind and M.M. Bartosik. Clustering of motion data from on-body wireless sensor networks for human-imitative walking in bipedal robots. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–6, June 2009.

[29] A. Bates, M. J. Ling, and D. K. Arvind. Respiratory rate and flow waveform estimation from tri-axial accelerometer data. In *Wearable and Implantable Body Sensor Networks, Proceedings of the 1st International Conference on*, 2010.

[30] Rong Zhu and Zhaoying Zhou. A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package. In *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, volume 12, pages 295–302, June 2004.

[31] *InertiaCube3 Datasheet*. InterSense, Inc.

[32] *MTx: 3DoF Orientation Tracker Datasheet*. Xsens Motion Technologies.

[33] *Inertia-Link Datasheet*. MicroStrain.

[34] *Moven Datasheet*. Xsens Motion Technologies.

[35] T. Harada, H. Uchino, T. Mori, and T. Sato. Portable orientation estimation device based on accelerometers, magnetometers and gyroscope sensors for sensor network. In *Multisensor Fusion and Integration for Intelligent Systems, Proceedings of IEEE International Conference on*, pages 191–196, 2003.

[36] T. Harada, H. Uchino, T. Mori, and T. Sato. Portable absolute orientation estimation device with wireless network under accelerated situation. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pages 1412–1417 Vol.2, 2004. ISBN 1050-4729.

[37] Tatsuya Harada, Taketoshi Mori, and Tomomasa Sato. Development of a Tiny Orientation Estimation Device to Operate under Motion and Magnetic Disturbance. *The International Journal of Robotics Research*, 26(6):547–559, 2007.

[38] X. Yun and E. R. Bachmann. Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking. *Robotics, IEEE Transactions on*, 22(6):1216–1227, 2006.

[39] J.L. Hill and D.E. Culler. Mica: a wireless platform for deeply embedded networks. *Micro, IEEE*, 22(6):12–24, Nov/Dec 2002. ISSN 0272-1732.

[40] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 364–369, 15 April 2005.

[41] H. Dubois-Ferrière, R. Meier, and P. Metrailler. TinyNode: a comprehensive platform for wireless sensor network applications. *Proceedings of the fifth international conference on Information processing in sensor networks*, pages 358–365, 2006.

[42] Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 95–107, New York, NY, USA, 2004. ACM. ISBN 1-58113-879-2.

[43] Kai-Juan Wong and D. K. Arvind. Speckmac: low-power decentralised mac protocols for low data rate transmissions in specknets. In *REALMAN '06: Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 71–78. ACM, 2006. ISBN 1-59593-360-3.

[44] Kai-Juan Wong and D.K Arvind. A hybrid wakeup signalling mechanism for periodic-listening mac algorithms. *Networks, 2007. ICON 2007. 15th IEEE International Conference on*, pages 467–472, Nov 2007. ISSN 1556-6463.

[45] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves. Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks. *Wireless Networks*, 12(1): 63–78, 2006.

[46] S. Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks*, 5(2):81–94, 1999.

[47] I. Rhee, A. Warrier, M. Aia, and J. Min. Z-MAC: a hybrid MAC for wireless sensor networks. *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 90–101, 2005.

[48] A. Lynch, B. Majeed, J. Barton, F. Murphy, K. Delaney, and S. O'Mathuna. A wireless inertial measurement system (WIMS) for an interactive dance environment. *Journal of Physics: Conference Series*, 15:95–100, 2005.

[49] Javier Torres, Brendan O'Flynn, Philip Angove, Frank Murphy, and Cian O' Mathuna. Motion tracking algorithms for inertial measurement. In *Body area networks, Proceedings of the ICST 2nd international conference on*, pages 1–8, ICST, Brussels, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[50] C. Park, J. Liu, and PH Chou. Eco: an ultra-compact low-power wireless sensor node for real-time motion monitoring. *Information Processing in Sensor Networks, Fourth International Symposium on*, pages 398–403, 2005.

[51] Jack B. Kuipers. *Quaternions and Rotation Sequences*. Princeton University Press, 5th edition, 2002.

[52] D. Baraff. An Introduction to Physically Based Modeling: Rigid Body Simulation I—Unconstrained Rigid Body Dynamics. *SIGGRAPH Course Notes*, 1997.

[53] C. V. C. Bouten, K. T. M. Koekkoek, M. Verduin, R. Kodde, and J. D. Janssen. A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity. *Biomedical Engineering, IEEE Transactions on*, 44(3):136–147, 1997.

[54] Yun Xiaoping, E.R. Bachmann, and R.B. McGhee. A simplified quaternion-based algorithm for orientation estimation from Earth gravity and magnetic field measurements. *Instrumentation and Measurement, IEEE Transactions on*, 57(3):638–650, March 2008. ISSN 0018-9456.

[55] G. Wahba. A least squares estimate of satellite attitude. *SIAM Review*, 7(3):384–386, 1965.

[56] FL Markley and D. Mortari. Quaternion attitude estimation using vector observations. *Journal of the Astronautical Sciences*, 48(2):359–380, 2000.

[57] MD Shuster and SD Oh. Three-axis attitude determination from vector observations. *Journal of Guidance and Control*, 4(1):70–77, 1981.

[58] Richard Boynton. Precise measurement of mass. In *60th Annual Conference of the Society of Allied Weight Engineers*, 204 Hubbard St, Glastonbury, CT, May 2001. S.A.W.E Inc.

[59] ESA GOCE mission website, Last accessed: 15 Feb 2010. URL `http://www.esa.int/esaLP/ESAK4XZK0TC_goce_0.html`.

[60] J. H. Poynting. *The mean density of Earth.* Charles Griffen & Co., London, 1894.

[61] G. Welch and G. Bishop. An introduction to the Kalman filter. *ACM SIGGRAPH 2001 Course Notes*, 2001.

[62] Dan Simon. Kalman Filtering. *Embedded Systems Programming*, June 2001.

[63] E. Foxlin. Inertial head-tracker sensor fusion by a complementary separate-bias Kalman filter. In *Virtual Reality Annual International Symposium, Proceedings of the IEEE*, pages 185–194, 267, 1996.

[64] *Application Note AN-203: Compass Heading using Magnetometers.* Honeywell, .

[65] S. Łuczak. Advanced algorithm for measuring tilt with MEMS accelerometers. *Recent Advances in Mechatronics*, pages 511–515, 2007.

[66] Xiaoping Yun, C. Aparicio, E.R. Bachmann, and R.B. McGhee. Implementation and experimental results of a quaternion-based Kalman filter for human body motion tracking. *Robotics and Automation, Proceedings of the 2005 IEEE International Conference on*, pages 317–322, April 2005.

[67] G.M. Lerner. *Spacecraft Attitude Determination and Control*, chapter Three-Axis Attitude Determination, pages 420–428. Kluwer Academic Pub, 1978.

[68] D. K. Arvind and K. J. Wong. Speckled Computing: Disruptive Technology for Networked Information Appliances. In *Proceedings of the IEEE International Symposium on Consumer Electronics*, pages 219–223, September 2004.

[69] *dsPIC 30F3014 Datasheet.* Microchip.

[70] *CC1100 Datasheet.* Chipcon.

[71] *Application Note AN039 - Using CC1100/CC1150 in European 433/868 MHz bands.* Chipcon.

[72] *CC2500 Datasheet*. Chipcon.

[73] *MMA7260Q Datasheet*. Freescale Semiconductor.

[74] *HMC 105x Series Datasheet*. Honeywell.

[75] *INA321 Datasheet*. Texas Instruments, July 2004.

[76] *TLE2426 Precission Virtual Ground Datasheet*. Texas Instruments, May 1998.

[77] *ADXRS300 Datasheet*. Analog Devices.

[78] *Application Note AN-625: Modifying the Range of the ADXRS150 and ADXRS300 Rate Gyros*. Analog Devices.

[79] NIST/SEMATECH e-handbook of statistical methods, Last accessed: 15 Feb 2010. URL `http://www.itl.nist.gov/div898/handbook/`.

[80] *PTU-D46-17 Specifications*. Directed Perception.

[81] *REG711 Datasheet*. Texas Instruments.

[82] *dsPIC 30F3014/4013 Rev. A1 Silicon Errata*. Microchip.

[83] *IDG-650 Datasheet*. InvenSense Inc.

[84] *LYPR540AH Datasheet*. STMicroelectronics, November 2009.

[85] J. J. Kavanagh, S. Morrison, and R. S. Barrett. Coordination of head and trunk accelerations during walking. *European Journal of Applied Physiology*, 94(4):468–475, 07 2005.

[86] H. Luinge and P. Veltink. Measuring orientation of human body segments using miniature gyroscopes and accelerometers. *Medical and Biological Engineering and Computing*, 43 (2):273–282, 04 2005.

[87] Radu Stoleru, John A. Stankovic, and Sang H. Son. Robust node localization for wireless sensor networks. In *EmNets '07: Proceedings of the 4th workshop on Embedded networked sensors*, pages 48–52, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-694-3.

[88] Nirupama Bulusu, John Heidemann, and Deborah Estrin. Gps-less low cost outdoor localization for very small devices. *IEEE Personal Communications Magazine*, 7(5): 28–34, October 2000.

[89] R. Nagpal. Organizing a global coordinate system from local information on an amorphous computer. AI memo 1666, MIT Artificial Intelligence Laboratory, 1999.

[90] K. Whitehouse, C. Karlof, and D. Culler. A practical evaluation of radio signal strength for ranging-based localization. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(1):41–52, 2007.

[91] J. Sallai, G. Balogh, M. Maroti, and A. Ledeczi. Acoustic Ranging in Resource Constrained Sensor Networks. *Proc. of International Conference on Wireless and Mobile Computing (ICWN), June*, 2004.

[92] L. Girod and D. Estrin. Robust range estimation using acoustic and multimodal sensing. *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, 3:1312–1320 vol.3, 2001.

[93] T. Judd. A Personal Dead Reckoning Module. *ION GPS*, 97:1–5, 1997.

[94] H. Weinberg. Using the ADXL202 in pedometer and personal navigation applications. *Application Notes AN-602, Analog Devices*, 2002.

[95] G. Welch and E. Foxlin. Motion tracking: no silver bullet, but a respectable arsenal. *Computer Graphics and Applications, IEEE*, 22(6):24–38, 2002.

[96] Xiaoping Yun, E. R. Bachmann, H. Moore, and J. Calusdian. Self-contained position tracking of human movement using small inertial/magnetic sensor modules. *Robotics and Automation, 2007 IEEE International Conference on*, pages 2526–2533, 2007.