In-situ Athlete Monitoring:
Data Collection, Interpretation & Feature Extraction

by

Andrew James Wixted
B. Eng (Hons) University of Southern Queensland

Griffith School of Engineering
Science, Environment, Engineering and Technology
Griffith University

Submitted in fulfilment of the requirements
of the degree of Doctor of Philosophy

May 2007
Abstract

The original core question of this thesis related to the real-time wireless collection of data from teams of athletes, in essence a technology - engineering issue. However, as time progressed, more fundamental questions evolved: What data is being collected, what does the data mean and how should it be interpreted? This question lies on the intersection of human movement studies, in the form of biomechanical & physiological sports science, and digital signal processing. The key biomechanical and physiological questions also feed back into the original real-time data collection issue. If data collected from athlete-mounted transducers\(^1\) is correctly interpreted, the salient features can be extracted and subsequently stored or transferred across a wireless network to a host system for analysis. Consequently this thesis addresses two intertwined issues, the real time in-situ data collection and the data processing necessary to extract some specific information useful to sports scientists.

In-situ athlete monitoring - data collection.

In the past, much athlete monitoring was performed under invasive conditions, either laboratory conditions or in specially prepared circumstances that were in-effect portable laboratories. Biomechanical analysis occurred on specially prepared rowing sculls or on running tracks with built-in instrumentation or tracks monitored by specialised high-speed cameras. Physiological monitoring required heart-rate monitoring, sampling of body fluids, monitoring of expired gasses and numerous other invasive techniques. With the advent of Micro-Electro Mechanical Systems (MEMS) such as gyroscopes, magnetometers, accelerometers etc., the opportunity existed for the in-situ monitoring of athletes both in training and in competition. While in-situ athlete monitoring can be simple data logging from a single transducer, the modern sports scientist needs more flexibility to gather the necessary information. Whether data is collected from multiple intra-athlete sensors or multiple inter-athlete sensors there must be a mechanism for synchronising that data so that it can be correctly recombined. This thesis addresses the synchronous data collection through the development of a real-time operating system, designed to operate as a node within a synchronised wireless sensor network. This thesis proposes wireless protocols, or

\(^1\) In this thesis the predominate transducer is the inertial sensor or 'accelerometer' although data from other transducers is also analysed.
topologies, necessary for the operation of low-powered synchronised wireless sensor networks. Finally, the issue of appropriate data sampling, necessary to maximise retained information and minimise bandwidth, is addressed in a discussion on data compression. This discussion analyses the raw data from the viewpoints of the data collection and transport process and the useful extractable content of the data. Each aspect of the system was filtered through the necessity of minimising the effect of the monitoring system on the system under observation. In the case of elite athletes this means making a system as unobtrusive as possible, which ultimately impacts the power available for data collection, synchronisation, processing and transport.

Data interpretation and feature extraction.
There appears to be enormous potential for data collection and analysis as it relates to elite athletes and elite level sport. In this thesis only one or two thin slices of sports science, as it relates to elite athletes, were investigated. These investigations addressed specific questions relating to the automation of physiological monitoring of rugby football players.

Accelerometers have been used as tools for the estimation of human energy expenditure (EE). The EE estimator was known as 'counts' and, while effective for estimating EE of humans walking, was ineffective for estimating EE during running activities. It was hypothesised that the ineffectiveness was due to some form of biomechanical running efficiency factor that upset the EE estimator, and that this biomechanical efficiency factor may be extractable from the collected data. To extract apparently useful information, a range of signal processing techniques were investigated and their usefulness, in terms of correctness, processing efficiency, accuracy and interpretability analysed. The 'running efficiency factor' was investigated through biomechanical analysis of systematically collected multi-athlete multi-speed treadmill data. The biomechanical analysis (documented in Ch.6) while inconclusive on the original running-efficiency question suggested the possibility of other applications, such as running technique coaching, for this form of data collection and analysis.

The biomechanical analysis, in conjunction with an understanding of the related kinematics ultimately lead to the identification of features in the athlete data that could
be used as EE estimators. Linear regression analysis identified that the athlete's leg-length appeared to have a correlation with several extracted parameters. One of these parameters was the athlete's natural step-frequency or cadence. By combining, in a linear equation, an athlete's step frequency with the athlete's anthropomorphic measures of leg-length and mass, an EE estimator that was more effective than the accelerometer 'counts' based estimators was generated. This estimator was immune to a range of errors that affect 'counts' based estimators.

Initially this EE estimator was extracted using the processing power of a desktop computer. It was necessary to move the processing to the lower power in-situ sensor platform. The underlying signal processing functions were replaced with potentially less accurate, low-power, signal processing functions. The effectiveness of these different techniques was evaluated.

While each chapter of this thesis is essentially stand-alone, most chapters describe some major component of the in-situ athlete monitoring system. Therefore, the in-situ athlete monitoring - data collection, interpretation and feature extraction is described by the low-powered real-time operating system (Ch.3) which provides the platform. The optimal sampling system is identified from the signal analysis of Ch.7 as is the data compression system used to log on-board data. Ch.4 describes the various signal processing techniques and identifies appropriate and inappropriate techniques for extracting information. Ch.5 analyses the extraction of the EE estimate and details the low-power implementation of the total rugby feature extraction. Ch.2 describes and proposes wireless network implementations necessary to bring together multiple individual in-situ monitoring systems, to generate a cohesive team-sport in-situ monitoring system.
STATEMENT OF ORIGINALITY

This work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Andrew Wixted
9 May 2007
Many PhD candidates feel that they live the life of Sisyphus. Odysseus, on his visit to Hades saw 'Sisyphus [a criminal], he was toiling painfully up a steep mountain's side, heaving a weighty stone before him, straining with hands and feet to push it to the summit. But every time he approached the top the stone slipped through his hands and thundered down the mountain's side until it reached the plain.'

The difference of course is that while a PhD may feel like eternal punishment, it doesn't actually go on forever. Meanwhile the underworld that the PhD candidate inhabits is populated by many people who make life bearable, possibly even enjoyable at times. To these people I extend my thanks.

To my principal & associate supervisors, Professor David Thiel & Dr. Daniel James, I thank you for your time, guidance and advice.

To the staff from the Australian Institute of Sport, Physiology department, particularly Professor Allan Hahn, Dr Chris Gore and Dr David Pyne, I also thank you for your encouragement, for suggestions on experiments and for arranging the collection of copious data.

I thank associates Dan Billing (Swinburne) and Grant Duthie (AIS) for sharing their athlete data, without which this thesis would not have come about. Also Philo Saunders (AIS) who conducted the physiological testing that is analysed in this thesis. At Griffith I had many associates, graduate and postgraduate students as well as staff, such as Neil Davey who assisted me from time to time and to them I extend my thanks.

Finally of course I thank my family, particularly my wife Michelle, who started me down the road of post-graduate studies and who has encouraged me to finish my thesis by reading aloud the job ads, every weekend. I know when she gets to the careers section of the newspaper because of the question "what is it that you do again?" My daughters will think of it not so much as finishing a thesis but as losing a taxi driver while my 8-year-old son, who does not remember life before "the thesis", will sadly discover that normal dads do not make them miss a morning's school just to watch trains in the goods yard. Thank you all for your patience.

---


3 Some people call this academia.
# TABLE OF CONTENTS

Abstract ............................................................................................................. i

Statement of Originality ................................................................................. iv

Acknowledgements .......................................................................................... v

1 **Introduction** ............................................................................................... 1-1

  1.1 Athlete Tracking System ............................................................................. 1-2
  
  1.2 Thesis outline and research direction ......................................................... 1-5

  1.3 Research Focus ........................................................................................... 1-7

    1.3.1 Wireless Sensor Networking for Monitoring Teams of Athletes ..... 1-8
    1.3.2 Low-Power, Real Time, Wireless Ready Operating System for Sensor Platform. 1-8
    1.3.3 Introduction to Accelerometer Derived Athlete Data ....................... 1-9
    1.3.4 Estimated Energy Expenditure .............................................................. 1-10
    1.3.5 Biomechanical Analysis ...................................................................... 1-11
    1.3.6 Compression of Athlete Data for Transmission and Storage ........... 1-12

  1.4 Research Outcomes ..................................................................................... 1-13

  1.5 References .................................................................................................. 1-16

2 **Wireless Networking** .................................................................................. 2-1

  2.1 Introduction .................................................................................................. 2-1
  
  2.2 Sensor Network + Wireless Network ......................................................... 2-3
  
  2.3 Network Topologies ................................................................................... 2-4
  
  2.4 Common Wireless Networks .................................................................... 2-5
  
  2.5 System Requirements and Constraining Regulations .............................. 2-8

    2.5.1 Radio Spectrum Management .............................................................. 2-8
    2.5.2 Health Impact ..................................................................................... 2-9
    2.5.3 Size, Weight & Power Constraints ...................................................... 2-9
    2.5.4 Data Rate Requirements .................................................................. 2-10
    2.5.5 Operating Environment & Required Path Length .............................. 2-11
    2.5.6 Antenna Size & Orientation ............................................................... 2-12
    2.5.7 Multi-link paths ................................................................................. 2-14
    2.5.8 Athlete Location Tracking ................................................................. 2-15
    2.5.9 Data Fusion Requirements ................................................................. 2-17
3.3.3 Scheduler Design ................................................................. 3-22
3.3.4 Interrupt Handling ............................................................ 3-26
3.3.5 Concurrency, Deadlocks and Semaphores ....................... 3-27
3.3.6 Dynamic Memory Allocation (malloc) ............................... 3-30
3.3.7 Low Power Operation ....................................................... 3-30
3.3.8 Communications State Machines ...................................... 3-36
3.3.9 Programmers Interface and Device Drivers ....................... 3-38
3.3.10 Loose Coupling Between Hardware and RTOS ............... 3-40
3.3.11 RTOS Clock System and Support for Synchronisation ....... 3-40
3.3.12 Support for Wireless Networking .................................... 3-50

3.4 RTOS Results ............................................................................ 3-51
3.4.1 Memory Requirements and Processing Load - Basic System ... 3-52
3.4.2 Memory Requirements and Processing Load - Complex System ... 3-53
3.4.3 MCU and Sub-system Power Requirements ....................... 3-54
3.4.4 Scheduler Testing ............................................................... 3-55
3.4.5 Clock Reception and Scheduler Synchronisation .................. 3-57

3.5 RTOS Summary ......................................................................... 3-59

3.6 References ................................................................................. 3-60

4 Introduction to Athlete Accelerometer Data ......................... 4-1
4.1 Introduction ............................................................................... 4-1
4.2 Operational environment and limitation of accelerometers ....... 4-2
  4.2.2 Accelerometer Factors & Limitations ................................. 4-4
  4.2.3 Factors & Limitations: Summary ....................................... 4-16
4.3 Signal processing and visualisation of Athlete Data ............... 4-17
  4.3.1 Test Data ........................................................................... 4-18
  4.3.2 Filtering ............................................................................ 4-21
  4.3.3 Orientation of Sensors - Extracting Angle ......................... 4-26
  4.3.4 Power Spectral Density (FFT) ............................................ 4-33
  4.3.5 Power Temporal Density .................................................. 4-35
  4.3.6 Autocorrelation ............................................................... 4-38
  4.3.7 Gait Cycle Normalisation of Acceleration Signal ................ 4-42
  4.3.8 Using Integration to Extract Velocity and Displacement .... 4-46
  4.3.9 Combining External Reference Data with Accelerometry .... 4-47
  4.3.10 Miscellaneous Processing ............................................... 4-49
5  Estimated Athlete Energy Expenditure ............................................. 5-1

5.1 Introduction .................................................................................. 5-1

5.2 Background .................................................................................. 5-1

5.2.1 Current Energy Expenditure Estimating Techniques ..................... 5-2

5.3 History: Energy Estimation Using Accelerometers ............................. 5-3

5.4 Limitations of Accelerometers as Energy Estimators ........................ 5-4

5.5 10x10 Treadmill Trial ...................................................................... 5-6

5.5.1 Method ...................................................................................... 5-6

5.5.2 Results: Accelerometer Counts vs Speed ..................................... 5-7

5.5.3 Results: Step Frequency vs Speed .............................................. 5-9

5.5.4 Results: Anthropometric Effects ............................................... 5-10

5.5.5 Discussion: Source of Accelerometer Count Variation ............... 5-10

5.5.6 Discussion: Triaxial vs Uniaxial Results ..................................... 5-12

5.5.7 Discussion: System Filtering ...................................................... 5-13

5.5.8 Summary: 10x10 Treadmill Trial ............................................... 5-14

5.6 Treadmill Trial for Energy vs Step Frequency .................................. 5-15

5.6.1 Method ...................................................................................... 5-15

5.6.2 Results ...................................................................................... 5-16

5.6.3 Discussion .................................................................................. 5-17

5.6.4 Summary ................................................................................... 5-19

5.7 MEMS based Per-Second Energy Expenditure Estimator ................ 5-19

5.7.1 Combined AC - SF Categorisation .............................................. 5-20

5.7.2 Results ...................................................................................... 5-23

5.7.3 Summary ................................................................................... 5-25

5.8 Chapter Summary ........................................................................... 5-25

5.9 References .................................................................................... 5-27

6  Biomechanical Analysis of Athlete Accelerometer Data ..................... 6-1

6.1 Introduction .................................................................................... 6-1

6.1.1 Biomechanics ............................................................................ 6-1

6.1.2 Biomechanics & Sports Science ................................................ 6-2
6.1.3 Kinematics .................................................................6-3
6.1.4 Accelerometer and Frame of Reference (FOR) Limitations ....6-4
6.2 Biomechanics Signal Categorisation ..................................6-7
6.2.1 Biomechanics Variables and Variable Quantification .......6-8
6.3 Biomechanical Analysis using CoM Accelerometers ...........6-11
6.3.1 In-stride acceleration, velocity and displacement relationship ....6-12
6.3.2 Consistency of Running Action ..................................6-15
6.3.3 Angle of Lean ..........................................................6-17
6.3.4 Treadmill Acceleration & Derived Information Analysis ....6-20
6.4 Overground Analysis (1500m Track Running) .................6-40
6.5 Discussion ..................................................................6-47
6.6 Conclusion ....................................................................6-51
6.7 References ....................................................................6-52

7 Compression of Athlete Data .................................................7-1
7.1 Introduction ...................................................................7-1
7.2 Athlete Data Information Content ...................................7-2
7.3 Data Compression Background .......................................7-7
7.3.1 Entropy Coding .......................................................7-7
7.3.2 Static & Adaptive Systems ........................................7-8
7.3.3 System order ...........................................................7-9
7.3.4 Dictionary Schemes ..................................................7-9
7.3.5 Lossy vs Lossless Compression .................................7-10
7.3.6 Transformation .......................................................7-11
7.3.7 Patents ....................................................................7-11
7.3.8 Other Systems ..........................................................7-11
7.3.9 Summary ...................................................................7-13
7.4 Proposed Athlete Compression .......................................7-13
7.4.2 Sampling System ......................................................7-14
7.4.3 Silence Compression ................................................7-15
7.4.4 Differencing ............................................................7-16
7.4.5 Companding .............................................................7-17
7.4.6 Minimum Redundancy Coding .................................7-20
7.4.7 Run Length Encoding ..............................................7-22
7.5 Accelerometry Data Analysis ..........................................7-24
<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5.2</td>
<td>Sampling System Analysis ......................................... 7-25</td>
</tr>
<tr>
<td>7.5.3</td>
<td>Silence Compression .................................................. 7-30</td>
</tr>
<tr>
<td>7.5.4</td>
<td>Differencing ............................................................ 7-31</td>
</tr>
<tr>
<td>7.5.5</td>
<td>Companding ............................................................... 7-35</td>
</tr>
<tr>
<td>7.5.6</td>
<td>Minimum Redundancy Coding ......................................... 7-39</td>
</tr>
<tr>
<td>7.5.7</td>
<td>Run Length Encoding .................................................. 7-42</td>
</tr>
<tr>
<td>7.6</td>
<td>Analysing Combined Compression Techniques ..................... 7-46</td>
</tr>
<tr>
<td>7.6.1</td>
<td>Combined Compression Techniques .................................. 7-46</td>
</tr>
<tr>
<td>7.6.2</td>
<td>Compression for wireless transmission on an unreliable link ... 7-50</td>
</tr>
<tr>
<td>7.6.3</td>
<td>Compression for storage or reliable transmission ............. 7-56</td>
</tr>
<tr>
<td>7.7</td>
<td>Implementation on sensor platform ................................ 7-57</td>
</tr>
<tr>
<td>7.8</td>
<td>Summary ........................................................................ 7-58</td>
</tr>
<tr>
<td>7.9</td>
<td>Athlete Data Compression Postscript ................................ 7-59</td>
</tr>
<tr>
<td>7.10</td>
<td>References ................................................................. 7-60</td>
</tr>
<tr>
<td>8</td>
<td>Concluding Comments .................................................. 8-1</td>
</tr>
<tr>
<td>9</td>
<td>Appendices .................................................................... 9-1</td>
</tr>
<tr>
<td>9.1</td>
<td>Appendix to Chapter 2 (Wireless Networking) ..................... 9-1</td>
</tr>
<tr>
<td>9.1.1</td>
<td>A Summary of 433.92 MHz License Conditions, ................... 9-1</td>
</tr>
<tr>
<td>9.1.2</td>
<td>Australian Class License Conditions ................................ 9-3</td>
</tr>
<tr>
<td>9.1.3</td>
<td>Class Licence and ISM Frequencies ................................ 9-4</td>
</tr>
<tr>
<td>9.2</td>
<td>Appendix for Chapter 4 Athlete Data Processing Toolbox ........ 9-6</td>
</tr>
<tr>
<td>9.3</td>
<td>Appendix for Chapter 5 (Estimated Energy Expenditure) .......... 9-8</td>
</tr>
<tr>
<td>9.3.1</td>
<td>10x10 Treadmill Trial Data ............................................ 9-8</td>
</tr>
<tr>
<td>9.3.2</td>
<td>SF vs $\dot{V}$O$_2$ based EE Treadmill Trial Data ............... 9-10</td>
</tr>
<tr>
<td>9.4</td>
<td>Appendix for Chapter 6 (Biomechanics) ............................ 9-12</td>
</tr>
<tr>
<td>9.5</td>
<td>Appendix for Chapter 7 (Data Compression) ....................... 9-15</td>
</tr>
<tr>
<td>10</td>
<td>Glossary and Acronyms ................................................ 10-1</td>
</tr>
<tr>
<td>10.1</td>
<td>Terms associated with Wireless Networking (Ch.2) ............. 10-1</td>
</tr>
<tr>
<td>10.2</td>
<td>Terms associated with Operating Systems (Ch.3) ............... 10-3</td>
</tr>
<tr>
<td>10.3</td>
<td>Terms associated with Signal Processing &amp; Biomechanics ....... 10-6</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Fig.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Overview of the Athlete Tracking Platform Concept</td>
<td>1-3</td>
</tr>
<tr>
<td>1.2</td>
<td>The Athlete Data Information System</td>
<td>1-6</td>
</tr>
<tr>
<td>2.1</td>
<td>Combining Sensor and Wireless Networks</td>
<td>2-3</td>
</tr>
<tr>
<td>2.2</td>
<td>Macro component combinations for wireless sensor platform</td>
<td>2-4</td>
</tr>
<tr>
<td>2.3</td>
<td>Basic network topologies</td>
<td>2-5</td>
</tr>
<tr>
<td>2.4</td>
<td>Radiation patterns of transmitters as a function of frequency</td>
<td>2-12</td>
</tr>
<tr>
<td>2.5</td>
<td>Topology of location tracking and data collection network</td>
<td>2-16</td>
</tr>
<tr>
<td>2.6</td>
<td>Location tracking using statistical estimation of position</td>
<td>2-16</td>
</tr>
<tr>
<td>2.7</td>
<td>Slave synchronisation using controllable oscillators or counters</td>
<td>2-24</td>
</tr>
<tr>
<td>2.8</td>
<td>Slave synchronisation using programmable counter/timers</td>
<td>2-25</td>
</tr>
<tr>
<td>2.9</td>
<td>Effect of Multi-link Path on Network Synchronisation</td>
<td>2-26</td>
</tr>
<tr>
<td>2.10</td>
<td>Sensor network FEC and MAC layer encapsulation</td>
<td>2-33</td>
</tr>
<tr>
<td>2.11</td>
<td>Sensor network data recovery</td>
<td>2-34</td>
</tr>
<tr>
<td>2.12</td>
<td>RF to MCU Interface Options</td>
<td>2-38</td>
</tr>
<tr>
<td>2.13</td>
<td>WiFi stations in Distributed Coordination Function using RBS</td>
<td>2-43</td>
</tr>
<tr>
<td>2.14</td>
<td>Bluetooth Piconets: single slave, a multi-slave and a scatternet</td>
<td>2-45</td>
</tr>
<tr>
<td>2.15</td>
<td>Two Bluetooth Topologies</td>
<td>2-47</td>
</tr>
<tr>
<td>2.16</td>
<td>Example data from receiver capture files for 150 and 200m</td>
<td>2-53</td>
</tr>
<tr>
<td>2.17</td>
<td>Collapsible Helical Antenna made with PVC pipe</td>
<td>2-55</td>
</tr>
<tr>
<td>2.18</td>
<td>Collapsible Yagi-Uda Antenna made with PVC pipe</td>
<td>2-55</td>
</tr>
<tr>
<td>2.19</td>
<td>Clock Drift: Slave clock is 3.75ms/minute fast</td>
<td>2-57</td>
</tr>
<tr>
<td>2.20</td>
<td>Clock Drift Roles reversed: Slave is 3.75ms/minute slow</td>
<td>2-57</td>
</tr>
<tr>
<td>2.21</td>
<td>Clock Drift CPU running 10ms primary interval</td>
<td>2-57</td>
</tr>
<tr>
<td>2.22</td>
<td>Reversed: Clock Drift CPU running 10ms primary interval</td>
<td>2-59</td>
</tr>
<tr>
<td>2.23</td>
<td>Packet construction for triplicated clock message</td>
<td>2-62</td>
</tr>
<tr>
<td>2.24</td>
<td>Message Loss Rate for FEC and ASCII Data</td>
<td>2-65</td>
</tr>
<tr>
<td>2.25</td>
<td>Triples Loss Rate for FEC and ASCII Data</td>
<td>2-66</td>
</tr>
<tr>
<td>2.26</td>
<td>Consecutive Message Loss Count for FEC and ASCII Data</td>
<td>2-66</td>
</tr>
<tr>
<td>2.27</td>
<td>Accepted Incorrect Message Rate for FEC and ASCII Data</td>
<td>2-66</td>
</tr>
<tr>
<td>3.1</td>
<td>Task States for Background and Real-Time Scheduler Tasks</td>
<td>3-19</td>
</tr>
</tbody>
</table>
Fig. 4.13  Magnitude of output of 1Hz low-pass filter. ........................................4-20
Fig. 4.14  Power Spectral Density for inter-tackle run phase. .........................4-20
Fig. 4.15  Vertical acceleration signal - with overload. ..................................4-22
Fig. 4.16  Single pole hardware filter or digital Infinite Impulse Filter (IIR) ....4-23
Fig. 4.17  The Hamming Windowed FIR Filter.................................................4-24
Fig. 4.18  The Rectangular Windowed FIR Filter. .........................................4-24
Fig. 4.19  First Order Lag Processing..............................................................4-24
Fig. 4.20  Comparison of Off-Normal Angles from various filters. .................4-25
Fig. 4.21  Alignment of accelerometers on and extracting orientation ............4-27
Fig. 4.22  Raw and rotated 2-dimensional views running acceleration ..........4-29
Fig. 4.23  Resolving angles with the cosine rule. ...........................................4-29
Fig. 4.24  Instantaneous orientation angle .....................................................4-30
Fig. 4.25  Comparison of off-normal angle with vertical acceleration ..........4-31
Fig. 4.26  2-Dimensional views of the 3-tackle sequence. ...............................4-32
Fig. 4.27  3-Dimensional view of the 3-tackle sequence.................................4-32
Fig. 4.28  Comparison of multispeed FFT analysis for different samplelengths...4-33
Fig. 4.29  Power Spectral Density, cross channel and inter athlete comparison ...4-34
Fig. 4.30  Applied and Detected Acceleration Vectors ....................................4-36
Fig. 4.31  Comparison of Power Temporal Density from three methods .........4-37
Fig. 4.32  Comparison of Power Temporal Density (1.0s averaging) ...............4-38
Fig. 4.33  Autocorrelation Output.....................................................................4-39
Fig. 4.34  Comparison of autocorrelation output for different running speeds. ....4-40
Fig. 4.35  Acceleration vs. autocorrelation for low speed walking ..................4-41
Fig. 4.36  Power Spectral Density (FFT) of low speed (3kmh⁻¹) walking signal ..4-41
Fig. 4.37  Creating a representative gait cycle signal ......................................4-44
Fig. 4.38  Processing ML channel data to distinguish left and right steps .........4-44
Fig. 4.39  Comparison of AP acceleration from different running speeds. .......4-45
Fig. 4.40  Gait Cycle normalised AP acceleration, different running speeds ......4-45
Fig. 4.41  Running Data: 3D Acceleration, Velocity and Displacement ............4-47
Fig. 4.42  Times series acceleration with foot contact reference points ..........4-48
Fig. 4.43  Foot contact reference marking on Velocity graphs .........................4-48
Fig. 4.44  Notional delays in detecting the Zero-Crossing ..............................4-50
Fig. 4.45  Step Frequency Estimation: Technique Comparison .......................4-51
Fig. 4.46  Comparison of signal power in orientation and dynamic signal .........4-52
Fig. 4.47 Multi-channel Power Spectral Density from 1500 metre race data......4-53
Fig. 4.48 3D Views of the orientation vector..........................................................4-54
Fig. 4.49 Desktop Processing for Rugby-Football Athlete Monitoring..............4-58
Fig. 4.50 Embedded Processing for Rugby-Football Athlete Monitoring...........4-58
Fig. 4.51 Two methods of generating Power Temporal Density.....................4-61
Fig. 4.52 Power Temporal Density: 150Hz floating-point vs 25Hz integer-only 4-61
Fig. 4.53 Close up of Power Temporal Density for 150Hz vs 25Hz.................4-61
Fig. 4.54 Angles: 150Hz FIR based orientation signal vs 25Hz simplified........4-62

Fig. 5.1 Oscillatory vertical movement during walking.................................5-5
Fig. 5.2 Uniaxial & Triaxial Accelerometer Counts from Treadmill Trial ....5-8
Fig. 5.3 Individual triaxial accelerometer counts for running 11-21kmh\(^{-1}\) ....5-9
Fig. 5.4 Step frequency results for 10x10 ..............................................................5-9
Fig. 5.5 Miscellaneous regression analysis ..........................................................5-11
Fig. 5.6 Triaxial accelerometer counts normalised by the 9kmh\(^{-1}\) SF ............5-12
Fig. 5.7 Gait-cycle normalised, multi-speed acceleration, selected athletes ......5-13
Fig. 5.8 Step Frequency vs. EE estimated from \(\dot{V}O_2\)......................................5-17
Fig. 5.9 Regression Analysis for Step Frequency vs Energy Expenditure after adjustments for step frequency and mass......................................5-19
Fig. 5.10 Comparison of vertical acceleration for walking..............................5-20
Fig. 5.11 Generating combined AC-SF categorisation.......................................5-21
Fig. 5.12 Activity intensity histogram based on rescaled AC output..............5-24
Fig. 5.13 Activity intensity histogram based on combined AC-SF output........5-24

Fig. 6.1 Acceleration vs Frequency of Oscillation ........................................6-11
Fig. 6.2 Recorded vertical acceleration vs derived velocity and displacement.6-14
Fig. 6.3 Reconstituted acceleration data vs original acceleration data .........6-15
Fig. 6.4 Averaging of sample strides. .................................................................6-16
Fig. 6.5 Accelerometer alignment with athlete and gravity...........................6-17
Fig. 6.6 Angle Y, Rotation about ML axis. Angle Z, Rotation about AP axis...6-17
Fig. 6.7 Change in off-normal angles with change in running speed............6-18
Fig. 6.8 Correlation coefficients (as R\(^2\)) for Y & Z angle..........................6-18
Fig. 6.9 Calibrated acceleration data (Subject A-5).......................................6-21
Fig. 6.10 Low pass filter (1.0Hz FIR Hamming Window) output..................6-21
Fig. 7.6  Linear and Companding Analog to Digital Converters (ADC)..............7-18
Fig. 7.7  Companding algorithm code implementation fragment......................7-19
Fig. 7.8  Matlab code for predictive differencing in the compander...............7-20
Fig. 7.9  Code used to count the length of a variable length code..................7-21
Fig. 7.10 Run Length Encoded Data Bytes..........................................................7-23
Fig. 7.11 Modified Huffman comparison to standard Huffman.............................7-23
Fig. 7.12 Entropy for sample rates with and without silence compression.............7-24
Fig. 7.13 Time and Frequency Domain comparisons (21kmh⁻¹ running)..............7-26
Fig. 7.14 Time and Frequency Domain comparisons (3 kmh⁻¹ walking)..............7-26
Fig. 7.15 Comparison of 21kmh⁻¹ running data sampled at 150Hz and at 25Hz...7-27
Fig. 7.16 21kmh⁻¹ accelerometer data sampled at 150Hz/10bit vs 75Hz/7bit........7-28
Fig. 7.17 3kmh⁻¹ accelerometer data sampled at 150Hz/10bit vs 75Hz/7bit.........7-28
Fig. 7.18 Triaxial Accelerometer Counts (AC) after halving the sampling rate....7-29
Fig. 7.19 Impact of silence compression on total number of zero-differences......7-30
Fig. 7.20 Histogram of differences for 21kmh⁻¹ running data (150Hz 10bit).......7-32
Fig. 7.21 Histogram of differences for 21kmh⁻¹ running data (150Hz 8-bit) .......7-32
Fig. 7.22 Histogram of differences for 3kmh⁻¹ walking data (10bit & 8-bit).......7-32
Fig. 7.23 3D Histogram of differences for speeds 3-21kmh⁻¹ (150Hz 8bit).........7-33
Fig. 7.24 3D Histogram of differences for speeds 3-21kmh⁻¹ (75Hz 8bit)..........7-34
Fig. 7.25 Distribution of number of symbols (150Hz, 8-bits per sample)............7-35
Fig. 7.26 Distribution of number of symbols (75 Hz, 8-bits per sample)............7-36
Fig. 7.27 Reconstructed companded output compared to source data.................7-37
Fig. 7.28 Aggressive companding of 150Hz 8-bit data ........................................7-37
Fig. 7.29 Aggressive companding of 75Hz 8-bit data ..........................................7-38
Fig. 7.30 Comparison of code bits for minimum redundancy coding schemes.....7-39
Fig. 7.31 Compression ratios for 150Hz 8-bit data for various activity mixes .....7-40
Fig. 7.32 Compression ratios for 75 Hz 8-bit data for various activity mixes .......7-41
Fig. 7.33 Percentage of zero-differences by run-length and speed....................7-42
Fig. 7.34 Percentage of zero-differences in runs of three or more for silence compression ON and OFF .................................................................................7-45
Fig. 7.35 Histogram of activity intensity from football athlete.........................7-45
Fig. 7.36 Impact of bit errors on the decoding of variable length codes..............7-51
Fig. 7.37 Input signal compared to output signals after single bit error..............7-52
Fig. 7.38 Comparison of 150 Hz 8-bit decompressed data and source data........7-53
Fig. 7.39  Instantaneous compression ratio using blocks of 8 samples ...............7-53
Fig. 7.40  Instantaneous compression ratio using blocks of 16 samples ...............7-53
Fig. 7.41  Interleaving of channel data including absolute reference values and variable length codes. ........................................................................................................................................7-54
**LIST OF TABLES**

Table 2.1 Specification comparison for different wireless devices / technologies. ...2-7
Table 2.2 Required Network Data Rates for Individual and Teams of Athletes......2-10
Table 2.3 Device Transmitter Power, Receiver Sensitivity and EIRP ...................2-12
Table 2.4 Free Space Loss (FSL) calculated for Frequency and Distance.............2-13
Table 2.5 Clock Precision for Crystal Oscillator Types........................................2-21
Table 2.6 Bluetooth SCO Protocol Data Units..........................................................2-35
Table 2.7 Bluetooth Protocol Data Units (PDU) for ACO Links..........................2-35
Table 2.8 Preamble timing in 802.11b/g Physical Layer for FHSS and DSSS........2-42
Table 2.9 nRF903 RF Channel Frequencies and Frequency Bands......................2-48
Table 2.10 RF field strength during 433MHz data transmission tests. ....................2-52
Table 2.11 Error probability vs simulation results for ASCII data. ......................2-64
Table 2.12 1-Bit Error probability for start & stop bits vs FEC simulation results..2-65

Table 3.1 System Events and Deadline Interval Duration.................................3-11
Table 3.2 Clock Cycles for Deadline and Clock Frequency Combinations..............3-11
Table 3.3 Approximate Cycle Counts for Common Processes. ............................3-12
Table 3.4 Current draw for accelerometer and ADC subsystem...........................3-33
Table 3.5 Crystal Frequency, Timer Bits, Timer Error and Timing Precision.........3-43
Table 3.6 Clock Synchronisation Sophistication for Embedded RTOS..................3-45
Table 3.7 Clock Correction Application Points.......................................................3-45
Table 3.8 Basic System Memory Requirements ....................................................3-52
Table 3.9 Compression & Encoding Algorithm Code Sizes ................................3-53
Table 3.10 MCU current for different frequencies and clock-tick intervals..........3-54
Table 3.11 MCU current for ALU and subsystems at different frequencies.........3-54
Table 3.12 Debug Output Key...........................................................................3-55

Table 4.1 Signal Processing Techniques and Application ..................................4-55
Table 4.2 Interdependencies of Signal Processing Steps ....................................4-56
Table 4.3 Processing Cost to Generate Outputs (Hitachi H8 Processor) ..........4-60

Table 5.1 AIS Manual Activity Classification .....................................................5-3
Table 5.2 Correlation coefficients for kinematic parameters during walking........5-5
Table 5.3 Correlation Coefficients for EE, SF & ACs vs Treadmill Speed ..........5-16
Table 5.4 Correlation coefficients for SF and AC vs EE estimated from $\dot{V}O_2$ .... 5-17
Table 5.5 Combined AC - SF Decision Table.........................................................5-21
Table 5.6 Correlation coefficient for reduced samplerate ACs vs walking speed ...5-23

Table 6.1 Multi-subject Acceleration Comparison..............................................6-32
Table 6.2 Comparison of 1500m results.............................................................6-45

Table 7.1 Data rates based on sample rate and bits-per-sample.......................7-15
Table 7.2 Example Implementation of Huffman Code Table............................7-21
Table 7.3 Percent compression (bits per sample) for 150 Hz 8-bit data using static
encoding models ..................................................................................................7-40
Table 7.4 Percent compression (bits per sample) for 75Hz 8-bit data using static
encoding models ..................................................................................................7-41
Table 7.5 Effect on compression ratio of adding RLE...........................................7-43
Table 7.6 Comparison of percent compression of zero-differences for $3kmh^{-1}$ sample
data using combined MRC and RLE ($9kmh^{-1}$ data bracketed)......................7-44
Table 7.7 Data rates based on sample rate and bits-per-sample........................7-47
Table 7.8 Compression of silenced / differenced / companded data using minimum
redundancy encoding .............................................................................................7-47
Table 7.9 Compression ratios for combined 150Hz test data using combined
compression techniques .......................................................................................7-48
Table 7.10 Compression ratio statistics for combined 150Hz 8-bit test data........7-48
Table 7.11 Compression of data from athlete engaged in football training ..........7-48
Table 7.12 Compression techniques compared by sport and data channel ..........7-49
Table 7.13 Differences mapped to variable length codes (for use in Fig. 7.36) .....7-51

Appendix Tables
Table 9.1 European 433MHz License Conditions.............................................9-1
Table 9.2 European, country specific 433MHz license conditions....................9-1
Table 9.3 FCC Regulations for Intentional Radiators .......................................9-2
Table 9.4 Existence of 433MHz Band in Africa, Asia and Americas (not USA) ..9-2
Table 9.5 Australian Class License ....................................................................9-4
Table 9.6 ISM Bands designated in the Australian Spectrum Plan....................9-5

xx
Table 9.7 Athlete Anthropometric Measures ................................................................. 9-8
Table 9.8 Step Frequency Results .................................................................................. 9-8
Table 9.9 Triaxial Accelerometer Counts ...................................................................... 9-9
Table 9.10 Uniaxial Accelerometer Counts .................................................................. 9-9
Table 9.11 Athlete Anthropometric Measures ............................................................... 9-10
Table 9.12 Step Frequency Results ................................................................................ 9-10
Table 9.13 Triaxial Accelerometer Counts .................................................................... 9-11
Table 9.14 Volume of Oxygen used (litres) ................................................................... 9-11
Table 9.15 Energy Expenditure (EE) estimated from $\dot{V}O_2$ .................................... 9-11
Table 9.16 Trunk lean angles, angle correlation coefficient and change in Angle .......... 9-12
Table 9.17 Non-rotated data: Multi-speed Peak to Peak acceleration and correlation coefficients for each athlete ................................................................. 9-13
Table 9.18 Rotated data: Multi-speed Peak to Peak acceleration and correlation coefficients for each athlete ......................................................................................... 9-14
1 Introduction

At a personal level, sport is an area of life where a person can strive to beat both themselves and their competitors. If a competitor of a comparable level is not available, an athlete can still obtain a high sense of personal achievement by competing against (and beating) a tape measure or stopwatch. Sport at the most elite national and international levels provides high levels of prestige, entertainment and commercial return for a variety of interests. This prompts considerable investments in legal and illegal methods of developing athletic prowess. Illegal athlete development programs are generally perceived to centre round the development and use of performance enhancing drugs and associated masking agents. In contrast, legal development programs focus on understanding the relationships between the physiological, biomechanical and psychological processes of humans engaged in sport. This thesis looks at developing techniques for monitoring, interpreting, collecting and conveying biomechanical and physiological data of interest from an athlete engaged in training and competitive activities.

The Australian Institute of Sport (AIS), in common with many national sporting development organisations, is engaged in the development of athletes using knowledge obtained through scientific research. This thesis describes research that formed part of a group of projects generically termed 'Athlete Tracking', where tracking refers to the process of monitoring the athlete's development, training and performance. The Athlete Tracking projects were jointly funded by the AIS, the Cooperative Research Centre (CRC) for Micro technology and other partners, including Griffith University. The formal overall name of the related projects within the CRC was "Interface Technologies for Athlete Monitoring". The scope of the projects includes monitoring athletes involved in team sports, swimming, boxing, rowing and athletics. The parameters of interest are also diverse, including but not limited to, step rates / stroke rates (running, team sports, swimming, rowing), energy expenditure (team sports, swimming), velocity, position, biomechanics (running, rowing), lactate levels (most sports), heart rate etcetera. At present, all of these factors can be measured but only under laboratory conditions invasive techniques. The development of miniature, portable technologies that enable the collection of processed sport-related information is intended to streamline the athlete's development by providing timely feedback. This
data is used by physiologists in assessing effectiveness of training regimes in the
development of fitness and strength, by biomechanists in the development of
technique, by coaches developing and monitoring training plans and by athletes
managing their preparation and competition performance.

This thesis deals with a specific subset of the above problem set. (1) A generic
platform for the collection of data in any sport, (2) the constraints of a wireless system
used in the real-time collection of data particularly from teams of athletes, (3)
identifying specific athlete activities in rugby union, (4) generating energy estimates
from athletic activity and (5) reducing activity data to summary form for storage or for
transmission on a wireless network. Due to the nature of sport this functionality should
be encompassed in a package that is acceptable to all athlete communities, therefore
requiring low weight and a small package. Demands on the operating system and
wireless network are partially determined by the fundamental frequency of physical
activity, which while relatively low, ranging from about 0.5Hz to 5Hz, can have some
strict timing requirements. Where data fusion is required, as in the combining of
wirelessly collected data from multiple sensors, the synchronisation becomes
important with the requirements determined on a sport-by-sport basis.

The constant advancement of technology has meant that some technologies that could
not feasibly be used for this project only a few years ago, have matured and are now
competitive for price, size and power. This offers opportunities to simplify a number
of the aspects of the physical implementation of athlete monitoring devices.

1.1 Athlete Tracking System

The 'Athlete Tracking' system is summarised in Fig. 1.1. Some aspect of athletic
activity is monitored by an appropriate sensor and data captured, processed,
transmitted (wirelessly in this example) and eventually, received, decoded and
interpreted by the appropriate sports scientist such as physiologists and biomechanists
or by coaches and athletes. Use of directional antenna or other spatial detection
techniques enables the location of athletes in the field.
The athlete tracking system concept developed from a review of the current methods of analysing the activities of elite athletes [1]. This review generated requirements [2], which, whilst attempting to set some definite, quantifiable requirements in terms of athletic parameters (speed, location, athlete energy expenditure, activity timing) and device parameters (size, weight, sensor bandwidth, on-time, logging capacity) were essentially long-term requirements. In keeping the long-term objectives, the plan was to develop a system as small as possible, with as much sensor and logging bandwidth as possible, battery power to last a training session, and do it as soon as possible within the financial and technological constraints. The current athlete tracking sensor platform was therefore a point-in-time development outcome that is currently being used to collect and process athlete data in a range of sports. The sensor platform
performance, analysis of activity data and feedback from a range of users are all inputs to the next iteration of the athlete tracking project developments.

While some aspects of the development of the athlete tracking system have been substantially completed (Platform [3], Operating System [4], Swimming [5], Rowing [6], In-sole sensors [7], Field Sports [8]), others are ongoing (Boxing [9], Lactate, Location etc). The initial prototypes of the sensor platform, sampling an accelerometer at 50Hz and transmitting the data over a single unidirectional point-to-point wireless link, were developed in 2001 [10]. Since then, some technologies have or are leapfrogging some aspects of the above system, Global Positioning System (GPS) chips running the appropriate algorithms operate sufficiently fast enough that they can replace the accelerometer components [11]. The prices of these chips are approaching those of the Micro Electro Mechanical Systems (MEMS) based accelerometers used in the athlete tracking system. Similarly, the Bluetooth wireless technology devices [12], which incorporate transceivers, wireless networking protocols, sensor inputs and processing capacity are approaching a price point of US$5 [13].

By combining the research outcomes from this cycle of activity with the technology now becoming available, the next development will see devices that, while not meeting all the requirements, will certainly be smaller, lighter, faster, more accurate than the current system.
1.2 Thesis outline and research direction

This thesis is focused on a small subset of the athlete tracking system outlined above. Initially, effort was expended in developing a generic networked wireless sensor platform, using commercial devices, for monitoring teams of athletes in real time. As the athlete-tracking project progressed it was realised that (a) the analysis and interpretation of data is of more importance than the wireless collection of data, (b) for individual and groups of athletes in training, collecting data wirelessly in real-time is actually more cumbersome than logging the data on board for later downloading, (c) teams are groups of individuals and understanding individual activity is a precursor to understanding team activity. From the viewpoint of the sponsoring group, the AIS department of physiology, physiological functioning and in particular athlete energy expenditure is of primary importance. The research focus has therefore been;

- develop a wireless ready real-time sensor platform,
- collect athlete data using accelerometers,
- undertake generic analysis of athlete data,
- undertake specific analysis with respect to,
  - energy expenditure,
  - biomechanics,
  - anthropometric effects,
- implement compression for data logging and data transfer.

Chapters in this thesis

Due to the diversity and multi-disciplinary nature of the content, the chapters of this thesis are generally independent, with each chapter incorporating the current state of research in the relevant field.

Chapter (1) Introduction.

This chapter presents a general introduction to the thesis, the athlete-tracking project and the overall research focus.
**Data Collection Chapters:**

**Chapter (2) Wireless Sensor Networking for Monitoring Teams of Athletes**  
This chapter presents an analysis of the requirements of a wireless sensor network for athlete monitoring and then analyses existing and proposed network synchronisation techniques necessary to meet those requirements.

**Chapter (3) Low-Power, Real Time, Wireless Ready Operating System for Sensor Platform.**  
An athlete data collection system that can operate in a standalone or networked scenario requires the implementation of a generic, hardware independent, synchronisable, real time operating system capable of low power operation, data collection, data compression, feature extraction and operation in a wireless network. This chapter analyses the minimal Operating System functionality necessary to meet the requirements and then proposes a suitable Operating system.

**Data Analysis and Feature Extraction Chapters.**  
Chapters 4-7, together these chapters look at various ways of identifying and/or extracting information content from the athlete data. The ability to extract information at the source minimises the volume of data to be transferred and therefore minimises the required storage or transmission circuit bandwidth. As suggested by Fig. 1.2, the difficulty lies in identifying what is relevant information.

*Fig. 1.2 The Athlete Data Information System. Although the accelerometers can detect and convert the body movement into digital values for transmission and storage, the values are useless unless they are in an understandable form.*

**Chapter (4) Introduction to Athlete Data.**  
The presents a general investigation and analysis of accelerometer collected athlete data using test and competition data. It identifies basic features of accelerometer sensed athlete movement and the applicable limitations of accelerometer data in the
context of monitoring elite athletes. It analyses the athlete data in a number of ways using a range of signal processing techniques.

**Chapter (5) Athlete Estimated Energy Expenditure**
Numerous papers have been published on the use of accelerometers for estimating human energy expenditure. The techniques described are sometimes suitable for lower speed ambulation but due to the running mechanics and the accelerometer limitations, they are totally inadequate for estimation of energy expenditure of running athletes. This chapter analyses systematically collected athlete data and identifies a more robust accelerometer based energy estimator.

**Chapter (6) Biomechanical Analysis**
Analysis of accelerometer-collected athlete data suggested that the data might contain information of interest in the field of biomechanics. The analysis of this chapter suggested that the data contained sufficient information to allow some form of categorisation of biomechanical activity based on trunk kinematics. It also identified areas of future study that may allow the extraction of a number of useful biomechanical parameters.

**Chapter (7) Compression of Athlete Data for Transmission and Storage**
Extracting an energy estimator (Chapter 5) or sensor orientation (Chapter 4) at the source is a form of data compression. This chapter analyses the efficiency of applying traditional compression techniques to the raw athlete data in a way that maximises the preserved information content and at the same time minimises the processor load, the memory requirements and the transmission bandwidth.

**Chapter (8) Summary**
This section presents results and key outcomes from the research.

**1.3 Research Focus**
The core content and research focus of each of the specific content areas is summarised below.
1.3.1 Wireless Sensor Networking for Monitoring Teams of Athletes

There are a considerable number of wireless protocols and devices both on the market and in development. Wireless Sensor Networking is an area of intense interest with wireless sensing being developed for large structural monitoring of static and mobile structures (for example: bridges and boats), small machines (such as engines) and large area coverage (such as agriculture and planetary surface monitoring). Commercial and domestic wireless sensor networking is also being developed as a method of reducing or eliminating certain types of building wiring such as air conditioner sensors and light switches. Structural sensors are static. They operate over short ranges with low data rates and because of their static nature, can develop routing tables and perform low-cost synchronisation. Wireless monitoring of machines can involve un-powered sensors that can be wirelessly interrogated - but not necessarily using Radio Frequency (RF) signals. Wireless agricultural monitors are very low data rate and not as power sensitive as an athlete system.

Low-power wireless monitoring of elite athletes operating over a medium range (greater than 10 metres, less than a kilometre) falls into a gap in the above scenarios. Wireless sensor devices must be very small and unobtrusive, limiting both the duration of operation and the peak current available. The wireless range is comparatively large. The wireless nodes are dynamic in that they constantly move in relation to one another and static or slowly changing data routing is not feasible. Athlete monitoring has potentially high data rates particularly when monitoring a team of athletes. Finally, the necessity of synchronising data from multiple diverse sensors requires a mechanism for distributing synchronisation messages that do require complex routing or high power use on the mobile device to deliver.

The research focus targets the implementation of low power delivery of synchronised data from mobile sensor devices using different technologies currently available.

1.3.2 Low-Power, Real Time, Wireless Ready Operating System for Sensor Platform.

Embedded systems today are ubiquitous, incorporated into telephones, household appliances, vehicle engine management systems and braking systems, household and personal radios, stereos etc. Small computer-network appliances incorporate very
powerful embedded systems utilising more computer resources than desktop computers of 10 years earlier. In most of these systems, power consumption and size is not a serious issue. System capacity is a function of market forces, a compromise between system requirements, technological advancement and appropriate price point.

For systems worn by elite athletes, the driving factor is size. Technological advancement must be utilised to drive down the system size. To keep size down, the hardware requirements of the operating system and on-board functionality must be kept small and the processing power required to perform the required functionality must be minimised.

Where data fusion is required across many sensor nodes, and wireless networking is necessary to carry data, the operating system of a sensor node must be capable of managing the networking protocols and keeping synchronisation with the network.

The research focus for this chapter has been the development of a loosely coupled, synchronisable, low-power, wireless ready, real-time operating system with small processing and resource footprints.

1.3.3 Introduction to Accelerometer Derived Athlete Data

The focus of this chapter is to outline the framework for the use of accelerometer derived data in subsequent chapters.

Using triaxial accelerometers, data was collected from a range of athletes in testing, training and competition. Tests included systematic testing of one particular biomechanical action using a range of athletes - to testing one athlete over a range of different activities. From the accelerometer data collected, there was an expectation that certain types of information could be derived. The extracting of useful information is subject to limitations due to the environment and the choice and implementation of the signal processing techniques.

While accelerometers are used to extract certain types of information in particular circumstances, the ability to extract some specific kinds of information from
accelerometers monitoring elite athletes, is subject to numerous technical and
environmental limitations. To understand the types of information that are extractable,
it is necessary to understand the source and impact of these limitations. This chapter
enumerates many of the key limitations.

Taking into account the limitations identified above, there is potential to extract useful
information using appropriate signal processing techniques. Because of the limitations
and the complexity of the signals involved, the appropriate signal processing and
parameter identification can involve trial and error. This chapter explores the
implementation of different signal processing techniques in an attempt to optimise the
information extracted.

Finally, some of the 'optimal' techniques are implemented in a degraded and limited
form using the processing limitations of an embedded system. The output from the
embedded system processing is qualitatively and quantitatively assessed against the
optimal processing.

1.3.4 Estimated Energy Expenditure

The research focus in this chapter is straightforward. It has previously been identified
in research, that for some forms of activity, the processed output from accelerometers
(known as 'counts') has a strong correlation with an individual's energy expenditure.
Therefore accelerometers have been used in many studies in the estimation of energy
expenditure in daily living. Subsequent research has attempted to extend this
accelerometer - energy expenditure correlation into more physically intense activities,
particularly running.

The initial hypothesis was that the individual athlete had a 'running-efficiency' factor
that may be discernable in the accelerometer signal, and that this factor caused the lack
of correlation in research results. Through researching this hypothesis it was
discovered that the variation in accelerometer output was a function of two obvious
kinematic parameters, and that these parameters were strongly influenced, primarily by
one anthropomorphic factor, namely leg-length.
It had long been known that, for physically fit individuals, there is a very strong correlation between running speed, the athlete's mass and their energy expenditure. From the results of the foregoing research, it was proposed that accelerometer derived step-frequency (instead of 'counts'), in conjunction with the athlete's anthropomorphic measures of mass and leg-length, would give a strong correlation with energy expenditure. Experiments were conducted to test this hypothesis. The results from this experiment confirmed the results from the previous experiment.

Where an athlete's mass and leg-length is known, extracting step-frequency from accelerometer data is sufficient to generate a useful energy expenditure estimate for running.

**1.3.5 Biomechanical Analysis**

The research focus of this chapter is the comparison of gait-cycle normalised data from a group of athletes to identify methods of categorising the biomechanical activities during running. Gait-cycle normalisation is a technique used by some researchers for biomechanical analysis, often for use with the elderly or infirm, sometimes for use with healthy subjects.

Researchers into biomechanics of running utilise a variety of investigative techniques most of which are non-portable or laboratory based. Typical of these techniques is the use of high-speed cameras and sophisticated computer processing software. Set-up, post processing and analysis of the data is time consuming and costly, limiting the usefulness and availability of the technique. Accelerometer based sensing is portable, cheap and has the potential for virtually instant feedback. A runner working on developing technique can acquire feedback within minutes of a training session.

Accelerometry has been combined with in-sole sensors to develop a replacement of the expensive and limited availability of the force-plate. Yet the accelerometer data contains useful and extractable information in its own right.

The research of this chapter, despite the completely different emphasis, came out of the research of the previous chapter where it was identified that particular kinematic and anthropometric parameters controlled the signal power in the accelerometer signal.
This identified that biomechanical acceleration patterns were linked to anthropomorphic factors and that these patterns were systematically generated across a range of running speeds. By combining gait-cycle normalised patterns from a single athlete running at different speeds, speed dependent and independent behaviours were identifiable. By combining multi-speed gait-cycle normalised acceleration patterns from multiple athletes, the variations in speed dependent and independent behaviours across the athletes indicated that categorisation of athlete technique, based on multi-speed gait-cycle normalised data, was possible. Combining this categorisation with the analysis from previous chapters, a more comprehensive interpretation of accelerometer-derived data becomes possible.

1.3.6 Compression of Athlete Data for Transmission and Storage

Data Compression is an enormous subject area and the breadth and depth of compression techniques available precludes anything other than an introductory analysis of athlete data, with the specific aim of reducing the required bandwidth of the communications and storage channels.

Data compression splits into two distinct types based on the information content, lossless and lossy compression. Loss-less compression is necessary where it is considered that all source data contains information that must be preserved. Lossy compression is based on the concept that the source data contains considerable 'noise' that is not important to the preservation of the contained information. This 'noise' can therefore be deleted during the compression process. The major difficulty in lossy compression of athlete data is identifying the source data components that constitute 'information'.

The previous two chapters are specifically concerned with the extraction of biomechanical or physiological information content. If the required information is the estimated athlete energy expenditure, a 4500bps data stream (150Hz sampling x 10 bits x 3 channels) can be compressed to eight bits (or less) per second, giving a compression factor of 99.8%. This is a prime example of a lossy compression system.

The core focus or analysis of this chapter is identifying the low power compression techniques that match the probability density function (PDF) of athlete data. The PDFs have been generated for a range of athlete activity types and at different sample rates.
and bits-per-sample. Combinations of compression techniques have been implemented, the processing load measured and the compressed data reconstituted for comparison to the source data. The effects of activity level on compression rates have been analysed.

1.4 Research Outcomes

Basic research outcomes have been generated through the various phases of this research. In some cases, future directions for further research have been established.

Low-Power Wireless Networking: Conceptual methodologies for the implementation of low-power wireless networks using different networking technologies have been presented.

Loosely-coupled, low-power, real-time synchronisable Operating System: A small footprint operating system for wireless sensor networks has been demonstrated. This operating system has been utilised in numerous projects, ported to different hardware and used as a teaching concept in university level computer engineering course content. This operating system published and presented at an IEEE conference [4].

Athlete Data Analysis: A range of data analysis and processing techniques were developed for the handling of athlete data. To implement these techniques the Athlete Data Analysis Toolbox (ADAT) was developed, documented and demonstrated to physiologists and other sports scientists. Other researchers used, and are continuing to use these tools to aid their research.

Estimated Energy Expenditure: An anthropometric - biomechanical interdependency was identified as a contributing component in an athlete's sub-maximal running step-frequency. This led to the identification of an energy expenditure estimator based on one measured parameter (step-frequency) and two anthropomorphic parameters (leg-length and mass). This estimator has not been previously published.

Multi-speed, gait-cycle normalised categorisation of biomechanical activity: A data processing technique and categorisation methodology was presented that enables
athletes and sports scientists to interpret and categorise centre-of-mass accelerometer data for running.

Low power, embedded real-time signal processing for extracting energy-expenditure estimators and activity indicators was published and presented at an IEEE conference [8]. An expanded paper was published in the peer reviewed IEEE Sensors Journal [14].

**Internal and External Presentations and Publications.**

**2002:**
IEEE: International Conference on IT and Applications. (Bathurst, Australia)

Cooperative Research Centre for Microtechnology, Annual Conference.
Poster Presentation.

**2003:**
Griffith University: Micro Electronics Research Conference (MERC)
*Acquisition and Compression of Activity Based Data for Transport and Storage* (Presentation & CD-ROM Proceedings)

Cooperative Research Centre for Microtechnology, Annual Conference.
2004
Cooperative Research Centre for Microtechnology, Annual Conference.
Project Presentation (*Grunts vs. Kilojoules*)

Australian Institute of Sport, Project Conference Melbourne October 2004:
Presentation: *Biomechanics of Running*

Australian Institute of Sport, Research Presentation, Canberra, December 2004
Presentation: *Anthropometric factors affecting step frequency and trunk displacement*
Presentation: *Accelerometry derived biomechanics data.*

2005
IEEE Sensors Conference, Irvine California

2007
*Measurement of energy expenditure in elite athletes using MEMS based inertial sensors.*

Asia Pacific Conference on Sports Technology
*A low cost self contained platform for human motion analysis* (Accepted)[15]
1.5 References


2 Wireless Networking

2.1 Introduction

Wireless networking, wireless monitoring, wireless tracking technologies are participating in a period of rapid market expansion. Wireless networking of computers is common with Internet providers providing wireless access points in 'hot spots' such as airports. The government of Texas provides free wireless Internet access at roadside rest areas. Many mobile phones now incorporate not only the inherent wireless technology, but also technologies such as Bluetooth, allowing the phone to communicate wirelessly with headphones, PDAs, printers and other devices. Large retail outlets rely on wireless monitoring of stock as a major component of their anti-theft strategies. Warehouses wirelessly monitor stock locations, and in some cases, this technology is used by businesses to track personnel.

The agriculture sector has used various forms of post-yield monitoring technology as a method of improving subsequent yields. More pro-active management techniques are becoming feasible with the advent of low-cost wireless monitoring systems. The Australian National Livestock Identification Scheme (NLIS) requires wireless tracking of food animals from birth to abattoir and the tracking of the meat product to market. Wireless location of animals is often performed during the conduct of scientific research, with animals ranging in size from whales down to possums and snakes located using wireless technology.

Wireless technology is now more prevalent in the sporting sector of the community. Telemetry for sporting applications such as motor racing is an accepted technology. The use of telemetry in ocean racing is a common tool for television coverage of such events. Runners in marathon races such as the New York Marathon have been monitored for speed and heart rate using a combination of wireless heart rate monitors, wireless telemetered accelerometer stride information, wireless data node (wristwatch) and mobile phone (using Fitsense technology). Runners in this and other marathons have also been tracked using wireless transponders. Position, movement

---

4 A search of the Internet will quickly identify numerous Governments around the world implementing WiFi for the travelling public; in this case 84 rest stops were to be provided.
and the physiological parameters of climbers on Mt. Everest have been monitored from the other side of the globe [9]. One company, Trakus [10], advertised wireless location monitoring of team sport athletes.

Given the above plethora of applications and technology it would appear that an athlete monitoring system that could monitor and/or locate team-sport athletes on-the-field would be an off-the-shelf item. From the time of initial investigation into athlete wireless monitoring through to the current time, no off-the-shelf or easily adaptable technology has been identified. The comprehensive sensor network survey of Akyildiz et al. summarised in [11] makes no mention of sensor networks in a sporting or in a similar mobile context.

This chapter identifies some of the technologies, restrictions and requirements for athlete-monitoring wireless networks. Results from some simple experiments are documented. Some conclusions on the implementations and requirements of the wireless networking for athlete monitoring are presented. The focus of this chapter is the implementation of a team-sport athlete monitoring wireless sensor network and covers the following topics.

- Wireless Sensor Network
- Network Topologies
- Common Wireless Networks
- System Requirements and Constraining Regulations
  - Radio Spectrum Management
  - Health Impacts
  - Size, Weight & Power
  - Data Rate Requirements
  - Antenna Locations
  - Athlete Location Tracking
  - Data Fusion Requirements
- Network Synchronisation
- Implementation Discussion
- Experimental Investigation


2.2 Sensor Network + Wireless Network

A Wireless Sensor Network (WSN) comprises a sensor network communicating via an underlying wireless communications network (Fig. 2.1). The logical and physical layers and the topologies of the layers may coincide but this is not a necessary condition for successful operation. In the context of a WSN monitoring teams of athletes, both the sensor network and the wireless network are likely to be hierarchical in nature. The sensor network would be homogeneous and the data from all sensors fused or combined for analysis by a central host. For reasons of power and bandwidth efficiency, the wireless communications network would typically be managed by a controlling node. The wireless network and the sensor network may share some services such as clocking information.

![Network of Sensor Nodes](image1)
![Network of Wireless Nodes](image2)

Fig. 2.1 Combining Sensor and Wireless Networks to form a Wireless Sensor Network.

The actual implementation of the WSN may involve varying degrees of integration of the hardware components. Fig. 2.2 suggests some arrangements of the macro-components making up a wireless sensor node. Fig. 2.2 (a) & (b) illustrate the generic or basic solution where a microcontroller (MCU) manages both the sensing and wireless networking functions, the RF component only provides the radio carrier and modem functions. In this thesis this form of node, one made of generic or non-specialised components will be referred to as the generic solution. Fig. 2.2 (c) & (d) use a complex wireless module that provides most or all of the required data transport facilities necessary to make the WSN. Some wireless modules incorporate sensor inputs and provide user-programming facilities that allow direct interfacing of the sensors to the wireless module. Not shown in Fig. 2.2 is the totally integrated solution where the sensor and wireless components are integrated within the same device. In this thesis, networks developed around specialised wireless modules will be referred to by the particular wireless technology involved.
(a) Discrete, MCU controls sensing and wireless networking.
(b) As for (a) but RF and MCU integrated in one chip.
(c) MCU controls sensing and interfaces to wireless network module.
(d) Wireless module performs networking and reads sensors

Fig. 2.2 Combinations of macro components to form wireless sensor platform for athlete monitoring

Throughout this chapter, the term node refers to a logical processing point within a network but device indicates the specific implementation under discussion. Terminology may change from general to technology specific where necessary, eg. in the context of IEEE 802.11 wireless networks, a node may be known as a Station.

### 2.3 Network Topologies

Wireless Networks operate in a number of different topologies; mobile telephones would normally be considered hierarchical, with data paths for handset-to-handset communications going via a hierarchy of switching points. An alternative terminology used with mobile telephones is cellular. Mobile telephones connect to a cell-phone base and as the phone moves, it may be handed-off to an adjacent base station. Many wireless networks consist of Point-to-Point links. WSNs can include Point-to-Point, Point-to-Multipoint (Master/Slave), and Peer-to-Peer networks with nodes autonomously forming in partial mesh networks. A WSN can be designed using a combination of topologies. In autonomously organised networks, discovery and routing protocols are necessary both to enable nodes to connect to the network and to route sensor information to the data collector at the lowest power cost per bit. The implementation of protocols for a network of dynamically located sensors has different considerations to those of a statically located sensor network. The term Ad-Hoc applies mainly to peer-to-peer networking although the definition is flexible.
2.4 Common Wireless Networks

The area of WSNs is an emerging technology with applications in small and large-scale plant and animal husbandry, building environmental monitoring and military applications [12]. The University of California at Berkeley, in conjunction with Intel Corporation and others, developed the Berkeley/Intel Motes [13]. These devices are designed so that the total device, silicon, microcode, processor, MAC layer, routing protocols, synchronisation and other protocols, are optimised for low power operation. The "Motes" or Remote Sensors are low data rate, low power, short range (< 10m) and are very small in size. The Motes use a very low powered pre-receiver (1microwatt) that can detect the appropriate preamble\(^5\) and wake the main receiver [14]. The targeted price point is US$0.50 [15], a price that is dependent on the ability to integrate the sensor, the wireless functions and, if possible, the power supply electronics.

As an alternative to the above purely sensor networks, the "ZigBee" consortium [16] is developing RF components for wireless-automation sensor networks. The automation functionality allows light switches to wirelessly control lights or air-conditioning sensors to monitor occupancy, temperature and humidity and wirelessly control the air-conditioning system. This technology proposes wireless sensors with a 10-year battery

\(^5\) The preamble is a component of most asynchronous communications systems and precedes the transmission of the message proper. In the case of a wireless system, the preamble is sufficiently long that it gives time for receivers to detect the presence of wireless carrier and to synchronise the wireless receiver to the transmitter. The preamble may consist of multiple parts designed to synchronise various decoding components in the receiver.
life\textsuperscript{6} [17]. Zigbee uses ISM bands, including the 2.45GHz band, and implements the IEEE 802.15.4 Wireless Personal Area Network (WPAN) standard developed by the IEEE 802.15 working group [18]. Zigbee is only a very recent development but promises some advantages over other WPAN devices such as small protocol footprint (28k byte) and fast network join time (30ms).

The \textit{Wireless Local Area Network} (WLAN) market, following the high market penetration of personal computers (PCs), has driven the development of the WLAN device. These devices include the 802.11a (5GHz, 54Mbps), the 802.11b (2.54GHz, 11Mbps) and the more recent 802.11g (2.54GHz, 54Mbps). The market volume of these devices is such that the price for an 802.11g Adapter is less than A$55 retail. The relevant standards are managed by the IEEE 802.11 working group [19].

The \textit{Personal Area Network} (PAN) is another mass market with worldwide production of the "Bluetooth" [21] PAN device expected to reach 600 million devices P.A.[22]. The target price point for sale to manufacturers is US$5 [22]. Bluetooth is a low power, short range, 2.54GHz device designed as a cable replacement. Bluetooth is used in laptop computers, mobile phones, headsets, printers, cameras, PDAs and other electronic devices. The relevant standards are the Bluetooth 1.2 specification and the IEEE 802.15.1 specification (based on Bluetooth 1.1[20]). These standards are available respectively from the official Bluetooth website [21] and the IEEE 802.15 working group [18].

The \textit{Cellular Mobile Telephone System} (CMTS) is worldwide in scope and offers several different protocols. The most common at this time worldwide are GSM\textsuperscript{7} [23] which operates a combined Time Division / Frequency Division Multiplex arrangement and CDMA (Code Division Multiple Access) [24] which utilises a spread spectrum technology.

\textsuperscript{6} While a 10-year battery life seems impressive, in the context of a light switch used for home automation this may only amount to 1000 seconds of usage.

\textsuperscript{7} GSM originally stood for Groupe Spécial Mobile and it was a study group, formed in 1982, of the Conference of European Posts and Telegraphs (CEPT). The GSM acronym is now more commonly expanded as "Global System for Mobile Communications"
There are many types of networks and devices available. Those listed are noted because they address mass markets requirements and therefore exerted downward price pressure and upward performance pressure in the wireless market sector. A small selection of wireless devices is given in Table 2.1. These are indicative of the devices available in various categories of interest. The existence of a Received Signal Strength Indicator (RSSI), 'Carrier Sense' or a 'sniff mode' affects the ability of a device to detect if the RF channel is available. The parameter nW/bit is an estimate of the nano-Watt cost of transmitting a single bit of data at the peak available data rate, eg when operating in the most efficient mode.

Table 2.1 Comparison of specifications of devices from various wireless technologies.

<table>
<thead>
<tr>
<th>Motes</th>
<th>WPAN</th>
<th>WPAN</th>
<th>WiFi</th>
<th>ISM</th>
<th>ISM</th>
<th>ISM</th>
</tr>
</thead>
<tbody>
<tr>
<td>I(ma) Tx/Rx</td>
<td>18/20 @ 1.6V</td>
<td>45/45 @ 1.8V</td>
<td>133/200 @ 3V</td>
<td>25/11 @ 3V</td>
<td>8/14 @ 3.3V</td>
<td>29/23 @ 3V</td>
</tr>
<tr>
<td>nW/bit Tx/Rx</td>
<td>115/128</td>
<td>112/112</td>
<td>7/11</td>
<td>3750/1650</td>
<td>412/721</td>
<td>1133/898</td>
</tr>
<tr>
<td>Range</td>
<td>&lt;10m</td>
<td>70m</td>
<td>100m</td>
<td>100m</td>
<td>200m</td>
<td>200m</td>
</tr>
<tr>
<td>bps</td>
<td>&lt;&lt;10k</td>
<td>250 k</td>
<td>720 k</td>
<td>54 M</td>
<td>20 k</td>
<td>64 k</td>
</tr>
<tr>
<td>Join Time</td>
<td>30ms</td>
<td>3s</td>
<td>2s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freq. Hz</td>
<td>1.9G</td>
<td>2.45G, 915M, 868M</td>
<td>2.45G</td>
<td>2.45G</td>
<td>433M</td>
<td>433M</td>
</tr>
<tr>
<td>Protocol Stack</td>
<td>Physical to App.</td>
<td>Physical to App.</td>
<td>Physical to App.</td>
<td>PHY MAC</td>
<td>PHY</td>
<td>PHY</td>
</tr>
<tr>
<td>RSSI or Sniff</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Figures given in this table are from manufacturer's data sheets, from standards or from other sources. Transmitter power and resultant current draw is usually software controllable and may reduce at lower data rates. Ranges are variable based on topography, activity, inclination, antenna type and other factors. The variety of highly complex RF components is constantly growing.

---

8 Sniff mode is used by low power receivers to save battery life. The transmitter precedes the message with a relatively long preamble. All receivers power up the receive circuit at an interval shorter than the preamble length. The receiver is only powered up long enough for the RSSI to detect carrier and if carrier is detected the receiver powers up fully and begins capturing the incoming message. If carrier is not detected the receiver shuts down. This system is used in pagers [31], Bluetooth and various RF components.
2.5 System Requirements and Constraining Regulations

Any system being developed for any purpose, with a manufacturable item as the end result, will need to fulfil both the system requirements and any number of Government regulations or standards. In many cases the regulations or standards may have little effect on the overall system design, dealing mainly with the manufacturing processes or packaging design.

In the case of a wireless device to be worn by elite athletes engaged in competition, the design requires numerous trade-offs between system requirements, health-related regulation, wireless-related regulation, available technology, venue of operation etc. The key system requirements, affecting regulations and resultant available design choices are enumerated here:

2.5.1 Radio Spectrum Management

In Australia, as in most other countries, the radio spectrum is controlled by legislation and appropriate standards.

Since the required telemetry system is to be a low cost, commercialisable system with multiple nodes, it was appropriate to consider using unlicensed frequency bands. The alternative would heavily impact on the take up of the technology worldwide. It would also impact the production costs since available frequencies would be different from locality to locality.

The frequency selected from those frequencies available world wide for unlicensed operation could either be in a designated Industrial, Scientific, Medical (ISM) band or covered by a specific class license. A list of ISM bands and Australian Low Interference Potential Device (LIPD) class license frequencies is given in Appendix Table 9.6.

LIPD licenses, or their equivalent in other countries, stipulate a number of conditions such as maximum Effective Isotropic Radiated Power (EIRP), duty cycle, channel existence and associated parameters. For the 433MHz band, some of these appear in Appendix Table 9.1 and Table 9.2.
The international community is currently negotiating to clear a number of bands globally to allow standardisation of technologies such as Wireless LAN, Mobile Telephones, Digital Cordless Telephones (DECT), Personal LAN, Wireless Medical equipment et.al. In Australia the Wireless LAN bands were originally covered under a separate Spread Spectrum license. In 2005 this license was disbanded and these technologies moved to the LIPD class license. This work may result in clarification of the status of particular frequencies/bands leading to the selection of an appropriate implementing technology.

2.5.2 Health Impact
In Australia, Radio Frequency (RF) transmitters are required to operate within the constraints of the appropriate Australian Standards for electromagnetic radiation exposure levels [32]. Generally the low anticipated power precludes this being a problem, however for frequencies from 800 MHz to 2.5 GHz (includes Cellular telephone technologies), where the device is likely to be operated within 20cm of the wearer's head, as is the case with head or neck mounted devices, testing must be conducted to identify the Specific Absorption Rate (SAR) of a human head exposed to the device to ensure it is within specification.

Some experimental work has been conducted that appears to indicate that exposure to 2450MHz signals has a short-term detrimental effect on the cognitive ability of rats [33] although other studies fail to observe any such effects. Proponents of some of these technologies consider that since these devices operate at much lower power levels than mobile telephones there is less health concern [34].

2.5.3 Size, Weight & Power Constraints
The athlete monitoring equipment requirements include an encapsulated sensor, processor, power and radio frequency equipment of a size small enough to be

---

9 2450MHz is the operational frequency of the common microwave oven as well as the approximate centre frequency of the Bluetooth, IEEE802.11b/g Wireless LAN, HomeRF™ LAN and some cordless phone technologies. The selection of 2450MHz for use in microwave ovens is based on a compromise between the ability of the microwave to penetrate an object (say a chicken) and the absorption of microwave energy by water molecules.
comfortably worn by an elite athlete. This generally means the device will not have interchangeable components and the physical capabilities of the device will be locked in a sealed unit at production time.

Devices require an internal source power capable of sustaining athlete sensing and wireless telemetry functions for up to four hours. Radio Frequency (RF) power and duty cycle is therefore limited by the capacity of the internal power supply. Non sensor-equipped data relay nodes are less restricted by size & weight. These data relay nodes need to receive data from the sensors and transmit that data to the central station.

### 2.5.4 Data Rate Requirements

As real time sensors generate large quantities of raw data and, in the team sports, there will be multiple systems; the potential required data rates could be very high. Estimated data rates are given in Table 2.2. These estimates do not include network overheads such as synchronization, data packaging, error detection/correction, data redundancy and other miscellaneous requirements. In multiple athlete or multiple remote device networks operating in real time, the networking and redundancy overhead requirements could exceed the required sensor data rates.

<table>
<thead>
<tr>
<th>Example System Data Rates</th>
<th>bps = bits per second</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System</strong></td>
<td><strong>Rate</strong></td>
</tr>
<tr>
<td>1 Athlete, sensor derived physiological effort measure</td>
<td>8 bps</td>
</tr>
<tr>
<td>1 Athlete, with tri-axial accelerometers @ 100Hz 8 bit Sampling</td>
<td>2,400 bps</td>
</tr>
<tr>
<td>Single Four-Sensor Force Shoe @ 500Hz 8 bit Sampling</td>
<td>16,000 bps</td>
</tr>
<tr>
<td>8 oars with tri-axial magnetic sensors @ 100Hz 8 bit Sampling</td>
<td>19,200 bps</td>
</tr>
<tr>
<td>12 Athletes with tri-axial accelerometers @ 100Hz 8 bit Sampling</td>
<td>28,800 bps</td>
</tr>
<tr>
<td>34 Athletes with tri-axial accelerometers @ 100Hz 8 bit Sampling</td>
<td>81,600 bps</td>
</tr>
<tr>
<td>50 Athletes, sensor derived orientation, effort and activity (4 bytes)</td>
<td>1,600 bps</td>
</tr>
</tbody>
</table>

Sensor data rates can be reduced by various system modelling or compression techniques. Chapters 4, 5, 6 and 7 investigate the extraction of information of various forms and the compression necessary to reduce required data rates.
Frequency of operation affects available data bandwidth. Higher frequency bands have relatively wide channel bandwidths and consequently channels can carry higher data rates. Table 2.9 (Section 2.9.3) defines 153kHz channels in the 433/868/915MHz ISM bands and the component operates up to 76.8 kbps. The Bluetooth and Wireless-G (802.11g) systems operate in the 2400MHz – 2483.5MHz range which is divided into seventy-nine 1MHz channels. Bluetooth has a raw data rate up to 1 Mbps while the Wireless-G device transmits at a maximum of 54Mbps. Systems operating at lower frequencies are limited to much lower bandwidth and hence lower data rates. While higher data rates require more power for the same physical coverage distance, the cost per bit is reduced significantly (Table 2.1).

2.5.5 Operating Environment & Required Path Length

In general, the operational environments will usually be open space, either in a stadium or in an open field. Path lengths required for sensor telemetry range from 200-300 metres for athletics down to a few metres for sports such as boxing or canoeing (in-canoe networking). The path generally can be line of sight. A data relay or intermediate link node needs to transmit a signal over several kilometres for activities such as canoeing; however weight, while still important, is not as critical in canoeing as in sprinting.

In some sports such as football, the devices may suffer high attenuation due to the close juxtaposition of many athletes. The actual location of the device on an athlete will impact the RF networking given that the human body attenuates the signal significantly. This attenuation is frequency dependent with higher frequencies being attenuated more. The diagram below represents the effect of increasing frequency on the polar plots for an antenna mounted on a person.

10 The 2400MHz – 2483.5MHz band is not fully available in some countries therefore fewer channels are available, 79 channels in many countries, 23 channels in others. The 802.11b/g standards use both Frequency Hopping Spread Spectrum (FHSS) (79 channels) and Direct Sequence Spread Spectrum (DSSS) which divides the same band into five overlapping 26MHz channels.
2.5.6 Antenna Size & Orientation

Antenna types, sizes and orientation are affected by both the restrictions required for the physical implementation and the available link budget. The link budget takes into account the available transmitter power (TxP), losses in RF connecting cables (TxL), antenna gain (TxG in dBi), propagation loss (FSL), receiving antenna gain (RxG), cable losses (RxL) to calculate the received power. The link budget receive power must be greater than receiver sensitivity. By calculating the link budget, the required antenna gains can be determined.

Equation 2.1   \( \text{Link Budget} = \text{TxP}-\text{TxL}+\text{TxG}-\text{FSL}+\text{RxG}-\text{RxL}=\text{RxP} \)

Table 2.3 Comparison of Transmitter Power, Receiver Sensitivity and permissible EIRP

<table>
<thead>
<tr>
<th>Technology</th>
<th>Zigbee</th>
<th>Bluetooth</th>
<th>WiFi</th>
<th>ISM</th>
<th>ISM</th>
<th>ISM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>ChipCon</td>
<td>Philips</td>
<td>Philips</td>
<td>Nordic</td>
<td>Radiometrix</td>
<td>Nordic</td>
</tr>
<tr>
<td>Part</td>
<td>CC2420</td>
<td>BGB203</td>
<td>BGW211</td>
<td>nRF403</td>
<td>BIM2</td>
<td>nRF903</td>
</tr>
<tr>
<td>Tx Power dBm</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>10</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Rx Sensitivity (dBm)</td>
<td>-94</td>
<td>-85</td>
<td>-73</td>
<td>-105</td>
<td>-96</td>
<td>-100</td>
</tr>
<tr>
<td>Maximum EIRP mW (dBm)(^\circ)</td>
<td>4W (36)</td>
<td>100(^\circ) (20)</td>
<td>4W(36)</td>
<td>25 (14)</td>
<td>25 (14)</td>
<td>25 (14) 1W (30)(^\ast)</td>
</tr>
<tr>
<td>Potential Antenna Gain</td>
<td>36dB</td>
<td>15dB</td>
<td>21dB</td>
<td>4dB</td>
<td>8dB</td>
<td>4dB 20dB(^\ast)</td>
</tr>
</tbody>
</table>

\(^\circ\) Maximum EIRP from Table 9.5, Section 9.1.3.

\(^\ast\) The class license conditions (Item 54 in Table 9.5) allow a maximum of 4W but the Bluetooth specification limits the power to 100mW.

\(^\ast\) Operating this multi-frequency component in a hopping sequence with more than 20 frequencies in the 915-928MHz allows a maximum EIRP of 1W (Item 52 in Table 9.5)
Equation 2.2  

\[ \text{Free Space Loss} = 10 \log_{10} \left( \frac{4\pi d}{\lambda} \right)^2 \]

Table 2.4 Free Space Loss (FSL) calculated for Frequency and Distance (using Equation 2.1)

<table>
<thead>
<tr>
<th>Distance (m)</th>
<th>Frequency 433MHz</th>
<th>Frequency 2.45GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>45dB</td>
<td>60dB</td>
</tr>
<tr>
<td>50</td>
<td>59dB</td>
<td>75dB</td>
</tr>
<tr>
<td>100</td>
<td>65dB</td>
<td>81dB</td>
</tr>
<tr>
<td>150</td>
<td>69dB</td>
<td>84dB</td>
</tr>
</tbody>
</table>

Note: Free space loss is a 'best case' loss assuming no interference reflections etc. In real life, in the sporting environment, these losses will be much greater.

**Example 433MHz Link Budget:**

433MHz nRF403 component with 50m separation between athlete and base station.

\[
\text{Link Budget} = \text{TxP-FSL-Receiver Sensitivity} = 10\text{dBm}-59\text{dB}- (-105\text{dBm}) = 56\text{dB}.
\]

The antenna designer has 56dB available if required. If a miniaturised antenna in the encapsulated WSN node has a loss of 10dB and the attenuation caused by the player's orientation relative to the base station is 16dB, there is still a 30dB buffer (Assuming the base station antenna and associated cables do not have a cumulative loss).

**Example WiFi 802.11g Link Budget:**

2.45GHz using BGW211 component with 100m separation between athlete and base.

\[
\text{Link Budget} = \text{TxP-FSL-Receiver Sensitivity} = 15\text{dBm}-81\text{dB}- (-73\text{dBm}) = 7\text{dB}.
\]

There is insufficient RF gain in this system; the 7dB would easily be lost in player attenuation or orientation relative to the base station. Options include reducing the data rate (improves the sensitivity) or increasing the overall system gain. Assuming an antenna gain of -10dBi on the athlete mounted system and a base station antenna gain of 14dBi the 7dB buffer grows slightly to 11dB. Geographic diversity of bases could be used to reduce the distance to 50m and the FSL to 75dB. (System gain now 17dB).
**Antenna Design & Orientation**

From the above link budget calculations, the necessity of obtaining the maximum gain from the antennas is apparent.

For maximum efficiency in converting battery power into radiated RF power the radiating antenna must be carefully matched to the drive circuit. The transmit antenna located inside the encapsulated device requires a radiation pattern that is approximately isotropic (omni-directional) since the orientation of the antenna will change with body movement. The changing orientation of the device changes the polarisation of the RF signal resulting in a changing link budget. For this reason the antennas must accommodate such variations.

Antenna design is heavily dependant on wavelength and hence the radiation frequency. At lower frequencies a larger antenna is required to achieve the same link budget. There is a minimum frequency limit for effective transmission from ultra miniature devices. At 433MHz, a quarter wave antenna is approximately 17cm but at 2.45GHz the antenna length drops to 3cm. While athlete mounted antennas have size restrictions that directly impact on antenna directionality, the base station antennas can be relatively large and therefore highly directional.

While the development of miniaturised antennas for personal communications systems is highly complex [35], there are a variety of suitable antennas available commercially. Antenna selection is a trade-off between efficiency, radiation pattern, size and cost while addressing the major issue of ensuring adequate signal to noise ratio at the receiver. Some modern RF components include the antenna inside very small packages\(^\text{11}\).

### 2.5.7 Multi-link paths

Data telemetry from a number of sensors can be collected by an intermediate node and retransmitted (store & forward). The intermediate node may be a fixed node, located to give additional coverage, or a mobile node such as the node on another athlete. In an

---

\(^{11}\) Spectec (www.spectec.com.tw) produce a 802.11g microSD card with incorporated antenna. The unit measures 23.3 x 11 x 0.7 mm with a weight of approximately 1.5g. (Receive sensitivity -73dBm)
agricultural setting, higher-powered nodes would act as data collection points for geographically localised groups of sensors. More closely located sensors using lower powered transmitters and receivers can carry data at a greatly reduced power cost per bit [36].

In team sports such as football, the size of the ground, the signal masking by athlete bodies and the numerous possible sources of interference means the network may require multiple base stations and therefore require the use of multi-link paths.

### 2.5.8 Athlete Location Tracking

Given the tracking applications identified in the introduction it would appear that tracking is a mature technology. In most instances, the tracking systems use RFID transponders that identifies the sensor at a specific location, eg marathon runners with shoe mounted RFID transponders pass over a sensor mat at a specific point in a race. This does not provide continuous information about athlete location at any point on a playing field. The Orad system [37] uses a system of directional 'radar'. RF signal pulses are transmitted across the playing field by directional antenna. The RF signal stimulates into transmission athlete-mounted microwave diodes and this signal is detected by the monitoring system.

For some sporting activities, the athlete-mounted devices perform the dual role of athlete data telemetry and location beacon. The location technique depends on the technology used for wireless node implementation. In a purely time division network or a fixed sequence frequency-hopping system, wireless nodes could be tracked by triangulation [38] using a number of electronically steerable directional antenna [39] [40]. In this scenario, the location detection nodes need to monitor the time slot, frequency, received signal strength and direction of each node (Fig. 2.5). Although triangulation requires directional information from three bases, in the event of occasional incomplete location data, the network host may still accurately infer position from the time series positional and athlete activity information.
If all nodes incorporate RSSI and have sufficient data bandwidth, alternative statistical location algorithms can be used [41]. In this case, both the fixed and mobile nodes monitor the other nodes via their own RSSI. This information is collated and the location determined (Fig. 2.6). Using the RSSI from mobile bases vastly increases the required power since mobile nodes must monitor all transmissions to identify the transmitting node and to measure the received signal strength. This additional information must also be transmitted back to the host, greatly increasing the volume of data traversing the network. Alternatively the number of fixed nodes could be greatly increased to reduce the power burden on the mobile nodes.

Statically located nodes have radiation patterns covering the field of play. Athlete mounted nodes have radiation patterns dependent on the antenna location on the body (in this example back mounted antenna).
Clearly location tracking via directional antennas and triangulation has a number of advantages.

- The mobile, power-restricted, nodes are not required to monitor the other nodes.
- The bandwidth required for transmitting positional information is less.
- The required processing power for position calculation is less.
- For triangulation using directional receiving antennas, direction is independent of field strength so the position of the antenna on the athlete does not affect the detected direction.

This discussion is rather simplified as there are numerous technical obstacles that are dependent on the system topology, the wireless protocol, the transmitter on time, the sporting venue, the number of interference sources, speed of operation of the electronic steerable antenna etc.

### 2.5.9 Data Fusion Requirements

Data fusion describes the process of combining data from multiple sensor nodes to obtain an overview of the system. A sensor network monitoring stress in a structure may have low time criticality and so differences in the timing of measurements may be as large as minutes and still the sensor data, when fused, gives an adequate overall picture of the steady state distribution of stress. Alternatively, sensor networks monitoring time-of-flight over short distances may require millisecond or microsecond timing precision.

For athlete monitoring, the data-fusion timing requirements vary from sport to sport. The data-sampling rate, in combination with the signal filtering, may limit the precision of event detection but the wireless network is required to maintain the clock synchronisation. The synchronisation-error may limit the ability to associate key simultaneous events such as the impact recorded by an instrumented boxing glove and the impact recorded by the opponent's instrumented vest.

Video recording of athlete performance is the most common form of data collection and when video data is combined with sensor data, the maximum synchronisation clock error for data fusion is governed by the picture rate of 25Hz. This becomes a
50Hz display rate. If sensor data is combined with video, the sensor node timing synchronisation should be of the same order as the frame timing (40ms or better). This ensures that all simultaneous events appear in the same frame or within two adjacent frames. For high-speed video (250Hz), the sensor node timing synchronisation should be 4ms or better. This ensures that data from the same event appears in either the same frame or adjacent frames. Both these values are larger than the synchronisation timing required to maintain a *Time Division Multiple Access* (TDMA) wireless network.

**Individual Competitive Events:** In events such as swimming and running, fusing athlete data with track data may be useful for coaching purposes. The timing requirements lie in the same range as the timing used to measure the event at the elite level, eg 0.01 seconds.

**Boxing:** This event is traditionally scored using boxing judges and a computer system requiring near simultaneous presses of a scoring button. Using WSNs, the data fusion is required to align the pre-punch acceleration of the glove, the post impact deceleration and the impact on the opposing boxer. The latency between the detected event and data fusion is of lesser importance than the timing accuracy of the sensors themselves. Network latency from contact to scoring should be shorter than human response time to ensure that human judges are not disconcerted by an excessive delay in the system. The timing synchronisation of the sensors would be one twenty-fifth of a second or better to match video picture rate.

**Canoeing:** In the case of instrumented canoeing, particularly when monitoring larger teams, the timing synchronisation of the sensors required is one fiftieth of a second or better. This is beyond the limit of visual perception.

**Contact Sport:** Team sports such as Rugby Union involve contact between players. These contacts can be very brief and can be part of a sequence of contacts. These contacts need to be correctly associated to identify the participants involved in the

---

12 This refers to the system used in Australia and a number of other countries where the full picture is redrawn 25 times a second but a screen of alternate lines is drawn at 50Hz. High speed video or systems used in France, the USA and elsewhere have different picture rates.
contact. As the peak force detected by the accelerometers can be isolated to better than a tenth of a second (depends on filtering, samplerate etc), the sensor node timing synchronisation should be better than this.

**In-Shoe plus Centre of Mass:** Where a wireless network combines data from multiple sensors from the same athlete, it might appear that timing requirements are very strict. This is not necessarily the case. For a sample rate of 500Hz, shifting centre-of-mass samples by five or even ten samples relative to in-shoe data is virtually imperceptible in graphically reconstructed data. This variation may even fall within the natural step to step variation. Regardless of this particular application, a system designed for synchronous 500Hz sampling implies that the samples are taken at the same instant therefore a ±2ms alignment is required. For in-shoe + centre-of-mass monitoring, the allowable drift becomes dependent on the final use of the data. A sprinter running with a 5Hz step rate and a stance\(^\text{13}\) time of 20% has a contact period of 40ms. A 10% error equates to 4ms, which is also the frame rate of a 250Hz\(^\text{14}\) high-speed video system. For this purpose 4ms would appear to be the maximum allowable drift.

### 2.6 Synchronisation

In a **wired** sensor network, synchronisation of sensor information can be established by the use of a distributed system clock. Latency of data transmission can be measured in nanoseconds, bandwidth is generally not an issue and the power supply is only a consideration in terms of decoupling and noise suppression. When measuring mechanical, biomechanical or physiological systems using a **wired** sensor network, sensor synchronisation, data latency, network bandwidth, available power and data fusion are often not of significant importance.

In **WSN** design, the non-issues of a wired network become the core issues. Wireless sensors share a common channel with limited bandwidth. To maximise the available bandwidth sensor nodes require good synchronisation. In the absence of a wired

---

\(^{13}\) Stance time is the percentage of the gait cycle (one stride) that a particular foot is in contact with the ground. This is covered in detail in Chapter 6.

\(^{14}\) In this case the 250Hz frame rate is that of the Redlake "Motion Scope" PCI High Speed Digital Imaging system used by the AIS.
distributed clock, the network must use the limited bandwidth to distribute a clock signal. Wireless sensor nodes employ RF transmitters and receivers each of which consume power. At some time during communications frames, all wireless nodes must capture clocking information. They consume considerable power in maintaining this synchronisation. Network synchronisation affects the available signal bandwidth, the capacity to accurately fuse data and the power consumption.

In some cases, synchronisation is required purely for data fusion; alternatively it is required purely for network control. More commonly, sensor networks require synchronisation for both accurate data fusion and network functioning. If one requirement has less stringent constraints than the other, one might assume these requirements have been met if the more stringent requirements are met. However, if the wireless networking layer is implemented independent of the sensor monitoring and analysis functioning, explicit synchronisation of the sensor system clock may still be required.

Synchronisation schemes must be designed to minimise the bandwidth and power used while meeting the timing requirements within the topology and operating constraints of the application specific WSN.

2.6.1 Loss of Synchronisation

Loss of synchronisation occurs when the time bases of two devices, initially synchronised, drift sufficiently that the devices no longer communicate or data fusion no longer provides the required space-time accuracy. The drift of time bases is generally due to manufacturing precision, age and temperature of the clock crystal (usually a controlled oscillator). As the following example shows, even a high accuracy crystal cannot maintain synchronisation without regular resynchronisation.

Example: Assume two devices occupy adjacent time slots in a TDMA scheme. Both use crystals with a 50 parts-per-million (ppm) precision and additional aging and temperature effects of 50ppm. The inter-device worst-case drift is 200ppm or 200µs per second. On a 4ms inter-timeslot separation with no timing correction applied, the timeslot separation could be lost in 20 seconds and data fusion synchronisation of 40ms lost in 200s. Much higher quality crystals can give an order-of-magnitude
improvement in drift, the timeslot separation would be lost in 200 seconds. Clearly some form of network synchronisation is required.

Table 2.5 lists a number of crystal technologies, their precision and the effect of their precision on the relative drift between two independent clocks implemented with the same crystal technology. As the crystal is often the source of a clocking signal distributed internally within a microprocessor, specific timing intervals are generated using a timer/counter counting a set number of clock pulses. If the time interval of one device is marginally different to the time interval of another device due to the difference in crystal frequency, the time interval difference may be correctable by using different clock counts. The 'precision' of the counter is a function of the crystal frequency, the required timing interval and the resolution of the timer/counter. Two timer/counter examples appear in Table 2.5.

Table 2.5 Clock Precision for Crystal Oscillator Types (from [42]).

<table>
<thead>
<tr>
<th>Technology</th>
<th>Precision</th>
<th>Relative Drift</th>
<th>Per second drift</th>
<th>1ms Drift</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCXO</td>
<td>±0.01~±0.1</td>
<td>0.2 ppm</td>
<td>200ns</td>
<td>5000s</td>
</tr>
<tr>
<td>MCXO</td>
<td>±0.1~±1</td>
<td>2 ppm</td>
<td>2µs</td>
<td>500s</td>
</tr>
<tr>
<td>TCXO</td>
<td>±1~±3</td>
<td>6 ppm</td>
<td>6µs</td>
<td>167s</td>
</tr>
<tr>
<td>XO</td>
<td>±25~±50</td>
<td>100 ppm</td>
<td>100µs</td>
<td>10s</td>
</tr>
<tr>
<td>16bit counter</td>
<td>±20</td>
<td>40 ppm</td>
<td>40µs</td>
<td>25s</td>
</tr>
<tr>
<td>24bit counter</td>
<td>±0.1</td>
<td>2 ppm</td>
<td>2µs</td>
<td>500s</td>
</tr>
</tbody>
</table>

OCXO Oven Controlled Crystal Oscillator (*requires heater current ~ 100mA [43])
MCXO Microprocessor Controlled Crystal Oscillator
TCXO Temperature Compensated Crystal Oscillator
XO Crystal Oscillator

# Per-second drift is independent of crystal frequency and can be calculated directly from the relative-drift eg. 2ppm = 2 millionths of a second or 2µs.

2.6.2 Synchronisation Taxonomy, Models and Language

Anceaume & Puaut proposed a taxonomy of wireless network synchronisation [44] which forms a basis of much subsequent published work. Sundararaman, Buy & Kshemkalyani presented a survey of wireless synchronisation proposals [45]. A number of proposals start from the basis of the Internet Network Time Protocol (NTP)[46] and extend this concept into the arena of wireless networking. Many of these networks assume complex topologies of unknown depth and breadth. The
synchronisation schemes were developed entirely in hardware or software or by some combination of the two. Tighter synchronisation is obtained using hardware systems [44]. Some of the proposed and implemented techniques include flooding protocols [47], post-facto synchronisation [48], tree based algorithms [49], pair-wise synchronisations and a variety of similar and derivative schemes. The resultant systems provide deterministic, probabilistic and statistical synchronisation solutions. The method used can affect the available power and bandwidth. To conserve bandwidth and power resources it is common for synchronisation schemes to use error estimates to provide time estimates in the absence of synchronisation messages [47] [48]. Error estimates are used to adjust the local logical clock in software or the physical clock in hardware. Hardware adjustment includes the use of voltage controlled or microprocessor controlled crystal oscillators (VCXO & MCXO)[42].

Kopetz and Schwabi [50] provide useful definitions in the development of a synchronisation model, abbreviated versions are:

- **Send time.** The time spent assembling the message at the sender.
- **Propagation time.** The time for the signal to propagate across the physical medium between two nodes.
- **Receive time.** The processing time required for the receiver to receive a message from the channel and notify the host of its arrival.
- **Access time.** The delay associated with accessing the channel, including carrier sensing.

These definitions are used in this thesis in subsequent sections.

### 2.6.3 Synchronisation Implementation Discussion

While there are a number of potential synchronisation schemes for ad-hoc wireless networks, not all schemes can be implemented in a team sport WSN. The continuous absolute and relative positional changes in conjunction with the overall system data rate, available power and the requirement for near real-time data, result in some different demands with regard to routing and in particular, synchronisation protocols. The use of ad-hoc networking may be inappropriate in this environment.

Using the Anceaume & Puaut taxonomy and given the nature of a team sport WSN, the asymmetric slave-controlled synchronisation scheme appeared to be the most
appropriate. This scheme can be considered native to TDMA systems. 'Asymmetric' indicates a master-slave system where the master provides a reference time on which the slaves resynchronise. In a master-controlled scheme, the master coordinates a clock synchronisation algorithm. Anceaume & Puaut suggest that one of the problems of such a scheme is the large power demand associated with the synchronisation messages. This problem can be overcome by implementing an appropriate error estimate algorithm in the power limited slave nodes. This reduces the number of transmitted synchronisation messages that the slave needs to monitor.

Both slave-controlled and master-controlled schemes are implementable - depending on the hardware and protocol used. Different slave-controlled schemes and topologies are discussed below. A low-power variant of a master-controlled scheme running on WiFi is also discussed.

**Slave-Controlled Reference Broadcast Scheme Variants**

The Reference Broadcast Scheme (RBS) proposed in [51] is applicable to solutions using both generic hardware and complex implementations such as WiFi. The RBS implements a MAC layer clock pulse message that is broadcast to all listening nodes. On receipt of this message the MAC layer of the receiver timestamps the message. In an RBS scheme, local events are time stamped with an offset from the RBS pulse. In this scenario, send-time, access-time and propagation-time become irrelevant. With the RBS scheme, access-time is not a consideration, since receivers measure time from the receipt of the pulse. Similarly, the send-time is not considered, and since a typical team-sport venue is small relative to the propagation rate, propagation-time is inconsequential in comparison to the required time accuracy. Receive-time, which depends on the implementation, is the only source of error and has been shown to be gaussian in distribution for various platforms [51].

With the implementation of a sensor network on top of a developed wireless protocol, RBS provides the accuracy required for data-fusion. For RBS to function the receivers must be capable of detecting and decoding the RBS pulse. This may require the receiver to be powered more often than necessary, or require the messages be preceded by a significant preamble so that the message start can be detected by a receiver in
sniff mode. RBS is not native to Bluetooth since this operates on a strict TDMA system with timeslots allocated to master or slave operation.

A more appropriate synchronisation scheme is formed by combining a TDMA wireless network with the sensor synchronisation. When used in a TDMA solution implemented on generic hardware, the RBS clock pulse is replaced with a repetitive master-to-slave TDMA synchronisation pulse. In this situation access-time is deterministically controlled, propagation-time is still irrelevant and receive-time still has a gaussian distribution. Variation in send-time may cause a small amount of jitter at the receivers. This can be minimised if the operating system prioritises sending the clock-pulse message above other master node activities.

The power load at slave nodes due to TDMA can be reduced by using clock error estimate algorithms. By initially monitoring sequential TDMA synchronisation pulses and determining the error of the local clock relative to the master clock, the slave node can reduce power by listening for fewer clock pulses and applying a local adjustment based on the measured error. Depending on the network operation, topology and effectiveness of error estimate algorithms, slave nodes may only need to listen to 5% of clock pulse messages. If an expected pulse is absent then the slave node can begin monitoring subsequent pulse positions until a clock pulse is captured. The error estimate can be used to modify the device's local crystal frequency, in the case of MCXO and VCXO, or alternatively used to modify the counting of clock cycles, refer Fig. 2.7. Post-facto synchronisation [48] evaluates the effectiveness of controlling of the crystal frequency of the slave node.

![Slave synchronisation using controllable oscillators or controllable counters.](image)

Fig. 2.7 Slave synchronisation using controllable oscillators or controllable counters.
The concept of using counters is further examined in Fig. 2.8, which depicts the possible implementation of clock correction using both 16 and 24 bit hardware counters. In this example, two receivers attempt to adjust their timing to match a master clock. The receiver using the 16 bit counter can only get within ± 10ppm (10µs per second) and will therefore alternately run marginally fast or marginally slow. This jitter can be estimated and allowed for in the error estimate. Using the 24bit counter, while an accuracy of ± 0.1 ppm is theoretically achievable, in practice the accuracy is controlled by the *receive-time* distribution, i.e. it is dependent on what operations the processor is performing when it is interrupted at the completion of the receipt of the timing message.

The implementation of TDMA in the topology depicted in Fig. 2.5 or Fig. 2.6 requires additional support. In this case, there are a maximum of two links involving one intermediate node. Any slave node that is out of RF range from the master node must obtain synchronisation from an alternative source. This can be arranged by alternating the source of the synchronisation clock pulse between the available static nodes. In a combined TDMA - *Frequency Hopping Spread Spectrum* (FHSS) scheme, the clock pulse can be made available on different channels based on the hopping sequence. This helps protect against loss of synchronisation due to interference on a single channel.
Similarly, using an RBS pulse in the given topologies, different static nodes can transmit an identifiable clock message. The network host, as well as transmitting its own RBS pulse, would detect the RBS pulses from the other static nodes. Mobile nodes transmit their data with the time offset from the RBS pulses of a specific static node. In this way the data synchronisation can be implemented accurately.

In a TDMA system with multiple clock sources some jitter is introduced. This affects the clock error estimations in the slave nodes. This compounding error is identified in [49] and depicted in Fig. 2.9. Where multiple clock sources exist, the send-time of the clock pulse must be calculated to ensure that the clock repeaters complete transmission as close to the correct interval as local error allows. It is assumed that the static or intermediate nodes used as clock repeaters are not as power limited as the mobile, athlete-mounted nodes. These clock repeaters can obtain their synchronisation either in-band or out-of-band from the master node and, with less restrictive power budgets, can maintain a stricter error limit.

![Fig. 2.9 Effect of Multi-link Path on Network Synchronisation](image-url)
Synchronisation on Bluetooth

For a sensor network running on top of a Bluetooth wireless network, sensor synchronisation can be obtained by extracting Bluetooth wireless synchronisation information or by using the wireless network to pass synchronisation messages. Synchronisation error can be reduced by identifying the sources of any errors. In the case of a synchronisation message carried on a Bluetooth network, the latency of the network is important. A Bluetooth master transmits in specific timeslots. A repeating node cannot utilise the same timeslot therefore the receiver must account for the timeslot offset between the master and the repeater. Many synchronisation schemes use the exchange of clock information between pairs of nodes to determine the relationship between the local and remote clock [49][53]. Where a multi-hop Bluetooth network is involved the value of the Bluetooth internal 28-bit clock (used to manage the TDMA access) at both sync message receipt and retransmission can be used to provide correction to downstream data fusion clocks.

Synchronisation on Generic Hardware

In a generic hardware based system (refer Fig. 2.2 (a) & (b)), the microprocessor handles sensor monitoring, data packetisation and the wireless protocols including synchronisation. The receive-time is affected by the active processing and sensing activities, since these affect the time to respond to an interrupt. A generic hardware intermediate node can reduce error by using a more accurate or tuneable crystal. Intermediate node send-time would normally have a very tight distribution since the only task at this point in time is to transmit the clock pulse.

Synchronisation on WiFi

If sensor network synchronisation relies on an underlying WiFi network, depending on implementation, the access-time can impact synchronisation. In a situation where clocks are exchanged or a message must be sent at a specific instant, the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) function of the WiFi may cause the message to be delayed due to other traffic. RBS overcomes this problem since RBS accuracy only depends on the receive-time. If a TDMA scheme is implemented on WiFi then the clock message must be transmitted precisely at the required time i.e. the transmission must be forced even if other traffic appears to be using the channel. Considering that a relatively large packet of sensor data can be
transmitted in less than a millisecond, despite the delay due to CSMA/CA, clock swapping may be an appropriate synchronisation technique for team sport monitoring. By transmitting the local clock with the sensor data, and requesting a clock in the reply, the mobile node switches direct to RF receive and the first static node to transmit gets the channel. The complete transaction may take less than 2ms, well within data fusion requirements. Clock swapping would not need to occur on every transmission of data.

**Low Power Master-Controlled WiFi Synchronisation**

Master-Controlled indicates that the master node calculates all the clock error information prior to correcting the slave nodes. In the context of this scheme the term *Master-Controlled* may be a misnomer since there is no intention to correct the slave. As a synchronisation message requires the mobile node to be receiving at the time of the clock message, mobile-node power is consumed. If no synchronisation messages are sent, no mobile-node power is required to receive. A master synchronisation algorithm implemented in the form of a statistical analysis of receive packet timing, estimating mobile node clock drift, transmit latency and relative node time could supply sufficient accuracy to perform data fusion. If mobile nodes are simply scheduled to transmit at a particular instant relative to their internal clock, these packets will be received at intervals statistically distributed around the mobile-node's internal interval.

As for other examples, *propagation-time* is not a factor. *Send-time* depends on the implementation but generally consists of a fixed deterministic period with some smaller probabilistic factor. Similarly *receive-time* would include some deterministic and probabilistic components. The major probabilistic component is the *access-time*. A CSMA mechanism does not guarantee access to the medium and therefore the individual node *access-time*, particularly during network congestion, cannot be determined. Despite this, the individual node's internal timing will be apparent by the statistically clustering of received message intervals. Where many nodes are operating together, periods of congestion will be apparent, as well as the relative timing of mobile-node events, with respect to events recorded by other mobile nodes. This scheme reduces the power-load of the mobile node but loses the absolute timing information. This may be estimated from knowledge of the network or by network timing using timing messages transmitted between the static nodes.

2-28
The above WiFi Master-Controlled technique is a proposal, it has not been modelled or analysed in depth. It would appear to be a solution that minimises network topology complexity and minimises node power consumption.

2.6.4 Data Fusion Protocol

For data fusion, simple agreed rules are required to manage latency issues when recombining data from the WSN nodes.

In a TDMA network, data samples can be aligned with either,

- the individual node's time slot or,
- the master time slot.

In both cases, the data can be recombined taking into account the particular implementation. If event timing is required then the event of interest would be accompanied by a time offset value.

Using an RBS scheme, data would be packaged along with the time offset from the reference broadcast. If an RBS clock signal is encoded with an identifier, the offset is encoded with the same identifier. In this way, the data is correctly time-associated with the relevant RBS event.

The use of an accurate master clock source, as in Fig. 2.7, allows the fusion of data from the WSN with data from other sources such as video.

2.7 Reliable Data Transfer

The required level of the reliability of transport of data across a WSN is dependent on the function of the sensor network. In some networks, there is an "inherent redundancy in sensor data" [54], such that errors in data, loss of data or even loss of a node, are of no serious consequence. In the case of monitoring a structure or some similar requirement, the data from all sensors represents a totality. The integration of the total data means the loss of data from an individual sensor either permanently or occasionally can be managed.
A WSN for monitoring teams of athletes requires sensors monitoring individuals involved in a variety of interactions. The interest is in both the individual and the team interaction. Because of this, real-time or near real-time wireless monitoring must occur in such a way that the information is transferred reliably.

Given that monitoring teams of athletes is constrained by the combined power and bandwidth limitations, particularly that of the mobile nodes, the overhead available to ensure reliability is limited. In this thesis the primary technique proposed for ensuring reliable data transfer is the use of redundancy of geographic coverage as depicted in various figures such as Fig. 2.5, Fig. 2.6, Fig. 2.13 and Fig. 2.15. Although these figures use four static nodes, this is only for simplicity of the figure. The required number of bases is determined by the sport and the required coverage area.

Since some team sports, such as rugby union, involve many athletes in close proximity, it is probable that geographic redundancy is insufficient since the massed athletes will completely mask the signal at some time. Complex routing via surrounding mobile nodes may reduce the lost packets but an athlete lying on the device will block the signal completely. Data recovery via time sequence redundancy or requested resends of data are possible although both have associated costs.

2.7.1 Reliable Data Transfer - Forward Error Correction

The macro architecture outlined in Section 2.7 will assist in maximising data capture. To complement this, improvements in the robustness of the actual data transfers at the link layer are required. Various techniques are proposed to deal with background and burst errors. Examples of these techniques include: *TCP-Cognizant Adaptive Forward Error Correction in Wireless Networks* [55] which proposes adaptive Forward Error Correction (FEC) as a technique to adapt the level of FEC complexity and redundancy to match the current error rates. As an alternative scheme, *Improving Wireless Link Throughput via Interleaved FEC* [56] uses interleaving to overcome the effects of burst error. The authors implemented this scheme on both Bluetooth and TCP over wireless and claim superior performance to other link-layer schemes. *Custom Coding, Adaptive Rate Control and Distributed Detection for Bluetooth* [57] addresses several aspects of the problems involved in team sport monitoring, in particular *Distributed*
Detection addresses the adaptation of the Bluetooth protocol to manage geographically distributed receivers. While the above papers primarily investigate techniques suitable for general data protection in wireless networks, Forward Error Correction in Sensor Networks [58] specifically addresses sensor networks and their associated limitations. This paper recognises the processing burden of complex FEC schemes and investigates the implementation of low cost byte and block mode FEC schemes. These were implemented and the data transferred using an RF modem component that incorporated binary synchronous data transfer capability.

To complement the results from the above paper, Section 2.10.3 simulates the effectiveness of simple FEC encoding to protect small packets of data transferred via an RF modem using byte oriented transfer through a UART.

Forward Error Correction Implementation

The adaptive FEC schemes identified above and those described in other papers require the system to generate feedback on the current error rate. As noted in different places in this chapter, the ability to send data to a mobile node is dependent on the available power and bandwidth and on the network topology. To minimise mobile node power consumption the nodes are either powered down or operate in a power save mode at the completion of transmission. If it is necessary to interact with the node to change the adaptive scheme, by the time contact is re-established, the conditions that existed at the time of the original broadcast may no longer exist. Rather than a receiving node attempt an immediate ARQ (Automatic Repeat reQuest), the network can be utilised to manage lost packets in a global configuration.

Data Reliability using Commercial Protocols

Implementing a sensor network over the top of a commercially developed wireless network protocol such as WiFi or Bluetooth can shift some of the network management to the Large Scale Integration (LSI) component. At an application level the reliability problem still exists. If nodes are momentarily lost to the network, the
data must be recovered. If a reliable, connection oriented protocol is utilised\textsuperscript{15}, the wireless network may attempt to recover the data on its own. This could lead to higher power costs as the wireless network components repeatedly attempt to transfer the data. The TCP-Cognizant Adaptive FEC scheme attempts to address this by managing the way a TCP protocol stack responds to repeatedly errored data. Alternatively, an unreliable, connectionless protocol\textsuperscript{16} may be more appropriate. In this situation the datagram would be loaded with current and past data, if a datagram or even a short sequence of datagrams are lost, the missing data can be reconstructed from the redundant data in a successfully received packet. In the case of 802.11b/g protocols, the use of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) necessitates the minimisation of network traffic. High network traffic due to ARQ and subsequent repeat messages quickly degrades throughput.

Rather than rely on point-to-point ARQ for successful transfer, geographic diversity (Fig. 2.11) is the preferred method.

\textbf{2.7.2 Reliable Data Transfer - WiFi}

In a WiFi system where the mobile nodes are operating in an 802.11 Power Management - Power Save Mode, if the mobile node initiates dialog directly with a static node (known as Access Points (AP)), the AP will indicate via a 'More Data' flag in the 802.11 header that the AP wishes to forward data to the mobile node. Alternatively, while in the power-save mode, the mobile node listens to the beacons of the AP. The AP can indicate, via a beacon, that data is waiting for the mobile node. Both these implementations require the mobile node to engage in some kind of dialog with the AP. These dialogs need to be minimised to conserve power but this functionality can be exploited as described below.

\textsuperscript{15} Using the TCP/IP protocol as a model, TCP is a reliable, connection oriented protocol, if receive errors are detected the network will request retransmission of the data until the data is transferred correctly, hence "reliable".

\textsuperscript{16} The User Datagram Protocol (UDP), which like TCP sits on top of the Internet Protocol, is an unreliable, connectionless protocol, the sender receives neither positive nor negative acknowledgement from the receiver. Using UDP allows the application developer to decide how to handle errors.
In a point-to-point operation, the transmitting node first checks if the receiving node is available via a Ready-to-Send (RTS) - Clear-to-Send (CTS) exchange. The transmitting node sends the data packet and waits for an acknowledgement (ACK). If no ACK is received within a predetermined time, the sending station retransmits the data. Using this inbuilt mechanism, data reliability can be achieved, but with a high power penalty since the mobile node first has to find a connectable static node.

The point-to-point technique requires multiple transmissions and attempts by the mobile node to contact a static node. To reduce power usage, the WiFi multi-cast and broadcast modes can be used to transmit data concurrently to all static nodes in the service set (SS). In this case, there is no RTS/CTS transaction and no ACK or associated retransmission. All WiFi MAC frames include a Frame Check Sequence (FCS). In normal operation, failure of received data to correctly match the FCS causes the data to be discarded. If transmitted data is FEC encoded (Fig. 2.10), operating the static nodes in test mode enables the node to ignore the MAC layer FCS error and pass the data packet for FEC decoding. If sensor network header data is detected in the data packet then it is decoded.

A combination of geographic diversity, FEC encoding and time sequence redundancy can be utilised to maximise the probability of data capture from mobile nodes (Fig. 2.11). The bandwidth of the WiFi network, even operating at 1Mbps or 2Mbps is such that a considerable volume of time sequenced data will fit into a small packet.
In the event that data packets are lost entirely, the system of Fig. 2.11 can use static nodes in the approximate location of the mobile node, as determined from the current RSSI, to advise the mobile node that a message is waiting. Then, from time to time, the mobile node engages a static node in conversation and identifies and fulfils the request for replacement data. This system combines a high level of redundancy in the high power static nodes with a minimised requirement for resend requests loading the mobile nodes.

The above technique is using only a very limited subset of WiFi functionality. Similar functionality could be obtained using some of the newer components such as those used in [58]. The major benefits of WiFi are the high bandwidth, the low cost per bit and the simplified management of the physical layer (reducing processing load).

### 2.7.3 Reliable Data Transfer - Bluetooth

Bluetooth uses a TDMA/FHSS system and implements both connectionless and connection oriented protocols[20]. Here the terminology appears reversed to normal usage with the links designated:

- Synchronous Connection-Oriented (SCO) link
- Asynchronous Connection-Less (ACL) link
With the SCO link, it is considered that a point-to-point connection exists between two Bluetooth devices and there is a symmetric bi-directional link between the Bluetooth devices. The datagrams or Bluetooth PDUs (listed by type in Table 2.6) provide either bit repetition, modified Hamming or no FEC coding. Modified Hamming with ARQ is also available. The SCO link is provided primarily for voice transmission at 64kbps. A master can manage three simultaneous SCO links in a variety of arrangements. A slave can manage two simultaneous SCO links including connecting to two masters simultaneously.

The ACL link is *point to point or point to multipoint* and used to transfer data within a piconet between the master and slave devices. Referring to Table 2.7, all the ACL PDUs except AUX1 implement error detection; the receiver will automatically request a re-transmission in the event of data corruption.

Implementing the data transfer in a Bluetooth environment using the ACL AUX1 PDU for data transfer, removes the automatic ARQ but requires the implementation of an
FEC or ARQ system at a higher level. Up to 29 bytes are available in the AUX1 PDU so bit, character and block coding techniques are all available. Valenti and Robert [57] used the AUX1 to measure the effectiveness variety of Reed-Solomon codes in comparison to the native Bluetooth FEC codes, over a variety of conditions. To further improve reliability they used a modified Bluetooth protocol to simultaneously transmit data packets to two nodes with the network designed such that any successful reception was acknowledged by only a single ACK. In their research, the mobile node acted as a piconet master with the base station nodes piconet slaves. This concept would be extendable to a network that utilises overlapping static nodes for geographic coverage. This concept is limited in the team sport scenario as slave nodes cannot handle the multiple master requirements.

The complexity of the Bluetooth protocols and the TDMA/FHSS technology prevents the use of a straightforward multicast/broadcast as proposed earlier. Simple geographic redundancy is not available and each mobile node must associate itself with a static node at some time (Fig. 2.15 (b)). This arrangement means that effective data transfer must occur via a point-to-point connection. This suggests that the preferred method would be for the mobile node to actively manage the transfer by buffering, transferring and confirming receipt of data whenever it makes an association with a static node. If the transfer is left to the Bluetooth protocol to manage via an ACL link using a medium rate FEC and CRC packet type (Table 2.7), packet sizes of 17, 121 and 224 bytes are available. For packets of 121 to 224 bytes in size, transmission requires five timeslots or approximately 1.9ms. Transferring data from the Bluetooth module to the sensor node processor, and then decoding an FEC message, may require more time and resources than allowing the Bluetooth module to manage the entire transaction.

The inherent functionality of Bluetooth indicates that the burden of ensuring a successful data transfer resides with the mobile node.

Any topology that routes via intermediate nodes requires store-and-forward functionality in the intermediate nodes. The store & forward functionality must decode and recode the FEC encoded packets to ensure that transfer of data via a sequence of links does not compound bit errors to the extent that data cannot be recovered.
2.7.4 Reliable Data Transfer - Generic Hardware Implementation

Implementations using general purpose MCUs and RF modem chips (Fig. 2.12 (a) & (b)), usually rely on a serial data link implemented with a Universal Asynchronous Receiver Transmitter (UART) (Fig. 2.12 (a)). In this case, data uses an asynchronous protocol and the FEC cannot protect against corruption of the UART synchronisation bits or the start and stop bits. In the event these bits are corrupted the UART loses synchronisation and an entire message may be lost. Over-sampling is one technique to reduce this form of error; the digital output of the RF receiver is oversampled using an MCU digital input port (Fig. 2.12 (b)). In effect, this changes the incoming data to a bit repetition coded signal. The processing necessary to perform this operation may require the MCU to run at a higher frequency and hence a higher power.

To improve data throughput, data links can operate asymmetrically with data from the sensor node to the host transmitted as synchronous data and data from the host to the sensor node sent as asynchronous data (Fig. 2.12 (c)). This has the benefit that the sensor to host, high volume link can reduce the overheads and the probability of error. Similar to asynchronous data, synchronous data is also susceptible to unrecoverable corruption of synchronising bits. Measures of these losses are discussed in [58]. The host is not as power or space limited as the sensor nodes and can therefore afford the extra complexity and power required to extract the data. In the host to sensor direction, synonymous with master-to-slave in the synchronisation discussion, the key message is the network synchronisation pulse. Simulations of UART decoding of raw data and FEC encoded data are included in section 2.10.3.

The primary protection against data loss in a generic network would rely on similar techniques to the WiFi proposal. The sports venue would include geographically distributed static nodes, spaced to maximise the probability of at least one node correctly capturing the data stream. Mobile nodes broadcast FEC encoded data with the data incorporating time sequenced redundant data. Due to the low bandwidth this scheme cannot incorporate as much redundant data as for WiFi. Requests for lost packets could be encoded in clock synchronisation messages.
2.8 Network Start-Up Protocols

Starting a WSN has several power costs; the burden of these costs should be placed on the nodes most able to bear them i.e. the host system or static repeaters. A WSN used for monitoring teams of athletes requires starting several hours prior to the sporting event to allow nodes to be distributed and affixed to athletes in dressing rooms. This may complicate the start-up and continued synchronism of the network, particularly when the devices are out of RF contact. The type of hardware also affects the start-up sequence. For example, if the mobile nodes are not equipped with a sniff mode or RSSI facility then they cannot easily detect if and when other nodes are broadcasting.

Any operating system developed to manage a sensor device must have the ability to manage these scenarios.

2.8.1 Start-up Scenario: WiFi Based WSN

This WSN WiFi protocol puts the major power load during start-up on the network host. WiFi devices are equipped with RSSI and can operate in the CSMA/CD mode. A WiFi based node upon power-up, performs a multicast broadcast looking for the network host using a pre-programmed Service Set IDentifier (SSID). The network host, if available, detects this broadcast and replies with some timing information including the current time and the time to begin monitoring. Network synchronisation messages assist mobile nodes in adjusting their local clocks. If the network host is not
available, the node sleeps for a time before trying again. These power-up transactions are sufficient to confirm the devices are operational prior to deployment.

At monitoring start time, the device again wakes and broadcasts, looking for a host node for synchronisation information. Once the host is identified and resynchronisation complete, the mobile node begins transmitting as the WSN protocol requires. Between data transmission and synchronisation monitoring the wireless module is sleeping, on-time is limited to only a few milliseconds-per-second (duty cycle 0.2%).

### 2.8.2 Start-up Scenario: Bluetooth Based WSN

Bluetooth has its own start up protocols, mobile PANs should wake and look for an appropriate piconet master, using a pre-programmed list of master devices. Similar to the WiFi, once the initial connection is established, the mobile nodes can shutdown until required.

Upon reawakening, the mobile nodes begin sensing and begin looking for an appropriate piconet master to link to. As soon as data transmission is completed, the node moves the wireless module to a lower power state.

### 2.8.3 Start-up Scenario: Generic Hardware Based WSN

Using the RF components from Table 2.1 in the arrangements (a) or (b) of Fig. 2.12, the lack of RSSI poses some problems. Mobile nodes cannot detect the presence of carrier and therefore can only be sure of data transmission when data from the RF modem indicates a valid message.

The generic solution is limited to relatively low bit rate devices, even short messages require substantial on-air time and subsequent power drain. Different approaches are possible with two scenarios below.

(1) The mobile node is turned on and simply begins broadcasting an "I'm here" message. The mobile node then switches to receive mode and listens for an acknowledgement. If the host detected the message it returns an acknowledgement addressed to the mobile node and includes timing information. If the node doesn't get a
response it waits a few seconds and retries. This start-up protocol will work in the situation where the network user turns on the host and then powers on each mobile node in turn. This is similar to the other technologies but without the security of knowing there are no data collisions.

(2) The host node turns on and begins transmitting in the master timeslot and other evenly spaced timeslots. The transmissions in the other slots include the timeslot number. A mobile node is powered on and immediately moves to receive mode. Within a short period, the mobile node will have captured a message indicating the location of the master timeslot. The mobile node synchronises to the master timeslot and begins transmitting in its own timeslot. The master node sequentially works through the mobile node timeslots and listens to all vacant timeslots to detect when a mobile node comes on line. The mobile nodes recognise the transmission sequence and therefore know when their slot is available. As more nodes come online, newer nodes quickly detect the timing information either from other nodes or the master node.

The timing signal must include control information such as, current-time, sampling start time, beacon channel number etc. The requirement for nodes to be away from the timing source is more of a problem with the generic solution. If the mobile nodes include MCXO or TCXO and post-facto adjustment, an initial synchronisation, in conjunction with a wider receiver window, may be sufficient to recapture synchronisation messages when the network begins monitor operations. E.g., relative drift is controlled to within ±5 ppm. If the mobile node is separated from the master node for an hour, the relative drift is 18 milliseconds. The mobile node initially should begin monitoring for the TDMA master 20ms early and continue monitoring 20ms beyond normal. On a multi-channel system, this small extra power cost could be overcome by using a beacon channel, constantly broadcasting synchronisation information.

2.9 Implementation Scenarios

Combining the discussions from the previous sections it is seen that each of the technologies investigated has particular requirements to enable implementation. The scenarios outlined below attempt to combine these requirements into a system capable of monitoring teams of athletes engaged in a sporting activity.
2.9.1 Implementation: WiFi - 802.11b/g

In this section, the word *station* is used to identify the WiFi device. This is in line with the terminology of the 802.11 standard. Mobile stations are athlete mounted and static stations are those mounted around the sporting venue to provide geographic coverage.

From Table 2.1, the Wireless-G stations transfer data at a power-per-bit cost that is at least an order of magnitude less than for other devices. This very low bit-cost is offset by the high current drain, a level of current that would require either a supply with an ultra-low internal resistance or a much larger battery than is necessary to supply the average load. This problem could be overcome using a combined battery/super-capacitor arrangement if the transmitter or receiver on-time can be minimised. If the 54Mbps data rate cannot be maintained due to interference, the data rate may fall as low as 2Mbps and the cost per bit rises to approximately 300nW/bit (from 11nW/bit).

802.11 protocols identify the Physical (PHY) and Media Access Controller (MAC) protocol layers. Some hardware devices completely incorporate these protocol layers, simplifying the development of a WSN. WiFi implements Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), this necessitates minimising network traffic since high network traffic degrades throughput. Note that at 54Mbps, 125 bytes of data take less than 18.5 microseconds to transmit and at 11Mbps, 90.9 microseconds. Fifty units transmitting 125 bytes (including network overhead) at 54, 11 and 2Mbps take 925, 4545 and 25000 microseconds respectively. Though these values appear encouraging, they are unrealistic given the transmitter/receiver hardware synchronisation requirements; the physical layer RF-sync preamble and header are transmitted at low data rates and require fixed timing.

---

17 The Philips BGW211 Low-Power WLAN SiP (System-in-a-Package) is such a device, incorporating RF Transceiver, Baseband/MAC controller and implementing the PHY and MAC protocols. The BGW211 measures 10x15x1.3mm and has connections for two antennas, crystal and serial input/output to the host system. For the 802.11b/802.11g modes, receive power is 300/400mW, transmit power is 550/600mW (+15dBm to the antenna port) [27].
Table 2.8 Preamble timing in 802.11b/g Physical Layer for FHSS and DSSS modes.

<table>
<thead>
<tr>
<th>Mode</th>
<th>RF preamble (micro-seconds)</th>
<th>PHY preamble (micro-seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FHSS</td>
<td>96</td>
<td>32</td>
</tr>
<tr>
<td>DSSS</td>
<td>144</td>
<td>48</td>
</tr>
</tbody>
</table>

If the data is transmitted at a lower data rate for reliability (2Mbps), a single packet of data of 125 bytes payload will take approximately 700 microseconds. Fifty units transmitting 125 bytes payload per second equates to 35 milliseconds per second. This is a very low utilisation of the available bandwidth. Using 550mW transmit power and 125 byte payloads; this model results in an amortised power cost of 385nW per payload bit.

To support a topology like that shown in Fig. 2.13, mobile stations must transmit data via a static station back to the host system. This is either by mobile stations pairing up with a static station or by all static stations accepting any data transmitted from the mobile station. A simplified network using a loose TDMA scheme schedules the WiFi devices to transmit at approximate time points. If the message is unicast (addressed to one receiver) and the other static bases are operated in promiscuous or monitor mode, they will be able to capture the traffic. Alternatively, as suggested in section 2.7, the message is multicast and all static receivers within RF range and within the network grouping (same SSID) capture and decode the message. Wired Equivalent Privacy (WEP) could be utilised but at the cost of requiring the mobile stations to direct a data packet to each of the potential receivers (static stations). Encryption of data, if required, would be performed at a sensor network level - not using the WiFi supplied functionality.

Multicasting may lead to complications. For example, if the network is operated in a WiFi infrastructure mode, the static stations (operating as Access Points (AP)) will want to forward all multicast messages to all nodes in the service set. This should be suppressed. A further complication occurs from operating the static stations in monitor mode, so as to ignore FCS errors. In this mode they are unable to transmit and therefore cannot forward data to the network host. To overcome this complexity and to
To enhance flexibility, static stations would consist of two WiFi components and a controlling processor, one WiFi station monitoring the mobile stations on one channel, the other relaying mobile data to the network host on a different channel.

![Diagram of WiFi stations operating in a Distributed Coordination Function (DCF) mode using RBS synchronisation. Mobile stations could transmit randomly or on a schedule. Mobile station multicast transmission received by any available static station.]

As the 802.11 protocol only goes up to the MAC layer, the sensor network must provide any higher-level protocols. These would only be rudimentary and look unlike anything that other protocols, such as TCP/IP or IPX/SPX, would expect. This in itself would reduce any casual monitoring or capture of the data but to prevent unauthorised use of the data it would be necessary to encode it.

To minimise network interference, the host system uses the attached station and other static stations to 'book' the channel when mobile stations are scheduled to transmit. This would then provide the Distributed Coordination Function (DCF) where channel access uses both physical and virtual CSMA.
The above descriptions and calculations are simplifications but are indicitative of the expected requirements for data transmission. The 802.11 specification is complex and has various additional timing constraints, message handling requirements etc. The data packets carried by the MAC layer must also include sensor network addressing, control messages, error correction and other data redundancy. The 35ms required for all 50 units to each transfer 125 bytes does not guarantee that sufficient bandwidth is available at a venue. Many venues make WiFi available for patrons and the high number of patrons with Bluetooth enabled mobile phones may impact the available bandwidth\textsuperscript{18}.

Implementing a WSN on top of a WiFi MAC layer still requires a synchronisation scheme for the sensor network. This synchronisation would control both data fusion and a hybrid TDMA-802.11 wireless network. An RBS or similar scheme appears appropriate but the cost of mobile nodes listening for a Reference Broadcast at a scheduled time may be excessive, given the probabilistic nature of CSMA networks. Further investigation of the statistical based master-controlled WiFi synchronisation proposal from the "Low Power Master-Controlled WiFi Synchronisation" discussion in Section 2.6.3 is required.

### 2.9.2 Implementation: Bluetooth

Bluetooth implements a full protocol stack from the physical layer to the application layer. In this case, the application layer consists of communications services, the most common of which are voice services, object transfer and serial communications. The term \textit{PAN} (Personal Area Network) will be used to indicate Bluetooth devices. \textit{Static} will indicate stationary PANs, those that form part of the network infrastructure; \textit{mobile} indicates an athlete mounted PAN. The role of the PAN as either master or slave (or both) will be indicated in the context.

\textsuperscript{18} As Bluetooth and WiFi inhabit the same spectrum, potential exists for interference between technologies. Ennis (1998) reviews the effect of Bluetooth operation on DSSS WiFi operation [59]. As Bluetooth operates on a FHSS mode, across all available Bluetooth channels, a Bluetooth transmission can hop into a channel already in use for a WiFi transmission, corrupting the entire WiFi packet.
Bluetooth is designed for short messages with much lower overhead. Although the longest MTU can be 65535 bytes, the Bluetooth specification identifies the smaller PDU or Protocol Data Unit. The PDU is one, three or five timeslots in length and the MTU can be split across multiple PDUs. The following extract from the Bluetooth specification (Table 2.7) gives the packet types, payloads and effective data rates. The packet types are mostly of the form DXn, where D stands for Data, X can be either M (medium rate) or H (High rate) and n indicates the number of 625 microsecond timeslots utilised (1, 3 or 5). The given data rates would appear to be adequate as system data rates if they can be maintained. Because Bluetooth implements the higher-level protocols, it is not easy to manipulate the system to fit the requirements of a system monitoring team sports. Although small amounts of low speed data can force the system to use small packets, the network management component of data transfer may make less regular larger packets more feasible.

The Bluetooth protocol allows the formation of piconets and combinations of piconets forming into scatternets. The piconet is a grouping of a master PAN and one to seven active slave PANs. Many more PANs can be part of the piconet but these must remain inactive (parked) until moved to the active state by the master. The master or slaves of one piconet can be slaves in another piconet, the combination forming a scatternet.

Due to the protocol limitations of the piconet and the requirements of monitoring team sport athletes on a large sporting field, a complex scatternet is required. Each athlete mounted PAN should be nearly continuously attached to the network. The possible
combinations of inter-PAN proximity requires the development of dynamic routing algorithms appropriate to a Bluetooth network. The basis for a simple routing algorithm requires the first seven PANs approaching a fixed PAN to become piconet slaves of that PAN and at the same time advertise themselves as piconet masters (Fig. 2.15(a)). Any mobile PAN that cannot directly connect to a static PAN connects via the mobile masters. As soon as the mobile masters move away from a fixed master, they stop being masters and instead search for a piconet to join. This technique allows more extensive coverage, especially during massing of athletes, but the power and bandwidth cost would be greater due to necessity of mobile PANS mastering piconets.

An alternative to the above scheme is depicted in Fig. 2.15 (b), static PANs have all mobile PANs and their secret-key\textsuperscript{19} registered in their database. As any mobile PAN approaches a static PAN, they set up a data link and transfer any buffered data. The link can then be dropped until the mobile PAN reaches a data threshold at which point it again attempts to pair with a static master and transfer its data. To minimise set-up/tear-down time, a master could maintain the connection with the slave (either active or parked), until seven connected slaves was reached. At this point, the oldest connection is dropped and the slave holds off before reapplying for a connection to that master. This allows other slaves to contend for a connection. In a network such as this, latency would take on a probabilistic value. This topology would appear to meet the requirements for robust data transfer outlined in Section 2.7.

\textsuperscript{19} For security, Bluetooth implements a Challenge Response security system. Both units in a pair must know a secret key. The secret key is used during the link set up to confirm that the other unit is one that is authorised for communication.
Although the Bluetooth specification allows for masters to connect to seven slaves and PANs can operate in multiple piconets, this is limited by the actual manufacturer implementation. A Bluetooth PAN for use in a headset or as a serial cable replacement will not necessarily have the same flexibility and functionality as a device used in a computer or PDA. The limitations of the devices available for general-purpose use may constrain the implementation of a WSN using Bluetooth as the wireless platform.

As for WiFi, the sensor network on top of the wireless network will still require synchronisation information for data fusion. If the latency of the data packets is unknown, every data packet would need to carry source and destination time stamp information.

As noted in the WiFi section, there is a high probability that major sporting venues will have a number of 2.45GHz devices operating including WiFi Access Points, Wireless enabled laptop computers, Bluetooth enabled phones and PDAs. Although these technologies have been developed to minimise interference, the proximity of these devices could severely inhibit the operation of a 2.45GHz-based athlete monitoring system.
2.9.3 Implementation: Generic Solution using ISM Bands

ISM bands appear in the various regions or are covered by local class licenses. Some of these bands have different restrictions, such as duty cycle or frequency hopping, in different countries. Generally, the devices available for these bands do not currently match the sophistication of WiFi, Bluetooth and other commercial technologies. This may be an advantage since it is possible to develop a wireless network that more appropriately suits the requirements. Some of the limitations of the previously discussed technologies are directly due to their sophistication.

Using a multi-channel device, such as the Nordic nRF903 in the arrangement described in Fig. 2.2 (a), implement a simple TDMA/FDMA network. Because of the low data rate (76.8kbps) a tight TDMA scheme is required to maximise data transfer. Data packets use a format similar to the sync message of Fig. 2.23 in Section 2.10.3.1. With fifty devices and an update rate of once per second, each device has a maximum timeslot width of 20ms. Allowing for a 100 byte message along with RF and data synchronisation, giving a duration of 10ms, the transmit current of 29mA results in an average transmit current of 0.58mA. This compares to an average current of less than 0.1mA for a single WiFi transmission carrying the equivalent data load.

Table 2.9 nRF903 RF Channel Frequencies and Frequency Bands (Table 5 from [30])

<table>
<thead>
<tr>
<th>Channel</th>
<th>Frequency band</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>433.05MHz - 434.79MHz</td>
</tr>
<tr>
<td>0</td>
<td>433.1904 (-10%)</td>
</tr>
<tr>
<td>1</td>
<td>433.3440 (-10%)</td>
</tr>
<tr>
<td>2</td>
<td>433.4976 (-10%)</td>
</tr>
<tr>
<td>3</td>
<td>433.6512 (-10%)</td>
</tr>
<tr>
<td>4</td>
<td>433.8048 (-10%)</td>
</tr>
<tr>
<td>5</td>
<td>433.9584 (-10%)</td>
</tr>
<tr>
<td>6</td>
<td>434.1120 (-10%)</td>
</tr>
<tr>
<td>7</td>
<td>434.2656 (-10%)</td>
</tr>
<tr>
<td>8</td>
<td>434.4192 (-10%)</td>
</tr>
<tr>
<td>9</td>
<td>434.5728 (-10%)</td>
</tr>
<tr>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>-</td>
</tr>
<tr>
<td>...</td>
<td>-</td>
</tr>
<tr>
<td>168</td>
<td>-</td>
</tr>
</tbody>
</table>

Use must be according to ETSI- and FCC frequency regulations. Table 2.9 lists the available channels for the three different frequency bands. Legal channels are shown shaded; maximum allowed transmission duty cycle is shown in parenthesis for the European LPRD/ISM-frequency bands.
**Example Wireless Network Design Using Generic Hardware.**

This design considers the operation of a simple network using an RF baseband device and an MCU incorporating a serial UART, the node discovery and node configuration phases are not discussed.

1. The device does not have RSSI capability.
2. The device has multi-channel capacity and 5 channels are identified as available.
3. The allowed duty cycle is <1%
4. The device's raw data rate is 64kbps.
5. The application requires 10 updates per second from every sensor.
6. A clock sync message is delivered every second.
7. The maximum receiver and transmitter start up delay is 1ms.
8. Some data loss is acceptable.
9. High precision crystals are used and a 2ms slot separation is considered maintainable.
10. RF receiver synchronisation requires 16 bits, UART synchronisation requires 11 bits and data synchronisation requires three bytes.

In this scenario, using a TDMA/FHSS system, the maximum time slot size is constrained by the duty cycle to be 10ms (640 bits). With five available channels, the update rate for any node can only reach five packets per second unless the time-slot size is reduced to 5ms (320 bits). The master station operates predominately as a receiver and receiver start-up delay is of no consequence. The 1ms transmitter start-up delay is not incorporated in the 2ms time slot separation therefore bringing the total unavailable time during one time slot to 3ms leaving 2ms for data. This equates to a maximum packet size of 128 bits. After allowance for the RF receiver and UART synchronisation (refer Fig. 2.23), there are 101 bits available for the message header including data synchronisation, the message and any error correction/detection or redundancy. Using a UART there are a minimum of two additional overhead bits per message character (stop bit plus start bit). If the message consisted of four 8-bit bytes using a Hamming (7,4) FEC coding with the UART configured for seven data bits, one stop bit and no parity, the message uses 72 bits. Allow a further three 7-bit bytes for data synchronisation and the total RF+UART+Data Sync+Message takes 126 bits.
The TDM synchronisation message, only being transmitted once per second, has considerably more bits available (448 bits or 640 bits less the time slot separation and Tx start up delay). The larger packet size available for synchronisation allows the use of control messages and redundant synchronisation messages.

If the base consists of a single receiver, there are 198 time-slots available per-second for slave units with each slave using 10 time slots, the system could cater for 19 slave units. Total throughput is 4 bytes per time slot or approximately 800 bytes per second (6400bps). The data channel is less than 10% efficient. Synchronous data transmission from slave to master, as opposed to asynchronous transmission using a UART, could improve this rate marginally. In this case, more computing power or specialised hardware is required at the master station. The biggest improvement in RF channel efficiency is obtained by a reduction in the update rate from 10 per second to 5 per second. This increases message latency since the updates are not as regular but provides substantial increases in payload and ability to improve data redundancy and transfer robustness. After allowance for FEC and UART overheads, the payload could be increased from 4 bytes to 21 bytes increasing channel efficiency to more than 25%. This additional capacity allows useful tradeoffs between additional payload, more complex forward error correction and additional data redundancy. As the most vulnerable phase of the data transfer in a UART based system is the UART synchronisation, the additional capacity may be best utilized by providing a duplicated message from the UART synchronisation onwards.

Depending on which of the above scenarios is implemented, a tracking system's directional antenna has either 2ms or 7ms to detect and identify the direction to a specific node. The tracking receiver requires the RSSI functionality.

Data Fusion timing is generated by either relating data samples directly to the time of transmission (slot number) or to the TDMA master clock pulse (all nodes synchronise sensor sampling to the wireless TDMA timing).

Interference from co-located systems operating on the same channels would seriously affect the operation of such a system.
2.10 Experiments & Simulations.

Various experiments were conducted involving node synchronisation, clock drift, Forward Error Correction and RF transmission using the 433 MHz band. This band was chosen because of the availability of 433 MHz components and because the RF and data transmission and reception could be directly controlled. Since the initial experiments were conducted, the availability of 433 MHz components has changed very little but there has been a huge expansion in the variety and availability of 2.4 GHz components. The synchronisation, clock drift and FEC experiments are relevant in any RF scenario however, due the vagaries of high frequency RF, the 433 MHz experiments are frequency specific.

2.10.1 RF Experiments

A few simple experiments were conducted with off-the-shelf 433MHz components as well as experiments in antenna construction and antenna location. The 433MHz transmission experiments tested the ability of the components to successfully transmit data over various distances.

RF Transmission:

Using Nordic 433 MHz Evaluation Kits [60] connected to the serial port of laptop computers, data was transmitted at different speeds and over different distances. The carrier signal was also monitored using a Protek 2GHz Field Analyser (Table 2.10).
Table 2.10 RF field strength during 433MHz data transmission tests.

<table>
<thead>
<tr>
<th>Distance</th>
<th>Field Strength Measured (est*)</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>2m</td>
<td>-9dBm (-17)</td>
<td>In hallway</td>
</tr>
<tr>
<td>42m</td>
<td>-26dBm (-44)</td>
<td>Along hallway (Tx on ground Rx at 1.5m)</td>
</tr>
<tr>
<td>42m</td>
<td>-19dBm (-44)</td>
<td>In open field (Tx at 1.5m, Rx on ground, line of sight)</td>
</tr>
<tr>
<td>100m</td>
<td>-37dBm (-51)</td>
<td>In open field (Tx at 1.5m, Rx on ground, line of sight)</td>
</tr>
<tr>
<td>150m⁎</td>
<td>-53dBm (-55)</td>
<td>In open field (Tx at 1.5m, Rx on ground, NOT line of sight)</td>
</tr>
<tr>
<td>200m</td>
<td>-47dBm (-57)</td>
<td>In open field (Tx at 1.5m, Rx on ground, line of sight)</td>
</tr>
<tr>
<td>250m</td>
<td>-47dBm (-59)</td>
<td>In open field (Tx at 1.5m, Rx on ground, line of sight)</td>
</tr>
</tbody>
</table>

* Estimated field strength from Equation 2.2 and assuming 10dBm transmit power and approximately 2dB gain from each quarter-wave whip antenna. Either the transmitter was outputting an additional 10dB (unlikely since 10mW is the rated maximum) or the Protek 2GHz Field Analyser was operated incorrectly (most likely). The transmitter was switched off and on and monitored by the analyser to confirm the transmitter was the signal source.

# For this distance an intervening mound of dirt prevented it being line of sight.

**Transmission within a lab:**

Data transmissions within a lab had little success. A spectrum analyser identified the presence of a relatively strong signal source operating in the proximity. Investigation identified the presence of computer with a microprocessor operating at 433MHz.

**Transmission down a hall (42m):**

Data was successfully transmitted down a 42m hallway at data rates of 9600 and 19200 with only one or two errored characters per thousand. Occasionally bursts of errors occurred. These bursts appeared to last around a tenth of a second. This would result in the loss of possibly 40% of characters in a 0.1 second block.

**Transmission in the open:**

Data was transmitted between two sites at speeds of 9600 and 19200 bps. The distances used are given in Table 2.10. Although the field strength measured by the signal analyser appears weakest at 150m, the received data at 150m was a much higher quality than the data captured at 200 and 250 metres. Some examples of the captured data appear in Fig. 2.16. This data was of the form "Line nnnn aabcdefg..." where
"nnnn" was a four-digit line number. The data was transmitted and received on a computer UART port therefore there was no additional UART synchronisation signal and no mechanism for detecting errors occurring in the UART start or stop bits.

RF Antenna Masking.

Using a Radiometrix evaluation kit (comprising two transceiver components with quarter-wave whip antennas) data was transmitted between two units over a distance of 100m. The units were configured to operate in a 'ping' mode with one unit (the master) transmitting sequenced packets then waiting for the other unit (the slave) to echo the sequenced packet. Provided the master was detecting carrier and correctly received the echo packets, an indicator LED remained lit.

---

**19200 bps 150m**

<table>
<thead>
<tr>
<th>Line</th>
<th>1501</th>
<th>abcdefghijklmnopqrstuvwxyz</th>
<th>ìiïe 1502</th>
<th>abcdefghijklmnopqrstuvwxyz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ìnêe</td>
<td>1504</td>
<td>abcdefghijklmnopqrstuvwxyz</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**<More Severe Corruption>**

<table>
<thead>
<tr>
<th>Line</th>
<th>1506</th>
<th>abcdefghijklmnopqrstuvwxyz</th>
</tr>
</thead>
</table>

**9600bps 200m**

<table>
<thead>
<tr>
<th>Line</th>
<th>1507</th>
<th>abcdefghijklmnopqrstuvwxyz</th>
</tr>
</thead>
</table>

**19200bps 200m**

<table>
<thead>
<tr>
<th>Line</th>
<th>1508</th>
<th>abcdefghijklmnopqrstuvwxyz</th>
</tr>
</thead>
</table>

| Ìnêe| 1509 | abcdefghijklmnopqrstuvwxyz  |

---

Fig. 2.16 Example data from receiver capture files for 150 and 200m separation.
At a transmission rate of 64kbps, with one unit attached to a human body, the master LED remained lit at distances of 100m, both line of sight and behind an earth mound. Transmissions continued to be correctly received at 100m non-line of sight and with the unit on the body facing away from the other unit.

In the last situation detailed above, packet loss only began occurring when the antenna itself was masked (wrapped in one hand). Packets were still correctly echoed and received but not the complete sequences. All packets were blocked by completely masking the antenna (required two hands and the antenna pressed against the body). Even with this masking, partial packet round trip completion would occur if the masked antenna were faced toward the other unit. The rate of packet reception increased if line-of-sight was restored, even with the antenna masked.

**RF Transmission Conclusions:**
These RF experiments were intended as proof of concept experiments, testing whether the commercial devices could achieve the required data transfer over the required distance. In these cases the evaluation kits were designed and manufactured by experienced RF engineers and were assumed to be designed for optimal operation. In these tests data transfers were effected in various scenarios. In particular, the continued error-free reception and transmission of the Radiometrix unit while one unit was mounted on a human suggested that using these devices for transmission from an athlete was completely feasible.

**433Mhz Antennas:**
A variety of 433MHz antennas were developed and tested. Results from these experiments could be summarised as "Variable". Small PCB loop antennas that appeared to be effective in the open ceased to operate effectively in the presence of a human body. A quarter-wave length of loose wire had similar problems; data could be received from a bench unit but not from a unit mounted on a body. Small wire-wound helical antennas made to specifications given by the chip manufacturers changed their
RF properties over time due to changing physical properties such as slight stretching. Coating the antenna with wax or epoxy to lock the shape also changed the RF properties.

Larger antennas were developed for use as base stations. Because of the changing orientation of athletes and the attached devices, some experiments were conducted with helical antennas. A non-portable prototype antenna was developed with a modest gain of around 5 (7dB). To simplify the network set-up at a sporting venue some prototype collapsible antennas were developed (Fig. 2.17 & Fig. 2.18). These antennas weighed very little and could be stored in a very small space. The antenna of Fig. 2.17 was not an optimal design and had no gain (0dB). The antenna of Fig. 2.18 only operated spasmodically due to a problem with the balun.

![Collapsible Helical Antenna made with PVC pipe](image)

**Fig. 2.17** Collapsible Helical Antenna made with PVC pipe (Ground plane not fitted)

![Collapsible Yagi-Uda Antenna made with PVC pipe](image)

**Fig. 2.18** Collapsible Yagi-Uda Antenna made with PVC pipe.

**Antenna Conclusion:** Use commercial devices if available.

### 2.10.2 Node Synchronisation and Clock Drift.

In this experiment, two nodes using ordinary crystal oscillators and running a real time operating system were synchronised and then synchronisation was maintained using timing messages. This synchronisation was carried out over a wire. The message format was similar to that of Fig. 2.23. The experiment included synchronising both a one-millisecond clock and a ten-millisecond clock to within +/- 1ms. Other similar experiments using nodes running at different internal frequencies and at different
temperatures were conducted with no significant difference to the results presented below. The 10ms system clock was used to reduce processor load when the processor was running at a reduced internal frequency.

The purpose of the experiment was to test the implementation of a simple clock synchronisation technique. This provided synchronisation between nodes either for data fusion or to maintain a wireless TDMA scheme. The results indicate the feasibility of utilising clock error estimate algorithms as a method of reducing the RF receiver on-time.

2.10.2.1 Primary Interval @ 1ms

**Hardware:**
Hitachi H8-3664 Microcontrollers.
9.8304MHz crystal.
System Clock Divider set to divide by 8 (effective crystal frequency = 1.2288MHz)
Serial Communications @ 38400bps
Internal System Software Clock @ 1ms

**Operation:**
*The master* set its own clock to a value of 600000 milliseconds and then sent continuous sync messages for 9 seconds. At 10 seconds (clock = 610000) it began a 60 second cycle of sending sync messages (lasting about 15ms each) (one 15ms burst per 60 seconds).

*The slave* started up and was immediately looking for sync messages. The slave only listened for 20ms every second for 10 seconds or until a clock message was captured. The slave was also configured to listen for a sync message once per minute starting at a clock of 610000ms. For this test, the slave was started within about 7 seconds of the master (before or after) to capture the initial sync cycle.

Since the clock messages were so widely separated in time, the SLAVE began listening 10ms before messages were due and continued listening 10ms after the messages were due. This was to allow for any clock drift between the two units. As the
clock messages were received and decoded, the slave unit printed out its internal clock value before and after updating as in Fig. 2.19 & Fig. 2.20.

<table>
<thead>
<tr>
<th>START: (pre-update value -&gt; post update value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOCK: 16-&gt;608000</td>
</tr>
<tr>
<td>CLOCK: 610003-&gt;610003 (0ms)</td>
</tr>
<tr>
<td>CLOCK: 670007-&gt;670003 (4ms fast)</td>
</tr>
<tr>
<td>CLOCK: 730007-&gt;730003 (4ms fast)</td>
</tr>
<tr>
<td>CLOCK: 790006-&gt;790003 (3ms fast)</td>
</tr>
<tr>
<td>CLOCK: 850007-&gt;850003 (4ms fast)</td>
</tr>
<tr>
<td>CLOCK: 910007-&gt;910003 (4ms fast)</td>
</tr>
</tbody>
</table>

Fig. 2.19 Clock Drift: Slave Clock, slave is approximately 3.75ms/minute fast.

<table>
<thead>
<tr>
<th>START: (pre-update value -&gt; post update value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOCK: 15-&gt;600944</td>
</tr>
<tr>
<td>CLOCK: 610003-&gt;610003 (0ms)</td>
</tr>
<tr>
<td>CLOCK: 669999-&gt;670003 (4ms slow)</td>
</tr>
<tr>
<td>CLOCK: 729999-&gt;730003 (4ms slow)</td>
</tr>
<tr>
<td>CLOCK: 789999-&gt;790003 (4ms slow)</td>
</tr>
<tr>
<td>CLOCK: 850000-&gt;850003 (3ms slow)</td>
</tr>
<tr>
<td>CLOCK: 909999-&gt;910003 (4ms slow)</td>
</tr>
</tbody>
</table>

Fig. 2.20 Clock Drift Roles reversed: Slave is approximately 3.75ms/minute slow.

From the results in Fig. 2.19 and Fig. 2.20 and tracking the clock discrepancy over a long period, the sequence of clock shifts followed the pattern 4-4-4-3-4-4-4-3 or 3.75ms/minute. This result was symmetrical.
2.10.2.2 Primary Interval 10ms

Operation:
In this experiment, the system clock operated at 10ms intervals. Similar to the previous experiment, the master output was a regular sync message. In this case, the message only contained one clock signal. To obtain the alignment of the clock message with a 10ms boundary the serial asynchronous data protocol was adjusted to pack additional bits into the message. An alternative technique is to trigger a timer to start the sync message at the appropriate moment. The results of Fig. 2.21 & Fig. 2.22 were from sync messages sent at ten second intervals. The experiment was repeated using 60 seconds between sync messages.

To measure the clock discrepancy a timer was triggered on each 10ms interval. The value of this timer at receipt of the sync pulse was printed to allow calculation of the drift.

From the results in Fig. 2.21 and Fig. 2.22, it appeared that there was a discrepancy of about +/- 0.22ms/minute. This can probably be accounted for in the additional processing cost of extracting the sub-interval details. Eg: previously the unit designated "64PIN EVB" was faster by about 3.75ms/minute but when operating as the slave in this test, the extra processing cost (0.22ms/minute) reduced the lead to 3.53ms/minute. As the master it should have been about 3.75ms/minute faster but the extra processing done in the slave extended the lead to 3.97ms/minute.

To check this analysis, the interval between clock messages was set to 60 seconds. The average sub-interval value was 4558 or approximately 3.71ms/minute. Allowing for one additional processing block (0.22ms/6), the timing becomes 3.75ms/minute.

Results:
Printout Explanation for Fig. 2.21 and Fig. 2.22 (over page)
CLOCK: 731000->11502
The first number is the clock value immediately prior to adjustment.
The second number is the sub interval immediately prior to adjustment.
(sub-interval runs to 12288 (equivalent to 10ms) then wraps)
64PIN EVB MASTER (identifying which of two devices is the master)

42PIN EVB SLAVE

CLOCK: 731000->11502  \((\text{slave slow by } 10\text{ms} \times (12288-11502)/12288)\)
CLOCK: 732000->11438
CLOCK: 733000->11498
CLOCK: 734000->11438
CLOCK: 735000->11502
CLOCK: 736000->11502
CLOCK: 737000->11434
CLOCK: 738000->11502
CLOCK: 739000->11502
CLOCK: 740000->11438

Note Slave is running slower, average sub-interval value 11475
10\text{ms}*(12288-11475)/12288 = 0.66\text{ms per 10s or } 3.97\text{ms per minute}

Fig. 2.21 Clock Drift CPU running 10ms primary interval.

REVERSED ROLES

64PIN EVB SLAVE

42PIN EVB MASTER

CLOCK: 1->11956
SYNC FAIL
CLOCK: 602001->1833  \((\text{not counted in average})\)
CLOCK: 603001->745
CLOCK: 604001->685  \((\text{slave fast by } 10\text{ms} \times 685/12288)\)
CLOCK: 605001->749
CLOCK: 606001->685
CLOCK: 607001->749
CLOCK: 608001->749
CLOCK: 609001->685
CLOCK: 610001->749
CLOCK: 611001->749
CLOCK: 612001->685

Note Slave is running faster. Average sub interval value 723
10\text{ms} \times 723/12288 = 0.59\text{ms per 10s or } 3.53\text{ms per minute.}

Fig. 2.22 Master / Slave Reversed: Clock Drift CPU running 10ms primary interval.
2.10.2.3 Node Synchronisation and Clock Drift: Conclusion

In these experiments, the relative drift between clocks was consistent over a long period of time (many hours) and in a variety of conditions. The short-term consistency in the relative drift was sufficient to suggest that a clock error algorithm would be functional in maintaining synchronisation in the absence of numerous synchronisation messages. For the hardware used in these tests, it was concluded that it should be possible to run a TDMA scheme with a 1ms inter-slot separation using synchronisation pulses once per second. For data fusion, clock error estimates can provide the required accuracy even with clock pulses minutes apart.

2.10.3 Simulations of Sync Message Reception.

The purpose of this simulation was to investigate the probabilities of a UART receiving an uncorrupted clock synchronisation message during different perturbing conditions. Different message types were generated then the messages were perturbed by bit errors and the perturbed messages decoded by a simulated UART. The messages were sent in triplets, as in Fig. 2.23, to assist in capturing the clock message. For low power operation the receiver was shut down immediately a message was captured.

The messages include:

- A four-byte clock value carried in a packet.
- A four-byte clock value carried in an FEC encoded packet.
- A clock message (no data, but a specifically defined message type)
- An FEC encoded clock message.

Each of these messages was subjected to both a flat Bit Error Rate (BER) and burst errors. The Bit Error Rate (BER) started at 1 in 1000 (0.1%) and increased to 1 in 10 (10%). Burst Errors were generated by increasing the BER immediately after the first bit error occurred. The increased BER could last for up to 10 bits. The burst errors had an error rate of the square root of the background rate. EG: At a background error rate of 1:1000 the first bit error occurred at bit 98. For the next 10 bits (bits 99 to 108), the BER was raised to 1 in 31. Burst errors could occur at multiple times in a message.
Ten thousand triples or 30,000 sync messages were generated and tested at each bit error rate. This process was repeated ten times for each type of message and for both flat and burst error modes. The results from the ten trials were averaged.

The particular messages used in this simulation cover the use of RBS, TDMA master clock pulse and the transfer of information such as current time and activity start time.

2.10.3.1 Message Format

For this simulation the target system was a generic hardware solution using a Nordic nRF403 coupled to an MCU UART. The nRF403, in common with many other transceiver chips requires a sequence of marks and spaces to train the RF demodulator. Since the RF demodulator does not necessarily begin detection on a character boundary, it is necessary to synchronise the UART. Finally the data must carry a header that identifies the packet. Together this covers the Physical and MAC layers. In a TDMA system the Link Layer is inferred from the sequencing of the packets. The destination of the packets was determined by the header type. In the following text, for simplicity, the un-encoded message is simply referred to as ASCII.

The total message length of ASCII message data was 326 bits consisting of three repetitions of a sync message as described in Fig. 2.23. The first message consisted of 122 bits, the subsequent messages required 102 bits each. At 19,200bps, 326 bits equates to just less than 17ms. Allowing for 3ms inter-timeslot separation this message construction allows for 20ms timeslots. For FEC encoded messages, sync messages were about 55% longer (190 + 170 +170 bits for total of 27.5ms @ 19,200bps). The FEC message can be shortened. The trailer was only used for packing and for timing the shutdown of the transmitter oscillator.
Depending on message type, the clock value carried as data may be constant or different across the three packets. If synchronising a real time value, the value in each packet will be different.

For the ASCII data transfer the data was generated as eight data bits, no parity, 1 stop bit (8,N,1). The FEC encoding uses (7,4) Hamming codes therefore each byte of the original data is split into nibbles and the nibble encoded into seven bits. For this format the data was generated and subsequently decoded as 7,N,1 format.

### 2.10.3.2 Processing

Decoding of the message began at a random point within the second byte of the RF synchronisation block. Data was read bit by bit and treated similarly to a UART processing. Counts were kept for a range of error and processing results:

- Framing Error (Missed stop bit).
- Message correct but check-sum error. (Corruption of check-sum)
- Message error but check-sum says OK.
- All 3 sync messages in packet missed.
- Consecutive sync message errors (longest run or corrupted messages)
- Total sync message errors
- No data errors
- Received OK.
A framing error is a serious error in a single stop bit system since the UART relies on the stop-start transition to detect the next asynchronous character transmission. For ASCII messages the count of "no data errors" and "received OK" need not be identical since, depending on the demodulator lock-in, some bit errors near the start are not critical.

Decoding FEC nibbles was slightly more complex than decoding raw data. Although the original nibble can be easily decoded from the FEC byte, there is nothing to identify which nibbles were originally associated. Therefore to synchronise the incoming data stream, the state machine that detects the header, detects a sequence of six bytes that equate to the six nibbles of the original 3-byte header. From this positive identification, the data bytes were correctly reassOCIated.

### 2.10.3.3 Results

Results approach the expected theoretical results (Table 2.11). An ASCII sync message has either 122 or 102 bits. On average, approximately 88 of these bits are critical to the message. These bits comprise the header (3 bytes), the data value (4 bytes) and the check-sum (1 byte), giving 8 bytes or 80 bits. The additional 8 bits are the critical bits in the UART sync block. This is only approximate since it is affected by the random start point in the RF demodulator and errors in non-critical bits.

Although the FEC encoded data has message lengths of 190 and 170 bits, single bit errors that are critical are limited to part of the UART sync block and to all the start and stop bits up to but excluding the trailer. This identifies 32 critical start and stop bits plus approximately 8 critical UART sync bits. Error estimates based on single bit errors in the 40 critical bits were noticeably lower than results from the simulation (Table 2.12). This can be attributed to the occurrence of unrecoverable double bit errors in the FEC data.

This simple FEC mechanism provided a considerable improvement in the ability to capture clock data in the presence of noise (Fig. 2.24 and Fig. 2.26). The ability to capture at least one clock message in a triple (Fig. 2.25) provides a more robust mechanism than transmitting un-encoded data. The ASCII data and check-sum combination exhibited a high probability (greater than 1 in 1000) of accepting an
incorrect clock value (Fig. 2.27) at virtually any BER. This was minimised using the FEC encoding.

Transmitting a shorter TDMA timeslot alignment message instead of a clock value improved on these figures by 20% (more for the middle range of tested BER values). Four distinct FEC encoded alignment messages can be carried in a 20ms slot, further improving the probability of capturing a message. From these simulations and calculations, it appeared acceptable for generic hardware nodes to utilise a UART in lieu of more complex components such as a binary synchronous (BiSync) receiver or signal processor.

Table 2.11 Probability of Error in Critical Data vs. Simulation Results for ASCII Data.

<table>
<thead>
<tr>
<th>BER</th>
<th>P(\text{error})</th>
<th>Estimate %</th>
<th>Result %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 in 1000</td>
<td>0.9157</td>
<td>8.4</td>
<td>8.5</td>
</tr>
<tr>
<td>1 in 666</td>
<td>0.8761</td>
<td>12.4</td>
<td>12.7</td>
</tr>
<tr>
<td>1 in 444</td>
<td>0.8200</td>
<td>18.0</td>
<td>18.1</td>
</tr>
<tr>
<td>1 in 296</td>
<td>0.7424</td>
<td>25.8</td>
<td>25.8</td>
</tr>
<tr>
<td>1 in 197</td>
<td>0.6390</td>
<td>36.1</td>
<td>36.2</td>
</tr>
<tr>
<td>1 in 131</td>
<td>0.5095</td>
<td>49.1</td>
<td>49.4</td>
</tr>
<tr>
<td>1 in 87</td>
<td>0.3616</td>
<td>63.8</td>
<td>64.1</td>
</tr>
<tr>
<td>1 in 58</td>
<td>0.2164</td>
<td>78.4</td>
<td>78.4</td>
</tr>
<tr>
<td>1 in 38</td>
<td>0.0957</td>
<td>90.4</td>
<td>90.5</td>
</tr>
<tr>
<td>1 in 25</td>
<td>0.0275</td>
<td>97.2</td>
<td>97.3</td>
</tr>
<tr>
<td>1 in 16</td>
<td>0.0034</td>
<td>99.7</td>
<td>99.7</td>
</tr>
<tr>
<td>1 in 10</td>
<td>0.0001</td>
<td>100.0</td>
<td>100.0</td>
</tr>
</tbody>
</table>
Table 2.12 Probability of 1-Bit Error in Start & Stop Bits vs. Simulation Results for FEC Data.

<table>
<thead>
<tr>
<th>BER</th>
<th>( P(e_{40\text{bits}}) )</th>
<th>Estimate % (Critical Bits)</th>
<th>Actual % (Message)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 in 1000</td>
<td>0.9608</td>
<td>3.9</td>
<td>4.1</td>
</tr>
<tr>
<td>1 in 666</td>
<td>0.9417</td>
<td>5.8</td>
<td>6.2</td>
</tr>
<tr>
<td>1 in 444</td>
<td>0.9138</td>
<td>8.6</td>
<td>9.0</td>
</tr>
<tr>
<td>1 in 296</td>
<td>0.8734</td>
<td>12.7</td>
<td>13.4</td>
</tr>
<tr>
<td>1 in 197</td>
<td>0.8158</td>
<td>18.4</td>
<td>19.8</td>
</tr>
<tr>
<td>1 in 131</td>
<td>0.7360</td>
<td>26.4</td>
<td>28.8</td>
</tr>
<tr>
<td>1 in 87</td>
<td>0.6297</td>
<td>37.0</td>
<td>40.5</td>
</tr>
<tr>
<td>1 in 58</td>
<td>0.4987</td>
<td>50.1</td>
<td>55.6</td>
</tr>
<tr>
<td>1 in 38</td>
<td>0.3441</td>
<td>65.6</td>
<td>73.2</td>
</tr>
<tr>
<td>1 in 25</td>
<td>0.1954</td>
<td>80.5</td>
<td>88.5</td>
</tr>
<tr>
<td>1 in 16</td>
<td>0.0757</td>
<td>92.4</td>
<td>97.6</td>
</tr>
<tr>
<td>1 in 10</td>
<td>0.0148</td>
<td>98.5</td>
<td>99.9</td>
</tr>
</tbody>
</table>

Fig. 2.24 Comparison of Message Loss Rate for FEC and ASCII Data for Flat and Burst Error Rates
Fig. 2.25 Comparison of Triples Loss Rate for FEC and ASCII Data for Flat and Burst Error Rates

Fig. 2.26 Comparison of Consecutive Message Loss Count for FEC and ASCII data for Flat and Burst Error Rates

Fig. 2.27 Comparison of Accepted Incorrect Message Rate for FEC and ASCII Data for Flat and Burst Error Rates
2.11 Wireless Summary

Although this chapter focused on the problem of a Wireless Sensor Network consisting of low-power mobile nodes in communication within a range from a few metres to hundreds of metres, many of the options discussed can be used in other contexts.

For international operation of an athlete monitoring WSN, the choice of technologies is limited to a few common unlicensed bands and to specific mass produced, and therefore low cost, hardware items. For local or national operation, some additional options may exist, depending on local regulations. Any athlete-mounted device will suffer from environmental constraints including interference from competing RF sources, device size, weight and power, antenna size, orientation and location and the sport-imposed network topology complexity.

For the purpose of investigating potential technological solutions to the WSN problem, three options were tested against various networking requirements. First, a generic solution using a radio-frequency modem chip controlled by a microprocessor, where all protocol layers are managed by the local node's operating system. Second, an LSI device (WiFi) where several protocol layers are managed within the device and third, an LSI device (Bluetooth) where virtually all protocol layers, with the possible exception of the application layer are managed within the device. Each of these potential solutions had benefits and obstacles to the implementation of an athlete-mounted WSN device.

WSNs for athlete monitoring require synchronisation for the purpose of data fusion and may intrinsically require synchronisation for network functioning. The synchronisation functionality can have a high power cost, therefore low-power synchronisation techniques, including slave-controlled and master-controlled algorithms, were investigated. Slave-controlled synchronisation implementation for TDM/FHSS has low complexity, as it is native to the network topology. For the CSMA technology of WiFi, slave-controlled synchronisation is problematic as the timing of synchronisation messages is not deterministic, and therefore considerable power could be consumed monitoring for these messages. An alternate master-controlled technique was proposed (but not analysed).
Reliable data transfer is constrained by the environment and the available bandwidth and power. For any of the proposed technological solutions, the problem of reliable data transfer requires: geographic diversity of receivers, forward error correction and data redundancy. If necessary, a method of re-requesting data may be implementable although this may raise both the network topological complexity and mobile node power consumption.

Any of the technologies reviewed could be used to implement an athlete monitoring WSN but each has its own benefits and costs. WiFi has tremendous bandwidth but high current drain, Bluetooth has moderate current drain and bandwidth but implements restrictive networking. Finally, a generic solution can be designed to meet the requirements of location tracking, synchronisation and data transfer but the bandwidth is severely limited. Further, implementation in an ISM band that is not available worldwide is not desirable. Zigbee chips operating in the 2.45GHz ISM band may be more adaptable than Bluetooth or the proposed generic solutions but as yet this technology hasn't made any market impact.

Regardless of the wireless technology, the WSN requires a sensor platform that can manage both the sensing and the wireless network. This requires a real-time OS that can manage sensing functions and operate in a synchronised network environment. At the wireless networking management minimum, the OS needs to manage the set-up and teardown of Bluetooth links. For use with WiFi, the OS must manage all the link-layer processing and transmission sequencing. In the generic solution, the OS manages the total wireless protocol including the physical layer, MAC layer and any higher layers along with TDMA scheduling, FHSS sequencing etc.

**Technology Summary**

**WiFi**

Even at the lowest data rate, WiFi makes available enormous raw capacity. The 802.11 standards are flexible enough to implement a simple, high coverage system and the CSMA/CA simplifies network start-up. CSMA/CA makes synchronisation slightly problematic since mobile nodes need to wake their receivers to capture incoming
synchronisation messages. Channel booking and virtual CSMA/CA in a loose TDMA arrangement possibly overcomes this problem. Alternatively, the mobile nodes can occasionally engage in clock swapping with static nodes when transferring sensor data. If the high instantaneous current drain of WiFi can be sustained, the very low duty cycle results in the lowest cost per bit. Newer WiFi components incorporate antenna switching allowing access points to utilise antenna space diversity to improve signal capture. Athlete tracking may be feasible using statistical methods but, given the relatively long preamble, a directional antenna based tracking system should have sufficient time to accurately locate and identify the transmitting node.

**Bluetooth**
Bluetooth draws three to four times less current than WiFi but the data rate is far lower. Bluetooth is most power-efficient when it is transferring high volume data point to point. The Bluetooth inbuilt networking functionality limits the ability to easily create a network for operations on a sporting field. A Bluetooth system must constantly be setting up and tearing down some form of routing to get data from mobile nodes to the network host. This additional routing requirement could seriously impact power costs and would substantially raise the payload cost per bit. Because of the TDM-FHSS networking, synchronisation is intrinsic to the technology, however the master-slave networking functionality introduces latencies into the timing. This would require data to be packaged with a local (sensor) clock value or additional timing information added to the data packet with intermediate node receive and send times. While geographic diversity of static nodes is necessary for RF coverage, Bluetooth networking limitations make geographic diversity a necessity, as only a limited number of pair-wise links can be made at any time. For an athlete tracking system, statistical tracking may be the only option as the network topology, the number of transmitters and the frequency hopping restricts the ability to track with a directional antenna.

**Generic Technology**
The benefit of developing a solution from generic components is that the solution can be made to fit the system requirements. Limitations include the low bandwidth of current devices, the subsequent high cost per payload bit and the different license requirements across the different regions. Slave-controlled synchronisation is feasible
and integral if the network is implemented as a TDM or TDM-FHSS system. For devices without RSSI, initial start-up may be costly but alternatives exist. Athlete tracking using directional antenna is easily integrated as devices operate in specific time slots using a globally known hopping sequence - simplifying the identification of a particular node.

2.12 Conclusions

A wireless sensor network designed for monitoring teams of athletes in real time is a new wireless paradigm. A number of conflicting requirements constrain the solution in a number of ways. The requirement for very small size and very small weight limits the available power, which in turn places constraints on the network bandwidth and the network synchronisation mechanism.

A number of different market driven technologies were investigated as suitable solutions. Each particular technology had limitations due to the original technology design purpose. Using a generic solution made with off the shelf microprocessors and RF modems, a network protocol could be developed but with limited bandwidth. Using consumer technology such as Bluetooth or WiFi, the WSN requires these devices to be operated in non-standard modes (and sometimes these modes are not supported by every manufacturer). These devices improve the bandwidth and reduce the power cost per data bit but at the cost of higher instantaneous current.

The implementation of a number of WSN functional requirements were investigated for several of the identified technologies. Each technology required compromises to achieve the functional requirements. While a working WSN could be implemented in any of the technologies, the potential of WiFi would appear to be the greatest provided the short duration high instantaneous current can be supported within the WSN node. This technology has enormous bandwidth, continuing downward price pressure and an expansion of the available hardware including newer smaller lower-powered devices.
2.13 References


[38] Hanna B., 2001, Feasibility Study for Using Directional Antenna for Object Tracking in Confined Areas, CRC for Microtechnology - Griffith University, (internal paper)


2-72


[60] nRF401 Evaluation Kit Product Specification (Rev 1.2), 2000, Nordic VLSI ASA, Oslo, Norway

[61] Radiocommunications (Low Interference Potential Devices) Class License 2000 (as amended), Australian Communications and Media Authority, Canberra, Australia, 2006.


[64] Australian Radiofrequency Spectrum Plan [Draft], 2002, Australian Communications Authority, Canberra, Australia.
Data Acquisition Platform

3 Low Power Embedded Real Time Operating System

3.1 Introduction

A data acquisition platform, designed to monitor elite athletes, must of necessity, be small in size and lightweight, yet capable of extended operational periods. Initial development of such a data acquisition platform use existing technologies to minimise implementations costs, to obtain data and to develop an understanding of the overall complexity of the problem. Where possible, tools developed for early versions of an acquisition platform should be portable and easily adapted to later implementations.

The design of a wirelessly networked data acquisition platform required a system capable of performing sensing, data acquisition, signal processing, communications and wireless network management on a limited power budget. This degree of system complexity required the development of a loosely-coupled, low-power, embedded, Real Time Operating System (RTOS) on the data acquisition platform. The RTOS was loosely coupled to the hardware by providing appropriate hardware abstraction and minimising dependencies on specific hardware. The RTOS minimised its own power consumption and at the same time was designed for application developers in minimising application power use. The system was by nature embedded, since the required small size precludes anything more sophisticated. Finally, the RTOS was real-time since the data acquisition platform monitors real-time events and is required to operate within a data network.

The prototype data acquisition platform developed for athlete monitoring was a modular system comprising an inductive charging power supply unit, the processor/sensor unit and a communications unit. A description of the system has previously been published [3][4]. Aspects of the RTOS have also been published [4]. This system has been used in a variety of sports related monitoring projects such as monitoring elite swimmers [7][5], monitoring rowing [7], athlete gait analysis [7][8][9], and a variety of other uses including estimating athlete energy expenditure [8], assessing limb segment acceleration [11], studying swordsmanship [12], multi-limb motion monitoring [13], boxing suit monitoring [9], and others. Both the
architecture and implementation of this RTOS has been utilised as a teaching tool within Griffith University and used by other organisations.

The core of this chapter is concerned with the implementation of an Embedded, Low Power, Real-Time Operating System, loosely coupled to a low power microcontroller (MCU), the Hitachi H8-3664. Intentionally this MCU, in common with many low cost MCUs, has little or no inbuilt features designed to support more traditional operating systems. The operating system is designed to exploit the power efficiency of the processor.

3.2 Operating Systems.

What is an Operating System? Definitions of an operating system (OS) vary. Finkel [15] opens his paper "What is an Operating System?" with the statement; "In brief, an operating system is the set of programs that control a computer". This brief definition is extended a little over the following sixteen pages. Galli [16] likens it to "... the computer's project manager. It controls, regulates, coordinates, and schedules all the processes and their utilization of the resources." Pinkert & Wear [17] add the useful term "resource utilization maximiser". Generally, an OS provides:

- Process and resource scheduling, including managing resource contention.
- Hardware abstraction (hiding low level hardware programming complexity).
- Interfaces, whether programming or user interfaces.

Resources include hardware such as memory, input-output and storage media, and also include CPU time.

Despite the apparent beliefs of school students and many undergraduates, OSes haven't always existed and do not all come from the one company. Computers of the late 1950s and early 1960s were single task or single purpose systems. Multi-tasking meant that an operator would stop one task running and then load an entirely new program [18], scheduling was performed using a diary. The Compatible Time Sharing System (CTSS) [19], developed at Massachusetts Institute of Technology (MIT) and

---

20 In this paper the abbreviation OS is used for Operating System and OSes used as the plural form. Both OSs and OSes appear in various literature, with OSes favoured in many texts.
first demonstrated on a vacuum tube based IBM 709 computer in 1961, was the forerunner of the Multiplexed Information and Computing Service (MULTICS) system. While the MULTICS user base was never large, the users were significant. The last MULTICS system (belonging to the Canadian Department of National Defence) was retired in October 2000 [20]. The MULTICS specification grew out of the CTSS development work and incorporated many features that are considered core to modern OSes. In the words of MULTICS Project Leader Professor Corbató of MIT [21]:

"Multics became a paradigm for a comprehensive solution to a host of system problems that even today are not fully addressed in many systems. In no particular order, some of the key ideas I think of are: a hierarchal file system, system backup policies, rings and memory protection, symmetric multiprocessing, paging and memory management, dynamic linking, access control, and a full character set."

Systems such as MULTICS, the later spin-off UNIX, and some subsequent developments are well documented. The Open Group publish the 'Single UNIX Specification' [22], which defines all the functionality required for a UNIX branded OS. The principles underlying OSes are widely taught in Universities and technical training schools.

3.2.1 Embedded Environment

Embedded systems are those where the hardware and controlling software are incorporated in a single unit. Usually the operating programs are stored and run from some form of programmable memory. While this memory was often one-time-programmable (OTP) memory or reprogrammable memory in the form of UVEPROM, many embedded devices today utilise FLASH memory technology. FLASH memory allows simple field upgrading of embedded system programs or data. Typical examples of embedded systems include motor vehicle engine management computers, portable electronics such as mp3 players, network routers, wireless hubs, even computer components such as hard disk drives, SCSI cards and ADSL modems.
While there are a small number of major OSes used in the desktop through to the super-computer environments, there are a large number of OSes in the embedded applications environment (over 70 commercial real time embedded kernels exist [32]). Even in this area, as micro-controller functionality increases, equipment developers are beginning to standardise on a smaller set of proprietary and open source OSes. With increasing memory density, small footprint multi-gigabyte memory components are available to embedded system developers. With this much available memory capacity, many embedded systems are starting to incorporate cut-down desktop computer OSes. As modern MCUs can contain considerable internal memory or address considerable external memory, it is becoming common to see larger embedded OSes.

The second CTSS system ran on an IBM 7094 that operated at approximately 0.35MIPS with 32k words of 36-bit memory (for a total 144k bytes). These statistics can easily be exceeded by a MCU in an embedded system.

3.2.2 Low Power Systems

With the advent of low-powered wireless environmental sensors, the power efficiency of the OS is important. Power efficiency extends not just to the power required to perform the required functionality but also to the system memory requirements since the size of the memory also affects power consumption [23]. Concerns over power consumption are not limited to wireless sensors. It is estimated that Information

---

21 Reviewing web pages such as [www.linuxdevices.com](http://www.linuxdevices.com), [www.linux.org](http://www.linux.org), [www.handhelds.org](http://www.handhelds.org), a large number of appliance type devices and embedded computers running a cutdown version of the GNU/Linux operating system can be found. The appliances range from traditional computer related products such as network switches, controller cards, routers and firewalls, to appliances such as audio and video systems, car stereos, cellular phones, GPS navigation systems, Point of Sale (POS) terminals, industrial control systems, security systems for home and commercial use.

22 At the time of the original development of the RTOS discussed in this chapter (2001-2002), the family of low-power embedded processors used in this development (Hitachi H8/300H family) could process at 5MIPS and included devices with 512k bytes of internal program memory and 8k internal RAM. At the time of writing (2005), embedded systems commonly incorporated high levels of processing capacity. As one example, Linksys (http://www.linksys.com), a manufacturer of computer networking products, has wireless routers with 8Mbytes of program memory, 32Mbytes of RAM running embedded GNU/Linux on a 200MHz 32 bit processor with greater than 250 Dhrystone MIPS performance. The memory components are separate to the processor. This processing capacity is similar to a 1996 desktop computer.
Technology (IT) is responsible for 10% of power consumption in the USA [24]. To address the power consumption of military IT in the field, the Information Processing Technology Office (IPTO) of the American Defence Advanced Research Projects Agency (DARPA) began the Power Aware Computing and Communications (PACC) project [25]. This project has funded a number of developments in the wireless-sensor field, including the MIT micro-AMPS project [26] and the Berkley PicoRadio project [27] identified in chapter 2.

As noted, power consumption is a concern of IT manufacturers because the power consumption and heat dissipation is a limiting factor in chip speed. Changes in manufacturing technology and the use of chips that turn functionality on and off as needed result in large reductions in power consumption [28]. As newer low-power or power saving components become available, it is necessary that the OS take advantage of these features. As is always the case, program algorithms can impact the power efficiency of a computer system.

### 3.2.3 Real Time Operating Systems

The requirement for a real-time OS is a function of the requirement to perform actions at specific instants and/or specific intervals and to operate in concert with other similar devices. In this thesis, the requirement is to monitor athlete biomechanical and physiological activity and wirelessly convey that information to a central site for data fusion. The following definitions summarise the essence of real time systems:

"A real time system is one in which time plays an essential role. Typically, one or more physical devices external to the computer generate stimuli, and the computer must react appropriately to them within a fixed amount of time."[29]

"Real-time systems are defined as those systems in which the correctness of the system depends not only on the logical result of the computation, but also on the time in which the results are produced."[30]

"..., real-time behaviour is achieved by dividing the program into a number of processes, each of whose behaviour is predictable and known in advance. These processes are generally short lived and can run to completion in under a
second. When an external event is detected, it is the job of the scheduler to schedule the processes in such a way as that all deadlines are met."[31]

Note the choice of the value of one 'second' in the last quote is probably ill-advised however, for the text's audience, a period of one second may seem extremely short compared to (say) 90 seconds to load an application or 90 minutes to generate a report. The principles of predictability and the ability to schedule to meet deadlines are key components of a real-time system. The deadlines themselves may vary from milliseconds to tens of seconds.

Just as a Storage Area Network (SAN) is developed around the ability to quickly and robustly store and retrieve large volume of data, a real time system, therefore, must be developed around the concept of time. In the development of a real-time kernel Stankovic [32] states:

"The time dimension must be elevated to a central principle of the system and should not be simply an afterthought."

In the same paper, Stankovic identifies the following five dimensions or measures of a Real Time system:

1. **Granularity of the deadline and the laxity of the tasks**
   If the time between the initiating action and the deadline is short, then the deadline is tight. If the deadline isn't tight but the processing takes nearly all the available time, then the laxity is small. It is necessary to measure the granularity (size) of both the deadline and the laxity.

2. **Strictness of deadline**
   a. Hard Real Time: zero value in producing result after deadline
   b. Soft Real-time: reducing value as time exceeds deadline

3. **Reliability**
   Tasks are rated on their criticality; real time systems incorporate a combination of soft and hard real time tasks. Some tasks, if missed, may have catastrophic results. Reliability is a measure of the systems ability to service those critical tasks.
4. Size of system and degree of co-ordination

Real time systems lay along a continuum from the simple, low complexity, one sensor data logger to the highly sophisticated and interconnected real-time control systems of a modern military aircraft or navel ship. As system size and function increases, the architecture of the system requires increasing levels of complexity in managing the resource contention.

5. Environment

The environment defines the system (eg: a simple data logger, an assembly line robot, an air traffic control system). Each system has input, processing and output requirements related to its environment. Some environments are deterministic with all variables known in which case the systems can implement static schedulers and all deadlines can be guaranteed.

These dimensions are used to define the requirements of the RTOS used in the athlete monitoring systems. Further, Tanenbaum & Woodhull [33] provide a measure of the ability of the RTOS to meet its demands using the criteria in Eqn. 3.1.

"A real time system that meets this criteria is said to be schedulable."

\[
\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1
\]

Where 
- \( m \) = number of periodic events \( i \). 
- \( P_i \) = period of event \( i \). 
- \( C_i \) = time to process event \( i \).

Clearly, from this criterion, there must be sufficient processing power available to (a) process all individual events within their own deadline and (b) process all relevant events within the available period.

Traditional personal computer OSes and business computing OSes are not designed around the requirements of a real time environment. The Microsoft NT family of OSes has the ability to schedule 'real time' tasks. All tasks have a priority within the range 0-31, real time tasks have a fixed priority within the range 16-31 while non real-time tasks have variable priority of 0-15 [34]. A real time task will always preempt a non-
real time task or even a real time task of lower priority, but there is no guarantee that a
real time task will begin executing within any specific time limit (p.750 in [31]). As
this family of OSes only runs on a few limited architectures it is not a contender for
any serious real-time functionality.

The Linux OS operates on a very wide range of architectures but suffers from the same
restrictions, that is, it not designed for real-time functionality. Due to its design using a
monolithic kernel, it is less suitable than the Microsoft NT OS that uses a micro-kernel
architecture. Real Time Linux overcomes these problems by inserting an interrupt
abstraction layer [35], giving the real-time processes the ability to transparently
preempt the kernel and allowing the real-time components of the system maximum
response speed.
3.3 Low Power Embedded Real Time Operating System

The preceding section identified that; (1) core OS functionality has existed for many years, (2) development of lower power system and chip technology is occurring and, (3) there is a plethora of real-time OSes available for embedded devices. It further identified the key role of timeliness of results in an RTOS as well as some of the measures of the system controlled by an RTOS. Despite the above, no system was directly suitable for athlete data acquisition and real time wireless networking. As detailed in an earlier footnote, modern embedded systems can exceed the memory capacity and processor performance of desktop computers from less than 10 years earlier in the computer development cycle. Embedded RTOSes are generally highly sophisticated with substantial tool sets and often rely on hardware features to correctly implement security, memory management and resource locks such as semaphores. To convert such an RTOS to meet the requirements would require the removal of considerable functionality, potentially resulting in an unstable system. The developments closest in terms of functionality included wireless sensor networks such as the picoRadio project [27] with the development of the TinyOS [23] apparently paralleling or preceding this development. This, and similar projects targeted networks of geographically static, low data rate sensors using complex routing via a mesh of custom fabricated sensor nodes in close proximity. For athlete data monitoring, hardware is generic, data rates are higher and the sensors nodes are constantly moving relative to each other and relative to the geographic context. This leads to different networking topologies and different synchronisation requirements, which in turn impact on the type of RTOS architecture. Together these factors presented a different paradigm.

As a wireless network ready RTOS was required for athlete monitoring, the skeleton of an RTOS was developed, deployed and utilised in a range of projects (as identified in the introduction). This RTOS incorporated the flexibility necessary for a system where other major components had not been selected or were subject to user choice, namely the wireless components and network protocols. The RTOS and associated utilities attempted to exploit the features available in the selected processor with the aim of minimising the processor power load while meeting the sensing and networking requirements.
This process, and a review of the system, identified some benefits and limitations of the hardware, the software and the combination of both.

This section covers a number of key aspects of the RTOS developed for use in the athlete monitoring projects, particularly those listed below:

- Dimensions of Athlete Monitoring Sensor Nodes
- System Architecture
- Scheduler Design
- Interrupt Handling
- Concurrency, Deadlock and Semaphores
- Use of Dynamic Memory Allocation (malloc)
- Low Power Operation
- Communications State Machines
- Programmers Interface and Device Drivers
- Loose Coupling Between Hardware and RTOS
- Support for Clock Synchronisation
- Support for Wireless Networking

3.3.1 Dimensions of Athlete Monitoring Sensor Nodes

Fundamentally the sensor nodes perform very simple processing. Data is sampled at a fixed rate, some processing occurs and the results are 'transmitted' to another device, which could be a log file in memory. A number of peripheral tasks occur at start-up or when downloading the log file. As this system was, in many ways a new paradigm, some system requirements could only be determined after the system was built and athlete data collected and analysed.

For initial data collection and analysis, the sampling rate for most activities was 100-150 Hz on three channels. Some systems, such as the 'Force Shoe' [7], required sampling at 1000Hz on seven channels. For most activities, data was logged onto an attached 32Mbyte FLASH memory card via an I²C bus. Serial communications speed was hardware dependent. For RF wireless, ISM band components operated between
20kbps and 64kbps. For infra-red (IR) communications, a speed of 38.4 kbps was used. Some typical system events and their associated deadline timing are identified in Table 3.1. In order to use the Hitachi MCU 3.3V modes, the crystal frequency was limited to 10MHz or less. Frequencies of 9.8304, 4.9152MHz and others, divide down evenly for operation at standard asynchronous communications speeds. One strategy for power minimisation was to reduce the operating frequency of the MCU. Because of this, Table 3.2 lists deadlines measured in terms of clock cycles available at different frequencies. These frequencies only include those that can divide down to the selected communications data rate.

Table 3.1 System Events and Deadline Interval Duration.

<table>
<thead>
<tr>
<th>Events</th>
<th>Miscellaneous Factors</th>
<th>Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial Communications Interrupt</td>
<td>Asynchronous 57600bps 8N1</td>
<td>0.1736ms</td>
</tr>
<tr>
<td>Serial Communications Interrupt</td>
<td>Synchronous 38400bps 8bit</td>
<td>0.208ms</td>
</tr>
<tr>
<td>Serial Communications Interrupt</td>
<td>Asynchronous 38400bps 8N1</td>
<td>0.26ms</td>
</tr>
<tr>
<td>Serial Communications Interrupt</td>
<td>Asynchronous 192000bps 8N1</td>
<td>0.5ms</td>
</tr>
<tr>
<td>1000Hz sampling</td>
<td>8 channels</td>
<td>1ms</td>
</tr>
<tr>
<td>TDMA</td>
<td>20ms timeslot including 4ms gap</td>
<td>2ms deadline</td>
</tr>
<tr>
<td>150Hz sampling</td>
<td>3 channels</td>
<td>6.66ms</td>
</tr>
<tr>
<td>100Hz sampling</td>
<td>3 channels</td>
<td>10.0ms</td>
</tr>
<tr>
<td>Data Fusion</td>
<td>See Data Fusion section 2.5.9</td>
<td>40/4ms</td>
</tr>
<tr>
<td>Clock Synchronisation Algorithm</td>
<td>Refer section 3.3.11</td>
<td></td>
</tr>
<tr>
<td>Signal Processing Algorithms</td>
<td>Refer section 5.7</td>
<td></td>
</tr>
<tr>
<td>Compression and Error Correction</td>
<td>Refer Chapter 7.</td>
<td></td>
</tr>
</tbody>
</table>

Note: Greyed line matches line in Table 3.2 and is used in later examples

Table 3.2 Clock Cycles for Deadline and Clock Frequency Combinations

<table>
<thead>
<tr>
<th>Deadline Duration</th>
<th>Internal Clock Frequency (MHz)</th>
<th>Available Clock Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1736ms</td>
<td>7.3728</td>
<td>1279</td>
</tr>
<tr>
<td>0.1736ms</td>
<td>3.6864</td>
<td>639</td>
</tr>
<tr>
<td>0.208ms</td>
<td>9.8304</td>
<td>2044</td>
</tr>
<tr>
<td>0.208ms</td>
<td>4.9152</td>
<td>1022</td>
</tr>
<tr>
<td>0.208ms</td>
<td>2.4576</td>
<td>511</td>
</tr>
<tr>
<td>0.208ms</td>
<td>1.2288</td>
<td>255</td>
</tr>
<tr>
<td>Item</td>
<td>Process</td>
<td>Cycle Count</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>0.266ms</td>
<td>9.8304</td>
<td>2615</td>
</tr>
<tr>
<td>0.266ms</td>
<td>4.9152</td>
<td>1307</td>
</tr>
<tr>
<td>0.266ms</td>
<td>2.4576</td>
<td>653</td>
</tr>
<tr>
<td>0.266ms</td>
<td>1.2288</td>
<td>326</td>
</tr>
<tr>
<td>0.5ms</td>
<td>1.2288</td>
<td>614</td>
</tr>
<tr>
<td>1.0ms</td>
<td>1.2288</td>
<td>1228</td>
</tr>
<tr>
<td>2.0ms</td>
<td>1.2288</td>
<td>2457</td>
</tr>
<tr>
<td>6.66 ms</td>
<td>1.2288</td>
<td>8183</td>
</tr>
</tbody>
</table>

Note: greyed line matched line in Table 3.1 and is used in later examples.

Table 3.3 Approximate Cycle Counts for Common Processes.

<table>
<thead>
<tr>
<th>Item</th>
<th>Process</th>
<th>Cycle Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analog to Digital Conversion (ADC) (from trigger to interrupt)</td>
<td>134</td>
</tr>
<tr>
<td>2</td>
<td>Background Schedule Loop (no jump to background scheduler)</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>Clock-tick Timer Interrupt (no jump to real time scheduler (RTS))</td>
<td>129</td>
</tr>
<tr>
<td>4</td>
<td>Interrupt (no code execution, includes JSR/return and RTI)</td>
<td>105</td>
</tr>
<tr>
<td>5</td>
<td>Interrupts blocked while reading/writing pipes.</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>Jump to RTS and prepare to execute schedule.</td>
<td>42</td>
</tr>
<tr>
<td>7</td>
<td>Kernel space interrupt including unmasking interrupts</td>
<td>25</td>
</tr>
<tr>
<td>8</td>
<td>Respond to ADC interrupt, read data and write to buffer.</td>
<td>150</td>
</tr>
<tr>
<td>9</td>
<td>RTS execute schedule item (does not include item code)</td>
<td>42</td>
</tr>
<tr>
<td>10</td>
<td>RTS inspect schedule item.</td>
<td>132</td>
</tr>
<tr>
<td>11</td>
<td>Serial Communications Interrupt (like item 4 + decide if Tx or Rx)</td>
<td>120</td>
</tr>
<tr>
<td>12</td>
<td>Simple character transmission (read from buffer and transmit)</td>
<td>25</td>
</tr>
<tr>
<td>13</td>
<td>Simple, near atomic, real time task (eg set an output line)</td>
<td>10</td>
</tr>
<tr>
<td>14</td>
<td>Simple subsystem set-up (includes JSR and return).</td>
<td>100</td>
</tr>
</tbody>
</table>

Note: Some detail and acronyms in this table come from later sections of this chapter
3.3.1.1 Granularity of Deadline & Laxity

While deadlines may be definable in terms of time, or clock cycles, the measure of laxity is complex, depending on an understanding of various interactions within a system. The following discussion is based on the interactions of the OS outlined the chapter.

Because serial communications not only have tight deadlines (Table 3.1) but also consist of long running repetitive sequences, the measure of deadlines and laxity during a communications sequence covers many contingencies. The following example and analysis proposes one possible implementation, analyses the serial communications and then combines the serial communications with other tasks to identify the interactions of deadlines and laxity.

Example Outline: A real time system running on a CPU clocked at 1.2288 MHz transmits synchronous data at 38.4 kbps. The serial communications is controlled by the serial interrupt. Other real-time tasks are controlled by a real-time process queue. This queue is only inspected if a task is due to execute and this will occur during communications. The queue has five items to check and is triggered by a repetitive timer based interrupt operating at a period known as the system clock-tick. Processing various tasks consumes system clock cycles and these are listed in Table 3.3. The task scheduled to execute is an Analog to Digital Conversion.

Serial Communications Analysis: Assuming a worst case scenario, where both a serial communications interrupt and a higher priority interrupt occur while interrupt protected critical code is being executed, 170 clock cycles could be consumed in responding to the interrupt (Table 3.3 items 5, 7, 11). Synchronous communications at 38.4 kbps with an internal clock frequency of 1.2288MHz allows 255 clock cycles between interrupts (Table 3.2), leaving 85 clock cycles to perform useful communications processing. While it is possible to perform useful communications tasks in as few as 25 clock cycles (Table 3.3 item 12), more complicated processing, such as handling and decoding incoming data will take more time. The best-case laxity
is 60 clock cycles. Where communication's interrupts are not affected by conflicting interrupts or interrupt masking, the laxity expands to 110 clock cycles.

**Combined Operation Analysis:** During the communications process the real-time scheduler (RTS) attempts to execute. From the clock-tick interrupt (Table 3.3 item 3) to completion of checking the schedule, including the triggering of an analogue to digital conversion, 883 clock cycles are required to perform the necessary processing (Table 3.3 items 3+6+10 (5 times) + 9+13). During this same period, the analogue to digital conversion will be completed and an interrupt triggered, which will take 150 clock cycles to process (Table 3.3 item 8). This will include two periods of 25 clock cycles each, where interrupts are masked. Total clock cycles required for the RTS to complete, 1033 clock cycles. These cycles must be obtained from the latency of the communications process. Using the 110 spare clock cycles per character, the RTS will take 9.4 character transmissions (1.96ms) to complete. No background processing would occur during this period since all available processor cycles have been consumed by interrupt processes. Based on this analysis, system clock ticks could not be programmed for less than 2ms. No penalty accrues if the serial interrupt is blocked since code is still being executed and the earlier analysis showed that there is sufficient laxity for the serial communications to complete before the deadline.

By modifying the example, much higher communications data rates can be obtained. If this system used a clock-tick at 10ms and triggered the serial communications by the last task in the RTS, the scheduler could complete execution and the system can respond to the analogue to digital conversion all within 0.84ms. If it is known in the system design that no other real-time task will occur in the 10ms interval, serial transmission using an interrupt protected polling routine could transmit 150 bytes of synchronous data at 153 kbps.

For each application, the environmental constraints affect the implementation of the system and consequently the relationship between deadlines, laxity and the correct operation. An analysis of the requirements and available resources, such as is given in Table 3.1, Table 3.2, Table 3.3 and the above example, is necessary to confirm that the system is schedulable, or alternatively, to identify the parameters (like crystal frequency or communications speed) necessary to make the system schedulable.
3.3.1.2 Strictness of Deadline

For athlete monitoring there are combinations of hard and soft real time deadlines.

**Data transmission and reception.**

*Synchronous* wireless serial communications, either transmitting or receiving must be a continuous stream of bits therefore data must be supplied or retrieved from the serial communications before the deadline (hard real time). Deadlines for receiving *asynchronous* communications are also hard real time since the receive buffer must be cleared before the next byte arrives. Deadlines for transmitting *asynchronous* data could be considered soft real time although the value of the data would be diminished if any extended delays occur. On a wireless carrier, increasing the inter-character spacing results in an increase in the probability of corruption of critical synchronising bits (such as UART start/stop bits). Multiple delays could result in the transmission process exceeding the allocated time and corrupting data from other nodes.

**Biomechanical Data Sampling**

Biomechanical activity exhibits a low fundamental frequency, of less than 7 Hz, with various biomechanical artefacts at harmonic frequencies (refer Chapter 4). Comparing original data with delayed data indicated that delaying the sampling by as much as 10ms made no visually discernable difference. From the examples in the previous section, it was observed that while operating the processor at low frequency, a scheduler could execute a short process queue, trigger data sampling and read and store the sampled data in less than 1ms. For low frequency biomechanical data, the deadline is not strict although if there is a high sampling rate (1000Hz) and insufficient latency to allow the handling of concurrent tasks, system scheduling is impossible as per Tanenbaum & Woodhull's criterion (Eqn. 3.1).

3.3.1.3 Reliability

Within the context of a wireless sensor network, usually the reliability of the communications functions is more important than the reliability of sensing. The relative importance depends on the environment. In wireless sensor networks monitoring static structures, because of the spatial relationships between sensors, data

3-15
fusion and interpolation can deal with missing or erratic information (Zhong, 2001, analyses this under the term 'Exploiting the Redundancy' in [14]). With athlete monitoring, each athlete is an independent entity within a dynamic context. In either application, unreliability in handling communications functions could impact multiple nodes, causing loss of data from nodes other than the unreliable node.

On this measure, unnecessarily missing a single incoming data character is highly critical since this will destroy the content of an entire packet. Delayed sending of a character will be critical if data is transmitted synchronously but less critical if asynchronous data is transmitted.

In a TDMA scheme, early or late sending of a packet will corrupt the transmission of the preceding or following node. For TDMA systems this raises the synchronisation mechanism to a high level of criticality.

Athlete sensing is low criticality as the usefulness of the data is not dependent on any single sample and, even repetitive actions do not reproduce exactly from cycle to cycle.

Athlete monitoring systems are not control systems and consequently there are no catastrophic consequences from failing to operate reliably. Alternatively, if individual athlete and team activity analysis was incorporated into live commercial television broadcasts, the reliability requirement would increase considerably.

### 3.3.1.4 Size of system and degree of co-ordination

Internally, within a sensor node, the system functionality and complexity is low. Data is sampled on only a few channels at very low rates (compared to some control systems). The data is often homogeneous, e.g., all data is from the same type of sensor, measured by the same subsystem. Data is sequentially passed through signal processing, compression and encoding functions before being stored or transmitted.

As part of a wireless network, this complexity increases with the size and dynamics of the network. A range of complexities can exist:
• A multi-instrumented athlete uses wireless sensors to synchronously log data from different limb segments. One node transmits time-stamp messages, which other nodes receive and use to time-stamp the samples. Logged data can be recombined post-facto based on the time stamping. System size and degree of coordination: LOW.

• Forty instrumented athletes and game officials are moving over a field of play measuring 160 metres by 80 metres. Physiological and kinematic athlete data is being collected wirelessly in real-time for individual and complex game analysis. Size of system and degree of co-ordination: HIGH

In the above scenario, if the RTOS is handling all levels of the wireless network protocols, the system complexity is considerably higher than if a large portion of the network protocol stack was handled by specialist hardware.

For wireless networking, it is necessary that the RTOS can communicate via a noisy medium, using a defined network protocol to send and receive data and to maintain a synchronised network clock.

### 3.3.1.5 Environment

Within a single sensor unit, the environment is non-complex and deterministic. A few sensors are monitored, the sensor data processed and then stored or transmitted. Inputs and outputs are known. Stored data is downloaded in an off-line mode.

With multiple nodes operating as part of a wireless network in a team sport scenario, system inputs still include deterministic sensing functions. Handling of serial communications is a known deterministic function. The overall wireless traffic and subsequent system output is a function of the wireless networking topography and protocols. As low-power constraints affect the wireless components more severely than the sensing components, there is a requirement for synchronisation of data transmission across the wireless network. There is also a requirement to offload complex processing and power intensive wireless functionality to the static components of the network, such as the base stations.
As a result of these requirements, the RTOS of the wireless sensor nodes still exists within a simple deterministic environment. At certain times sensing is performed, data packets created and sent. At other predetermined times, data packets are received and decoded.

### 3.3.1.6 Dimensions: Summary

**Granularity & Laxity:** Analysis of granularity and laxity requires detailed information on the processing costs of various OS and application functions. The example analysis showed that some forms of communications processing in this particular system could cause a processing bottleneck, but that sufficient laxity existed for correct functioning.

**Deadlines:** Sampling deadlines are not strict but failure to meet serial communications deadlines can result in lost data. Failure to maintain timeslot deadlines can result in corruption of data from multiple nodes.

**Reliability:** Required reliability would generally be low - although where commercial interests are involved this might change to high.

**Complexity:** Individual nodes are of low complexity. The total team sport monitoring appears to have a higher complexity (but not of the order of a warship's battle computer).

**Environment:** Environment is predominately a simple deterministic environment.

### 3.3.2 System Architecture

As small MCUs typically use very limited memory and do not support **privilege levels** or **page memory**, there is no attempt to differentiate between **user space** and **kernel space**. Context switching is not used and all processes share the heap and stack space. There is therefore no necessity to provide any specific memory management functionality. While the system described here uses a form of dynamic scheduling, this architecture can be implemented in a static scheduling system\(^{23}\). The dynamic scheduling exists to facilitate system development.

---

\(^{23}\) As an assignment exercise, 3\(^{rd}\) year Griffith University engineering students developed embedded real-time operating systems and applications based on this architecture. The application required periodic and aperiodic input and output functionality. The target system provided 768 bytes of EEPROM. The smallest functioning RTOS used static scheduling and was implemented in 490 bytes of EEPROM.
The core of the architecture contains two process queues in the form of linked lists of structures that point to tasks. These process queues are referred to here as schedulers. The Background Scheduler (BGS) lists various long-term processes that cooperatively share processor time and that can run in the background. The Real Time Scheduler (RTS) lists tasks that must be triggered at specific points in time. Tasks in the RTS are very short lived and usually trigger a separate processing thread. In this context, *thread* does not imply belonging to a *process context* but simply refers to the processing of a particular functional activity such as the wireless transmission of a data packet.

Processes in the BGS may be running, ready-to-run or blocked. Tasks in the RTS are scheduled to run, not scheduled to run, or running (Fig. 3.1). *Periodic* tasks can be either under control of the RTS, or spun off as a separate thread and run independently, depending on the required period. *Aperiodic* tasks are handled by the appropriate Interrupt Service Routine (ISR), they are not added to a process queue although they can add tasks to either scheduler.

![Task States for Co-operative Background and Real-Time Scheduler Tasks](image)

Inter-Process Communications (IPC) is implemented through the use of pipes. Pipes provide the buffering required between the real-time events and background processing tasks. The pipes also manage concurrent access by background and interrupt driven tasks. Since small RTOSes are used for small applications, there is no necessity for the RTOS to track resources such as pipes as the application programmer knows the location of all resources [37]. Pipes are therefore provided as a header file for inclusion by the application programmer. Fig. 3.2 depicts the implementation of a sampling system using pipes. While only one background task is shown, multiple background tasks could exist, where communication between tasks is also by pipes.
External communications are via serial data links and the architecture includes state machines for handling communications tasks. The state machines enhance the ability to reuse communications functions, cutting down on memory requirements.

When triggered, Comms process reads data from pipe and transmits to network.

Fig. 3.2 Example of Inter-Process Communications using Pipes.

As there is no compiler specific support for this architecture, application programmers must follow design guidelines and perform their own checks for correctness of operation. Applications were written in C and a considerable amount of task scheduling is performed by the manipulation of function pointers. Fig. 3.3 shows the scheduler architecture.

Application programming requires the programmer to write short device driver software for the appropriate input/output subsystem. Typically, a programmer is required to write a small number of application specific program functions such as:

- Subsystem initialisation.
- Interrupt Service Routine (ISR).
- Subsystem shutdown.
- Data processing functions.

If specific communications protocols are required these need to be written, however the OS supports communications protocols using a low-overhead communications state machine.
3.3.2.2 System Clock-Tick

The clock-tick provides the basic timing interval of the system. The clock-tick source is determined by the application programmer. The choice of clock-tick interval and source is a function of the system requirements and available system resources. As an example, a system sampling data at a rate of 100Hz and transmitting data once per second could use a clock-tick interval of 10ms. On each clock-tick the RTS would execute a sampling task and every 100 clock-ticks the RTS would trigger data transmission. To obtain or measure intervals of less than 10ms, other counter/timer resources, if available, would be used. A low power environmental sensor may use a
one-second clock-tick. Although the assumption here is that the clock-tick directly drives the RTS, alternate schemes are suggested in section 3.3.11 and related subsections.

### 3.3.3 Scheduler Design

Both the RTS and BGS utilize the same basic code to generate, add, and delete schedule items. The schedulers differ in the method of execution of tasks in their process list. Both schedulers use a singly linked list consisting of structures as shown in the code extract of Fig. 3.4. The value of `nextInterval` determines the point-in-time on which this task will next execute. For periodic tasks, `intervalSize` identifies the period, while `repetitionCount` indicates either a single execution, a finite number of executions or continuous repetitive execution.

Although the above numeric values are not used by the BGS, utilizing the same code minimises program code size. These values also consume RAM but, as these structures are generated using the limited available heap space, using identically sized structures for both RTS and BGS items minimises memory fragmentation.

```c
typedef struct scheduleItem {
    unsigned long nextInterval; // on a match sch_func will be executed
    unsigned long intervalSize; // will be added to nextInterval by scheduler
    void (*sch_func)(void); // the function to be called by this schedule item
    struct scheduleItem *nextItem; // next item in schedule
    unsigned char priority; // for ordering of list (if necessary) 1=highest p
    unsigned char misc; // future use (does not use any additional space)
    int repetitionCount; // how many times to repeat this item
} scheduleItem;
```

![Fig. 3.4 Code fragment defining a schedule item of either scheduler.](image)

When a schedule item is added to a schedule, the ordering is determined by a priority value, this gives the programmer control of the sequencing of tasks in either list.
### 3.3.3.2 Real Time Scheduler Operation

The RTS combines three C functions and two global variables. The global variable, `currentInterval`, holds the current value of the system clock. The amount of time this represents is determined by the period of the system clock tick, which in turn, is determined by the requirements of the system. A weather station may only require a clock tick once per minute while another system may require clock ticks every millisecond. In this particular case, the clock ticks are provided by an internal timer.

The ISR for the timer points to the `incrementInterval` function. The `incrementInterval` function updates the `currentInterval` value and then compares it to the `nextScheduleInterval` value. If these values match then at least one item in the real time schedule is due for execution on this interval, therefore the `executeRealTimeSchedule` function is called. This function iterates through the real time schedule inspecting the `nextInterval` value in each schedule item. On a match, the function referenced by the pointer `*sch_func` is called. If necessary, `repetitionCount` is decremented and `intervalSize` added to `nextInterval`. If the item existed for a single operation or has completed the specified number of repetitions, it is removed from the schedule.

As the iteration through the schedule occurs, the `nextScheduleInterval` is updated. If new schedule items are added, `nextScheduleInterval` is updated (if necessary). Since iteration through the list has a processing cost associated with it, load on the processor, and power supply, is reduced by only executing the real time scheduler when an item is scheduled for execution.

Tasks added to the RTS should be atomic (small indivisible action), or near atomic, to minimise the overall execution time. A typical task is a function containing a single line of code triggering an internal subsystem, such as starting an analog to digital conversion, or setting an MCU output line.
3.3.3.3 Background Scheduler Operation

Two versions of the BGS exist; the first version runs each task in the schedule to completion and then removes it before stepping to the next task. The second version operates as a Round-Robin scheduler where the tasks are designed for cooperative operation. In both cases, the scheduler will execute the list once and then the system enters sleep mode. When the system leaves sleep mode, which occurs on any interrupt, the BGS will re-execute on return from the interrupt. This operation is shown in the code fragment of Fig. 3.5.

```c
while (TRUE) {
    if(executeBGS){ /*may be set to false by clock-tick ISR*/
        executeBackgroundSchedule();
    }
    executeBGS=TRUE;
    sleep();
}
```

Fig. 3.5 Background scheduler execution loop.

As most data in the system is generated by external activities, putting the system to sleep between interrupts is a valid power saving mechanism. Where the background processes generate data for use by other background processes, then the sequencing on the BGS should be done to ensure the maximum processing is performed on an iteration of the BGS.

```c
void compress (void) {
    if ( data_avl(ADCQ) > 6){ /*data available, do processing*/
        /* processing code goes here */

        write(SIOTxQ,data); /*write results to Serial IO queue*/
    }
    return; //yield
}
```

Fig. 3.6 Example Co-operative background process format.
With co-operative background processes, the task will check if needs to perform any processing. If no processing is required the task returns immediately, allowing the BGS to step to the next task. In the example of Fig. 3.6, the task checks if there are more than six items available in the input buffer ADCQ, if not the task returns immediately. If the input data meets the threshold then some processing occurs, in this case data is read from one queue (ADCQ), processed (not shown) and the output written to another queue (SIOTxQ). The amount of processing should be controlled by the programmer.

### 3.3.3.4 Scheduler Discussion & Summary

This RTOS implements two task lists referred to as schedulers. While tasks are on the scheduler lists they are, in a sense, active tasks. This system does not schedule tasks based on an evaluation of their laxity or closeness of the deadline. No dispatcher is used to shift tasks from a blocked queue to the process queue. In this sense, the schedulers are static schedulers since the RTOS does not evaluate or alter priority. An analysis of the schedulers identifies that priorities do exist, and that the programmer is responsible for setting priorities. The background tasks have the lowest priority and can be pre-empted by any real-time process. While the BGS operates on a round-robin scheme, the proportion of background processing cycles available to any particular task is allocated by the programmer. This is done by predetermining the amount of data-processing the task performs before yielding.

Real-time tasks are a combination of periodic and aperiodic tasks and it is up to the programmer how these are handled. If the periodic tasks need to be synchronised with other activities, they would normally be handled by the RTS, while asynchronous periodic tasks could be moved to a separate thread, such as a combination of timer and ISR. Aperiodic tasks are handled by the appropriate ISR and have no dependency on the scheduler. The ordering of tasks on either scheduler is determined by a 'priority' value therefore the programmer can ensure that one task will be initiated before (or after) another.

This implementation has considerable flexibility without the necessity to implement complex scheduling algorithms. The low cost of running the schedulers assists in reducing the overall processing load, releasing processing cycles for necessary tasks.
3.3.4 Interrupt Handling

As identified in section 3.3.1 and the subsequent subsections, interrupt handling is a function of the dimensions of the system, particularly the strictness of the deadlines for serial communications. At lower system clock frequencies there are insufficient clock cycles available to allow a serial communications process to be interrupted or to be blocked by a higher level interrupt. In the case of stacked interrupts, and due to the hardware interrupt priority of the target processor, the serial communications interrupt (SCI3 in Fig. 3.8) will be one of the last to be serviced.

The interrupt policy therefore requires all ISRs, with the exception of the serial communications ISR, to immediately unmask interrupts. This policy is implemented in the RTOS as it differentiates between 'kernel' space ISRs and 'user' space ISRs. All ISRs operate as a two-step process. On an interrupt, an RTOS ISR initially handles the interrupt and then passes control to the user provided ISR. The ISR shown in Fig. 3.7 is the RTOS ISR which is called on an interrupt, in this case the ADC Subsystem Interrupt, the ISR clears the interrupt mask and then passes control to the user defined interrupt handler. To allow multiple uses of the various subsystems, the subsystem ISR uses a function pointer to point to the user defined ISR. In this example, adc_func is a function pointer. When a subsystem is not in use, the system level ISR points to an empty function.

```c
// *********** vector 25 ADI ***********
__interrupt(vect=25) void INT_ADI(void) {
    set_imask_ccr(0);    /* unmask interrupts */
    adc_func();          /* call user defined ISR */
}
```

Fig. 3.7 RTOS ISR for Analog to Digital Subsystem Interrupt.

Although the system wide policy is to keep interrupt masks off, some interrupt masking is deliberately performed to prevent corruption of critical data structures due to concurrent access by multiple processes. This is discussed in the following section.
Interrupt Priority Use
* Reset 0 System restart
* NMI 7
* Trap #0-3 8-11
* Break 12
* Sleep 13
* IRQ #0-3 14-17
* WKP 18 ('Wakeup' interrupts)
* Timer A 19
* Timer W 21 Used for main scheduler
* Timer V 22
* SCI3 23 Serial Comms: Rx buffer full, Rx error,
* IIC 24 Flash Memory & EEPROM Comms
* A/D end 25 A/D conversion end and data processing.

Fig. 3.8 Interrupt Priority for Hitachi H8-3664 processor.

An alternate ISR implementation to that described, would be the use of a First Level Interrupt Handler (FLIH)(p.259 in [17]). With the use of a FLIH, all interrupts are initially handled centrally. The OS can track the current level of interrupts and can assign interrupt priority instead of priority being decided by the hardware (as in Fig. 3.8). This concept has benefits in improving abstraction from a particular hardware and in improving the overall architecture. In this instance this technique has not been implemented as the processing cost, although small, would either reduce or eliminate the already small latency available to some tasks. The additional functionality would also consume some of the available volatile and non-volatile memory resources.

3.3.5 Concurrency, Deadlocks and Semaphores

In a multi-tasking OS, it is possible for multiple tasks to concurrently access a critical variable resulting in unpredictable or undesired behaviour of the system. One of the oldest mechanisms designed to protect critical variables from concurrent access is the use of semaphores. While blocking critical variables from concurrent access can reduce or eliminate the occurrence of one form of undesired behaviour, it can lead to another, in the form of deadlocks. In this situation, groups of processes are permanently blocked due to the inter-relationships of the processes and various protected variables.

This section reviews how these issues are managed within the embedded RTOS.
3.3.5.1 Concurrency

Concurrency occurs where two or more program threads attempt to access the same program variable or resource. Since background tasks in this RTOS operate sequentially, concurrent access can only occur if an ISR accesses the same variables as the interrupted task was currently modifying.

In this architecture, there are only two structures that potentially have this problem, the schedule lists and pipes. With schedule lists, concurrent access can occur if one process is adding or removing a task from a schedule when an interrupt driven process attempts to add a task to the same schedule. Concurrent access to pipes would occur regularly as an interrupt process reads or writes a pipe currently being written or read by another process.

For both the above scenarios the protective measure is the same, the current state of the interrupt mask is read and saved, the interrupt is masked, the critical code executed and the interrupt mask restored to its original state. If the currently running process is interrupted somewhere between the interrupt mask read and the interrupt masking, no problem occurs since the critical code has not yet been entered.

3.3.5.2 Deadlocks

Deadlocks occur when a process holding one resource is waiting on a resource held by another process, which in turn, is waiting on the resource held by the first process. Galli (p.119 in [16]) identifies four prerequisites for deadlocks:

1. "The system provides mutual exclusion to prevent concurrent access to a resource.
2. There is non-preemptive resource allocation.
3. Processes can hold one resource while waiting on another.
4. A cyclical path exists in a resource allocation graph."

While it is possible for a programmer to introduce deadlocks in developing an application, using the points above, the RTOS itself is not subject to deadlocks nor would it subject an application to deadlock conditions.
1. Mutual exclusion exists in this RTOS in the form of interrupt blocking. Interrupt blocking exists to protect short, self-contained code fragments modifying critical variables. These variables have no form of protected access, which would allow them, as resources, to be held by any other process. Interrupt blocking also exists to protect time-critical, low latency tasks from interrupt-generated delays.

2. Some non-preemptive resource access occurs but, as stated above, the resources themselves cannot be held by a process. Any non-preemptive activity is self contained and very short lived.

3. In this RTOS, resources cannot be held by a process. All resources of the system are available to the application programmer to decide how and when the resources are allocated. If an external aperiodic event dictates the pre-empting of a particular resource, that resource can be immediately taken over and used as required. It is assumed that the application programmer makes these decisions in the design of an application.

4. This RTOS, as implemented, does not include a cyclical path within a resource allocation graph.

Based on these criteria, the RTOS as is, is not subject to deadlocks.

3.3.5.3 Semaphores

While some RTOS developers consider semaphores important in some situations [38], semaphores have not been implemented in this RTOS. Small general-purpose microprocessors usually don't incorporate memory management mechanisms or define any privilege levels and therefore there is no hardware support or particular need for semaphores. It is possible to implement semaphore functionality in software such as in the freeRTOS [39], which implements semaphores through the use of C pre-processor macros. In this situation, an implementation would require application design protocols to be observed.
The overall requirements for this system did not indicate any necessity for the existence of semaphores. The use of semaphores could improve the architecture of a general purpose OS and make the OS more precise or controllable however, the additional overheads and rules governing the implementation would make any benefit negligible. Further, implementing semaphores would require a review of the deadlock situation, since the existence of semaphores implies that resources can be held.

### 3.3.6 Dynamic Memory Allocation (malloc)

The compiler supporting the Hitachi H8 processor supports the use of the `malloc` (memory allocation) and `free` (memory release) functions. This implementation of the RTOS architecture takes advantage of the flexibility offered by dynamic memory allocation although, in general, small sensor systems have known finite requirements. In systems with a small amount of memory, injudicious use of malloc may quickly lead to memory fragmentation and a system crash. To minimise the risk of a crash due to a memory allocation failure, variably sized system resources, such as pipes, were created before any schedule items. In this way, if schedule items are added and released during the operation of the system, the allocated areas of memory are the same size as any released areas of memory, preventing memory fragmentation.

As identified in section 3.3.2, the architecture is flexible in implementation and malloc is not strictly necessary. The two main OS services that utilise malloc are the generation of pipes and the generation of schedule items. In a system with known requirements, both these functionalities can be managed using static declarations to obtain the required memory space.

### 3.3.7 Low Power Operation

Wireless sensor networks under development in other applications face similar power restrictions and power related challenges. Chandrakasan et al. [40] identified the use of low energy software, achieved by optimising algorithms to reduce required processor cycles. This has a flow on benefit of reducing the microprocessor clock frequency and therefore the power. This 'low energy software' includes energy aware algorithms; algorithms that scale the processing to meet power requirements or even scale the power supply voltage. Li, Sutton and Rabaey [23], while covering some of the above
points also identify the power cost of RAM and program memory. Where large amounts of memory exist, power requirements increase. Minimising program and scratchpad memory requirements allows the use of hardware with small memory components.

![Diagram of Power Costs](image)

Fig. 3.9 Power cost inputs into system power requirements.

By isolating various components of the power budget (see Fig. 3.9), each component can be optimised with respect to the power requirements. Low power operation is therefore a combination of tactics designed to minimise MCU activity and the power requirements of all peripherals.

These tactics include:

- Select appropriate hardware technology.
- Minimise power use by subsystems.
- Minimise power use by controlling the frequency of the MCU.
- Maximise time MCU in sleep state.
- Algorithm optimisation.
- Low power wireless protocols.

As wireless protocols were discussed in the previous chapter, particularly in respect to power requirements and power conservation, they are not covered in this section.

In essence, all of these tactics are common in the embedded system environment. Stemming from the early periods of microprocessor development where resources such processor speed and memory were restricted, and covering the current period where embedded processing exists in many small low powered consumer devices.
3.3.7.2 Technology Selection

There is a wide range of MCU devices available with newer lower-powered devices constantly entering the market. The Hitachi H8-3664 MCU was a point-in-time decision based on meeting the required physical size, memory size, subsystem implementation, but most importantly this family of devices could operate at very low power. At clock frequencies from 1MHz to 16MHz the current drain is linearly related to clock frequency. At frequencies of 10MHz and below, the supply voltage can be as low as 3.0V. For ultra-low power use where processing requirements are minimal, the system can operate using current as low as 25micro-amps (4kHz operation). Processing power can be adjusted to meet demand by dynamically changing the system clock frequency. Like many others, this family of MCUs also implement a 'sleep' command which effectively shuts down the Arithmetic and Logic Unit (ALU), further reducing power.

3.3.7.3 Minimise Subsystem Power Use.

In common with many modern devices, the H8 MCU can reduce power use by disabling the internal subsystems. This RTOS disables all subsystems not specifically turned on by the application. Depending on the application, the programmer may also be able to reduce power consumption by the "in-use" subsystems and their associated peripherals.

As an example, when monitoring movement using triaxial inertial sensors at lower sample speeds, duty cycling of the sensors and the MCU subsystem can substantially reduce current drain. Table 3.4 reproduces values from data sheets for the H8 MCU [41], and the Analog Devices ADXL202E Accelerometers [42]. Using power cycling, current drain due to 150Hz accelerometer sensing can be reduced from 2mA (sensors always on) to 0.6mA. Reduced sampling rates further reduce current drain.
Table 3.4 Current draw for accelerometer and ADC subsystem.

<table>
<thead>
<tr>
<th>Subsystem/Device</th>
<th>Supply V</th>
<th>( f_{osc} )</th>
<th>I (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>5V</td>
<td>16MHz</td>
<td>2mA (max)</td>
</tr>
<tr>
<td></td>
<td>3.3V</td>
<td>10MHz</td>
<td>&lt;1.0mA</td>
</tr>
<tr>
<td>Accelerometer</td>
<td>5.25V</td>
<td></td>
<td>1.0 mA each (total 2mA)</td>
</tr>
<tr>
<td></td>
<td>3.3V</td>
<td></td>
<td>0.5 mA each (total 1mA)</td>
</tr>
</tbody>
</table>

**Duty Cycle Switching**

- Current drain = 2mA (1mA ADC + 2 x 0.5mA Accelerometers)
- Using ~2ms to switch on and sample Accelerometers.
- 2ms x 150 samples/s x 2mA = 0.6mA (100 samples/second = 0.4mA)

This example is somewhat simplified. The actual savings are greater as the ADC only needs to be switched on at completion of the Accelerometer power-up.

**3.3.7.4 Frequency control of MCU.**

Due to the nature of semiconductors, parasitic capacitances exist within the device and as the frequency of operation increases, power loss due to parasitic capacitance also increases. At MHz frequencies the Hitachi H8 processor has a linear relationship between the frequency of operation and the current drawn from the power supply. The H8 has provision for connecting two crystals and can operate at the frequency of either crystal or at various sub-frequencies, determined by internal programmable clock dividers. By operating at the lowest possible frequency, the current drawn can be minimised. For some applications the processing frequency can be varied to meet the processing load requirements. If the processing speed is varied then consideration must be given to the source of the clock-ticks used to drive the RTS. In the H8 example, clock ticks can be generated in three ways: (1) generated externally and connected via an interrupt line, (2) generated internally from the main crystal frequency combined with clock dividers and a counter-timer or (3), generated from a combination of the secondary crystal, dividers and a counter-timer. The appropriate solution is application dependant however the design of the RTOS enables any solution to be implemented with minimal changes.
In the athlete monitoring applications, the system operates at a fixed frequency that is appropriate for sampling, processing and communications. For a low-power weather station implementation or similar application, the system could operate both the clock-tick and internal processing off the secondary crystal (32kHz watch crystal) (35 micro-amp current draw). When higher processing power is required, such as for communications, the RTS switches the ALU and internal subsystems (excluding the clock-tick source) to operate from the main higher frequency crystal.

3.3.7.5 Utilising sleep state.

When no useful processing is being performed, the system can be set to a sleep mode. The sleep mode stops all processing by the ALU and this in turn reduces the current drain of the MCU. Ultimately, at the completion of all interrupt handling, the system returns control to the BGS control loop. As described previously and shown in Fig. 3.5, the BGS executes the schedule once and at completion, the system enters the sleep mode. Any interrupt, including the clock-tick source will cause the system to exit sleep mode. On systems with applications that require a high clock-tick rate for specific timing requirements, it may be that 95% of clock-tick interrupts do not result in the execution of any task in the RTS. In this situation no new event is triggered or data gathered and therefore the BGS has been triggered unnecessarily. To reduce this triggering of the BGS and to maximise sleep time, the volatile global variable executeBGS is set to FALSE by the clock-tick interrupt handler if no RTS tasks are scheduled.

If a system implements a high rate clock-tick in conjunction with aperiodic event handling where the aperiodic event requires immediate data processing, it may be necessary to disable the executeBGS negation step. This prevents a scenario where an external interrupt triggers some data collection activity and the data is placed in a queue for background processing. After the external interrupt but prior to completion of the interrupt handler execution, the clock-tick interrupt event occurs but no tasks are scheduled for execution. The executeBGS flag is set to FALSE. In this situation the BGS will not begin execution on completion of the interrupt. This is not a problem if sufficient other interrupts occur that will result in the BGS executing within the
necessary timeframe. Use of the executeBGS functionality is the programmer's choice based on the application determined system dimensions (as per Section 3.3.1).

### 3.3.7.6 Low power processing algorithms.

All processing algorithms have been designed to minimise the processing requirements without seriously impacting the available RAM and ROM resources. The following are examples of how processing costs are reduced.

- **RTS**: The real-time scheduler only executes when there is an item in the schedule due for execution. When the RTS does iterate through the list of tasks it tracks the next interval on which an item is scheduled to execute. Therefore it does not iterate through the schedule on every clock-tick, this saves considerable processing cycles.

- **BGS**: The BGS will execute after any interrupt and offer processing time to each task in the schedule. For a BGS operating in a co-operative mode, tasks are designed to return immediately if no processing is required. For a BGS operating in an execute and remove mode, if there are no tasks left to execute then no processing will occur. After executing the BGS, the system goes into sleep mode. The mechanism described in the previous section also saves BGS processing cycles.

- **Silence Compression**: This is one of the various forms of data compression used in different applications. Silence compression compares a number of successive data samples and where a single sample is different from the rest in the set, that sample is modified to match the others. Since the data set changes on each new sample, a large number of comparisons can be required. The implementation in this system 'remembers' the results from previous comparisons and minimises the necessary processing. Other compression algorithms are likewise written for both small code size and low processing cost.

In common with many embedded systems, complex calculations are replaced with table driven algorithms. As an example, Hamming Forward Error Correction (FEC) coding and decoding is performed using tables in memory.
3.3.8 Communications State Machines

During continuous operation as a Wireless Sensor Network node, the main communications tasks involve transmitting packets of data and the reception of clock synchronising messages. These communications tasks occur in half-duplex mode and therefore the system can be sending data or receiving data but never both. Both tasks can be broken down into a sequence of steps or tasks that, once a communications process is triggered, are controlled by the communications subsystem interrupt. To allow the communications interrupt to control the sequencing of the steps, an array of pointers is used to indicate each successive task as illustrated in Fig. 3.10. As a particular task or step is completed, the array index is incremented and upon the next interrupt, the next task in the sequence will be called.

![Diagram of Communications State Machine]

Because of the design of the communications system, reuse of communications functions can be maximised, communications function size can be minimised and the handling of the communications subsystem interrupt is standardised.

---

24 In the context of a wireless sensor node, there is no necessity for full-duplex operation given the operation of the communications medium. There is no restriction on the RTOS to only operate in half-duplex mode. Full-duplex mode requires a Full Duplex ISR, no other changes are necessary.
In the case of the clock synchronisation message of Fig. 2.23, the transmitted message is designed to enable synchronisation of the RF receiver, the receiving UART and the receiving OS. While the RF receiver is powered on and the UART is in receive mode, any RF noise will result in garbage (or random) messages coming in through the UART. The data synchronisation is required to identify the message start. The state machine of Fig. 3.11 is designed to handle the clock synchronisation format. In this case the message header "V2s" indicates a synchronisation message and puts the receiver state machine into the state where it can accept and decode the incoming data. Successful reception of a valid synchronisation message shuts down the receiver hardware. In the case of failure to successfully decode the message, the receiver will be shutdown by an appropriate task on the scheduler.

![Synchronisation Message State Machine (for ASCII based messages).](image)

To add a message type similar to that of Fig. 3.11 requires a different message header, a new 'success' exit function and a new message column in the array in Fig. 3.10.
3.3.9 Programmers Interface and Device Drivers

This section details the developed RTOS's Application Programmer's Interface (API).

Application programming has very few requirements due to the design of the system. All C language functions developed for:

1. Execution by a scheduler,
2. Use as an ISR,
3. Use by the communications state machine,

must be of the form \texttt{void function\_name(void)}. All RTS called functions or interrupt driven functions must be atomic or as near atomic as possible.

There are only two other components of the RTOS programmer's interface, adding tasks to either scheduler (Fig. 3.12), and the creation and use of pipes (Fig. 3.13).

\begin{verbatim}
BOOL addScheduleItem(unsigned int nSchedule, 
                     unsigned long nextIval, 
                     unsigned long ivalSize, 
                     void (*sch_func)(void), 
                     unsigned char pVal, 
                     unsigned char mVal, int rCount);
\end{verbatim}

Fig. 3.12 RTOS programmer's interface for adding items to a schedule.

\begin{verbatim}
int createQueue(pQueue q, unsigned char size);
void freeQueue(pQueue q);
unsigned char data_avl(pQueue q);
char write(pQueue q, unsigned char data);
char read(pQueue q, unsigned char * data);
\end{verbatim}

Fig. 3.13 RTOS programmer's interface for creating and using queues (pipes).

The \texttt{addScheduleItem} call of Fig. 3.12 and the Queue (pipe) interfaces of Fig. 3.13 describe the basic OS services available to the programmer. Due to the small size of the available memory, the Queue size is limited to a maximum of 255 bytes. Unlike traditional UNIX pipes (p.61 in [16]), these 'system calls' are non-blocking. A failure to write to a pipe must be handled within the calling function. With proper design of
the application this should never unexpectedly occur since the system should be
designed to be schedulable.

A number of serial communications functions exist which can be used by the
application programmer to send and receive data. Shutdown functions also exist for
various MCU subsystems.

<table>
<thead>
<tr>
<th>Configure Subsystem</th>
<th>Subsystem ISR</th>
<th>RTS called function</th>
<th>BGS called function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Called during setup or as required</td>
<td>Atomic or near atomic. Called by Interrupt</td>
<td>Atomic or near atomic. Executed by RTS.</td>
<td>Co-operative design. Executed by BGS.</td>
</tr>
</tbody>
</table>

Fig. 3.14  Functions to be written by application programmer.

The application programmer writes a number of small functions of the types identified
in Fig. 3.14. If the application requires a MCU subsystem to be configured, appropriate
code must be written. If a subsystem requires interrupt servicing, a short ISR must be
written. During configuring of the subsystem the system interrupt handler must be
pointed to the user ISR, this step can also be performed elsewhere. If an application
requires action to occur based on the RTS, the RTS function must be written. The RTS
function can be added to the schedule using the addScheduleItem system call.
Similarly, if background processing is required, a background processing function is
written and added to the BGS using the same addScheduleItem system call.

**Technical Limitations**

Some technical limitations exist when developing applications and some of these
limitations may be compiler dependent, depending on the target system. The H8 target
system in this case has a large number of registers that could be utilised to generate
speed improvements (Register based programming). These were not utilised to the
extent possible due to the lack of context switching. Most variables are passed on the
stack (Stack Based Programming). Because the system utilises the passing of function
pointers as a mechanism for responding to interrupts and function calls, the target
system compiler cannot calculate the function call network path. Because of this, the
calculation of stack depth, in particular the maximum stack depth, at any point must be
performed manually by tracing the possible function call paths.

For some application processes, the compiler may use many ALU registers. These may
not be automatically placed on the stack on an interrupt. This is not a problem if these
registers are not used by any other process. In the case where complex application
processes exist, and these processes use many registers each, the interrupt handling
should implement a FLIH (refer section 3.3.4) where the pushing and popping of additional register sets is centrally handled.

### 3.3.10 Loose Coupling Between Hardware and RTOS

An Embedded RTOS, by its nature, will have a closer relationship with the hardware layer than a traditional desktop OS. This is because desktop and business computer OSes are implemented with complex hardware and require many levels of abstraction to obtain the appropriate level of system calls. In contrast, many embedded RTOS applications are predominately concerned with the controlling of outputs and detection of inputs.

This RTOS architecture provides, at a minimal processing cost, isolation between the hardware and the core functionality. At the C programming level, neither the schedulers nor the pipes directly interact with the hardware. Of the library functions used by the RTOS, the function calls of malloc() and free() are standard C functions. The sleep() and set_imask_ccr() functions are specific to this family of hardware but can be substituted or abstracted for other systems. All user ISR functions are of a standard format and handling of ISRs uses RTOS provided function pointers rather than directly coding using hardware specific ISR names.

As application programmers of embedded systems must code directly for specific hardware, moving an embedded system to a different hardware platform will require considerable rewriting of application device driver code. With the RTOS developed in this thesis, the RTOS services and functionality do not require re-writing.

### 3.3.11 RTOS Clock System and Support for Synchronisation

Clock synchronisation requirements are application dependent and range from an application having no specific synchronisation requirement to applications requiring strict full-network synchronisation - to within some specified tight limit. Typically clock synchronisation across sensor nodes is required for data fusion or for wireless network operation such as TDMA implementations as covered in Chapter 2.
A brief literature search revealed a myriad of clock synchronisation algorithms. Any particular algorithm would require assessing against the hardware resources and the wireless network implementation however each implementation must have specific requirements that should be met within the structure of this RTOS. Two interrelated functions are required, clock error detection and clock error correction.

The mechanism for maintaining alignment in a slave-controlled network requires the node to receive a clocking message, determine the error and apply a correction. The places where the correction can be applied include:

1. Adjusting the crystal frequency (using microprocessor controllable crystals),
2. Adjusting the counter-timer (incrementing / decrementing the target count).
   a. Preferred counter size 24 bit.
   b. Common counter/timer size 16 bit.
3. Gross scheduler-clock adjustments (scheduler-clock stepped forward or back)

The synchronisation mechanisms will be discussed within the context of the RTOS clocking system identified below.

### 3.3.11.1 RTOS Clock System

The overall clock system of the RTOS (Fig. 3.15) was designed to meet the requirements of a variety of applications and wireless networking topologies. Specifically, the clock system was designed to handle; event triggering, measurement of elapsed time via the system clock, tracking of network global time, global time stamping and clock synchronisation mechanisms.

---

25 See Chapter 2, section 2.6 for discussion on network synchronisation.
3.3.11.2 Crystal

Crystal selection is determined by both MCU and overall system requirements. The clock system crystal may also be the MCU clock source, which places limitations on the available crystal frequencies. Some form of microprocessor controllable crystal is preferred but not necessary. Crystals that meet MCU requirements for serial communications are not optimal for generating milli-second timers. The target system (Hitachi H8) allows two different switchable clock sources, although the source of the clocking signal for global-time synchronisation could be a separate crystal connected through an external Counter/Timer input. Using the Hitachi H8, it is possible to run the RTOS system clock from one crystal, while internally distributing the other crystal signal as the internal clocking signal. This allows the internal clocking to be sped up or slowed down to meet power and load requirements, without affecting the real-time system clock. This would also be the case if the real-time clock source were an external source connected to the counter timer.

3.3.11.3 Hardware Counter/Timer

The hardware counter/timer counts crystal clock pulses and generates an interrupt when the count reaches a specific target. The precision of the timer is determined by; the number of bits in the counter-timer, the crystal frequency and the duration of the timing period. Various combinations of crystal frequency, timer bits, and timing periods, along with error estimates, are compared in Table 3.5. Using timers, the worst-case error occurs when the target period requires a count that is half a count in error. The resultant percentage error may be small but the cumulative error can
increment quickly. For small timing durations, modifying the count will only modify gross timing drift. From the examples of Table 3.5, using a crystal of 9.8304MHz and using a timing duration of 5ms measured with a 16 bit timer, if the local crystal is 30ppm slow relative to the reference, then the timer count would need to be reduced from 49152 to 49151 or 49150. The resultant error would be approximately 1ms every 98 seconds. By alternating the target count between 49150 & 49151 at a rate determined by the error estimate, the 1ms cumulative error period can be extended considerably.

Table 3.5 Example combinations of Crystal Frequency, Timer Bits, Timer Error and Timing Precision

<table>
<thead>
<tr>
<th>Nominal Crystal Freq.</th>
<th>Timer Bits</th>
<th>Duration / Count</th>
<th>0.5 count Error</th>
<th>Cumulative 1ms Error</th>
<th>1 count equivalent ppm</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.8304MHz</td>
<td>16 bit</td>
<td>1ms/9830</td>
<td>0.005%</td>
<td>19.7s</td>
<td>+/- 100ppm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5ms/49152</td>
<td>0.001%</td>
<td>98.3s</td>
<td>+/- 20ppm</td>
</tr>
<tr>
<td></td>
<td>24 bit</td>
<td>1s/9830400</td>
<td>0.000005%</td>
<td>19661s</td>
<td>+/- 0.1ppm</td>
</tr>
<tr>
<td>1.2288MHz</td>
<td>16 bit</td>
<td>1ms/1229</td>
<td>0.04%</td>
<td>2.46s</td>
<td>+/- 813ppm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10ms/12288</td>
<td>0.004%</td>
<td>24.6s</td>
<td>+/- 81ppm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50ms/61440</td>
<td>0.0008%</td>
<td>122.9s</td>
<td>+/- 17ppm</td>
</tr>
<tr>
<td></td>
<td>24bit</td>
<td>1s/1228800</td>
<td>0.00004%</td>
<td>2457s</td>
<td>+/- 0.81ppm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10s/12288000</td>
<td>0.000004%</td>
<td>24576s</td>
<td>+/- 0.081ppm</td>
</tr>
</tbody>
</table>

In systems with multiple counter/timers available, one counter/timer can be used to apply periodic corrections to the count generated by another. From the examples above, a 1.2288MHz system using a 16-bit counter to measure 1ms, could be periodically corrected by another 16-bit counter measuring 50ms. The last milli-second in each block of 50 would be marginally shorter or longer than the preceding 49 milli-seconds. This combination results in a 50-fold improvement in cumulative error timing.

The target system uses a single 16-bit counter/timer, in conjunction with selectable prescalers, to generate the required timing intervals.

3.3.11.4 System Clock (Local Time)

To allow timing of events based on the current value of the system clock, the system clock is a monotonically incrementing software counter, incremented by the clock-tick
interrupt source as shown in Fig. 3.15. Any application requiring the measurement of elapsed time, and where the resolution of the system clock is satisfactory, can read the value of the system clock and use this in calculations. In this implementation the System Clock was a 32-bit integer, allowing counts to $4.29 \times 10^9$, 49 days if used as a milli-second counter, 136 years if used as a one-second counter.

3.3.11.5 Global Time Implementation

Global Time was implemented as the storage of a 32-bit offset between the local time and the global time as shown in Fig. 3.15. Corrections to the local copy of global time are made by updating the value of 'Local to Global Offset'. It is possible to store global time directly to avoid the offset manipulation and therefore save a small amount of ongoing processing. This is unsatisfactory if the clock is used to measure elapsed time since this would create a further requirement to track clock adjustment values. Global time was set as part of the network node configuration.

3.3.11.6 'Global' Time-Stamps

Global time-stamps are not to be confused with global time, global time-stamps occur in networks using reference broadcasts to generate a known common point in time, from which local events are measured. The local system, on receipt of a reference broadcast, records the reference broadcast identifier, sequence number and the local system time. Event times are given as the elapsed time since receipt of a particular reference broadcast.

3.3.11.7 Scheduler Clock

The scheduler clock triggers the execution of the RTS. The scheduler clock can operate at the same frequency as the system clock or at a reduced frequency, using a software clock divider, depending on system requirements. For synchronised operation the 'Local to Global Offset' contains the appropriate offset. For independent operation, no offset is required.

3.3.11.8 Clock Synchronisation Algorithms

The complexity of a clock synchronisation algorithm is hardware and application dependent. The requirements of the application determined the allowable clock drift, the power budget, the network data rates and the networking topology. The available
hardware determines the applicable mechanism for clock correction. Selection of the synchronisation mechanism was therefore a choice for the application programmers.

Various levels of sophistication of clock synchronisation are given in Table 3.6. These can be applied at a single clock correction point or at multiple clock correction points as listed in Table 3.7 and as suggested in Fig. 3.16. A combination of methods can be applied to a combination of correction points.

Table 3.6 Clock Synchronisation Sophistication for Embedded RTOS

<table>
<thead>
<tr>
<th>Type</th>
<th>Basic Functioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous Correction</td>
<td>Point-in-time correction.</td>
</tr>
<tr>
<td>Single Estimate</td>
<td>Use a base line to make a one-time estimate of the error.</td>
</tr>
<tr>
<td>Adaptive</td>
<td>Track the error and fine-tune the error estimate.</td>
</tr>
</tbody>
</table>

Table 3.7 Clock Correction Application Points

<table>
<thead>
<tr>
<th>Application Point</th>
<th>Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crystal</td>
<td>Marginally increase or decrease frequency</td>
</tr>
<tr>
<td>Global Offset</td>
<td>Modify the global offset to bump the scheduler clock up or down.</td>
</tr>
<tr>
<td>Counter</td>
<td>Calculate error and apply correction as a ratio of periods at two different counts</td>
</tr>
</tbody>
</table>
Continuous Correction Algorithm:
If the local copy of the global clock is found to be fast (or slow), one or more of the following steps are performed (depending on implemented clocking mechanism).

Crystal: The crystal frequency is decremented (incremented) by one step.

Counter/Timer: The count value is incremented (decremented) by one step.

This technique of slowing or hastening of the internal time source in the above manner, in response to detected error, is referred to as the Berkley algorithm [44]. In this case, modification of the counter does not obtain the expected correction since the counter/timer resolution is relatively poor.

Scheduler Clock: The scheduler-clock is stepped back (or forward) an appropriate number of clock ticks. Stepping occurs by altering the Local-to-Global Offset. Stepping the scheduler-clock forward is limited by when the next event is scheduled.

Single Estimate
A single estimate uses a relatively short time period to estimate the inter-clock error and that estimate is used to apply a correction to the clock system.

Crystal: Not applicable.

Counter/Timer: An estimate of the error over a period of several minutes may indicate an ideal fractional counter/timer value. As counter/timers cannot operate in fractions, the count value is oscillated between two values using a ratio that gives the approximate fractional value.

Scheduler Clock: An estimate of the error over a period of several minutes indicates a cumulative error system-clock error. The period to accumulate an error of one-clock tick is estimated and an item added to the schedule to repeatedly make the correction.

Examples: Counter/Timer.
This example illustrates how to calculate error and apply correction as a ratio of periods at two different counts. A system requires 1228900 crystal counts per second but is counting in milli-seconds using a counter value of 1229. The system will run marginally slow and will accumulate a 1ms error in approximately 12 seconds. By alternating the count between 1229 and 1228 at a ratio of 90:10 (90 counts at 1229 followed by 10 counts at 1228), the system will count exactly 1228900 in one second. The instantaneous error will be very small and the cumulative error zero. Small variations on these errors can be managed by small variations to the ratio.
Examples: Scheduler Clock

A system monitors a series of synchronisation messages and estimates error at 3.71ms per 60 seconds. This can be estimated as 1ms error per 16.1725 seconds. Using a 1ms System-Clock and a 10ms Scheduler-Clock, a 1ms correction to Local-to-Global Offset can be scheduled every 1617 intervals. The new error is approximately 0.15ms per 60 seconds or 1ms every 399 seconds. Using a 1ms Scheduler-Clock, a correction can be scheduled every 16172 or 16173 intervals. The new error is approximately 1ms every 1973 or 2027 seconds. This is a viable system provided the RTS is not used for strict local timing functions.

Fig. 3.16  Clock Synchronisation Mechanisms for Generic MCU System. Software implementation may be determined by hardware implementation and MCU functionality.
Adaptive Error Correction

In this situation, as clocking errors are detected, any existing error estimate is adjusted. The initial error estimate baseline is the duration to the first detected error. As elapsed time continues, the estimate improves.

**Crystal:** Not applicable.

**Counter/Timer:** On detection of the first clock error, the system sets up the switching of the counter/timer counts using an estimated ratio of fast and slow counting. The next detected error allows further tuning of the estimate.

**Scheduler Clock:** On detection of the first clock error, the system adds a scheduled correction based on the detected clock and sub-clock errors (current timer count).

Adaptive Error Correction Examples:

This example builds on the previous example but assumes synchronisation messages are broadcast every second. The messages are not necessarily received.

**Counter/Timer:** The system has a cumulative error of approximately 1ms slow every 12 seconds, no error was detected at 10-seconds but a 1ms error is detected at the 15-second sync message. This corresponds to 1229 cycles per 10 seconds or 122.9 counts per second that need to be contributed to generate one additional milli-second count. An initial ratio of 1229 vs. 1228 counts of 87:13 will result in a new error of 30 cycles per second or 1ms fast per 40.97 seconds. Taking into account the delayed detection of the error, at the time of applying the correction the system is 1.25ms slow. This will take approximate 51 seconds to correct and a further 41 seconds before the system is detected as being fast. The estimate can be recalculated from when the error reduced to zero or from when it is detected as 1ms fast.

A problem with this system is that the sub-clock or counter value must be tracked to detect that the error is reducing. An alternative is to apply an immediate correction at the time of error detection and calculate new ratios based on the time of error detection as opposed to the last time with no error.

**Scheduler Clock:** A system that has a cumulative clock error of 1ms slow approximately every 16.1725 seconds, monitors a series of synchronisation messages. No error was detected at 15s but a 1ms error is detected at the 18-second synchronisation message. A 1ms correction is scheduled for 15.01s. This correction is early by 1.1625s and the new error will be 1ms fast every 208.8 seconds.
Error Detection.
In the context of a TDMA transmission scheme, synchronising messages are transmitted within a highly deterministic timeframe. The various system latencies are known in advance and included in synchronisation protocol design. Some small probabilistic variability exists, determined mainly by the interrupt management at the receiver. This variability can be determined to lie within specific boundaries.

The synchronising message may intrinsically indicate a specific instance in the TDMA cycle, or may explicitly state a specific time. In either case, upon the correct receipt and decoding of the synchronisation message, the system clock and the timer count is read and compared to an expected value. This is the raw error. If the system is already engaged in clock adjustments, it may only be necessary to determine that the error is reducing.

3.3.11.9 RTOS Clock System Summary
This section described the implementation of an embedded RTOS clock structure designed to provide local time measurement, global time tracking, control of real-time periodic scheduling and network synchronisation.

Various mechanisms for hardware and/or software clock synchronisation were suggested and examples given of how these would function. No particular method can be preferred over any other without an understanding of the system dimensions, hardware design and network topology.

For some systems, where the application allows it, the synchronisation mechanism may be nothing more than a direct correction of the local copy of the global clock. A slightly more sophisticated method uses the RTS itself, to schedule corrections based on an error estimate. Where it is impossible to alter the value of the local copy of the global clock, due to using the RTS as a timing sequencing mechanism (as intended), or where local periodic tasks must maintain synchronisation across the network, manipulation of the counter/timer is used to improve system timing and to bring the local clock into synchronisation with the system master clock. Although not all possibilities were developed, the examples were sufficient to indicate available alternatives. (Refer Section 3.4 and Section 3.4.5 in particular, for implementations)
3.3.12 Support for Wireless Networking

The RTOS, because it is an Operating System, provides a variety of services available to the application programmer. The major components relating to networking include the ability to communicate and the ability to synchronise. Beyond this, there is little extra required of the RTOS. Some functionality, that is not specifically a function of the OS, may be useful in supporting wireless networking. This may include the use of electronic serial numbers and the ability to retain network configuration data. These are not OS functions but can easily be accommodated. The H8 target device was chosen because, among other reasons, it incorporated a small EEPROM component that could be utilised for just this purpose.

One of the key components of wireless networking is the network protocol. In wireless networking, the protocol consists predominately of messages exchanged between nodes in the network. These messages may include preambles, RF receiver synchronisation information and other components, along with the information content. Therefore, a protocol consists of the messages and related responses that elicit particular actions. The communications state machine described in section 3.3.8 supports the construction and transmission of outgoing messages along with the reception, decoding and actioning of incoming messages. The transmission and reception of the clock synchronisation message is one example of this. Forward error correction (FEC) encoding and decoding is also supported by functions written for this RTOS.

Various wireless networking protocols and topologies are discussed in Chapter 2. The proposed wireless networking options are implementable using this RTOS. Different options place different requirements on the RTOS. Interfacing to a wireless modem chip requires the RTOS to manage hardware and media layer protocols. Use of a WiFi component removes some of this complexity and can reduce processor load in other ways. If, for instance, clock synchronisation functionality is implemented using the Reference Broadcast Scheme of [51], the decoding of clock synchronisation messages is moved to the wireless hardware, reducing processor load.

This RTOS supports the requirements of Wireless Networking.
3.4 RTOS Results

The RTOS as described here, is in daily use in a number of projects where a low power, small footprint embedded RTOS is required. A number of these projects are enumerated in the introduction. The RTOS has been ported to MCUs other than the original target system. The RTOS has not been advertised but has been used in projects across different universities.

While the RTOS is simple and small, applications written for the RTOS can follow normal procedural programming techniques provided they are designed for restrictive memory and processing capacity.

The size of the system, in terms of memory requirements, and the processing load attributed to the OS and the applications has been measured in a number of situations. These results appear below.
3.4.1 Memory Requirements and Processing Load - Basic System

The target system used 32k FLASH memory and 2k RAM memory. The RTOS along with a simple 3-channel 50Hz sampling system was implemented and analysed. Sampled data was output via the serial port at 38.4 kbps, in a single block, once per second. Memory requirements are identified in Table 3.8 and the processor load for different processor operating speeds and RTOS clock-ticks graphed in Fig. 3.17. The Global Variables identified in Table 3.8 include pipes created with static declarations.

<table>
<thead>
<tr>
<th>Component</th>
<th>Memory Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program</td>
<td>1748 bytes (ROM)</td>
</tr>
<tr>
<td>Global Variables</td>
<td>99 bytes (RAM)</td>
</tr>
<tr>
<td>Constants</td>
<td>40 bytes (ROM)</td>
</tr>
<tr>
<td>Stack Depth</td>
<td>128 bytes (max)</td>
</tr>
<tr>
<td>Stack Size</td>
<td>256 bytes</td>
</tr>
<tr>
<td>Heap Size</td>
<td>512 bytes</td>
</tr>
<tr>
<td>Total ROM</td>
<td>1788 bytes</td>
</tr>
<tr>
<td>Total RAM</td>
<td>995 bytes</td>
</tr>
</tbody>
</table>

Note: Stack depth is the calculated maximum depth based on compiler output analysis.

Fig. 3.17 RTOS and application processing load measured for different internal operating frequencies and RTOS clock-tick rates. Load broken down by processing layer. Application refers to sampling the sensors.
3.4.2 Memory Requirements and Processing Load - Complex System

This system has similar sensing requirements to those identified in 3.4.1. The system has additional processing requirements. The sampled data was processed using various forms of compression (discussed in Chapter 7) and encoded for transmission in a noisy medium. To confirm the correctness of the processing phases, the 32k memory was pre-loaded with 3-channel sampled data. This data was processed by the embedded system and transmitted to a host system, which then decoded and de-compressed the data for comparison with the source data. Additional memory requirements, related to compression and encoding algorithms are identified in Table 3.9. The processing load for the RTOS, sampling, processing and communications is graphed in Fig. 3.18.

Table 3.9 Compression & Encoding Algorithm Code Sizes

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Code Size (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silence Code</td>
<td>278</td>
</tr>
<tr>
<td>DPCM Code</td>
<td>214</td>
</tr>
<tr>
<td>Comanding Code</td>
<td>324</td>
</tr>
<tr>
<td>Huffman Code</td>
<td>134</td>
</tr>
<tr>
<td>Huffman Table</td>
<td>41</td>
</tr>
<tr>
<td>Hamming Code</td>
<td>16</td>
</tr>
<tr>
<td>Hamming Table</td>
<td>16</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1023 Bytes</strong></td>
</tr>
</tbody>
</table>

Fig. 3.18 Processor load identified by process. System running 3-channel sampling @ 50Hz.
3.4.3 MCU and Sub-system Power Requirements.

To analyse the overall system requirements the current drain of the MCU was measured under different circumstances. The system of section 3.4.1 was analysed for each of the MCU frequencies and RTOS clock-tick intervals given in Fig. 3.17. These results are recorded in Table 3.10. The reduced operating frequency and reduced clock-tick interval reduced current to less than half. The reduction in current drain for the lower operating frequency did not meet the expectations based on the manufacturer data sheets. This may be due to the presence of support circuitry that could not be disabled but could not be measured independently. Based on data sheets for these components, they should not have made any substantial contribution.

<table>
<thead>
<tr>
<th>@ 9.8304MHz</th>
<th>@ 1.2288MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1ms interval</td>
<td>10ms interval</td>
</tr>
<tr>
<td>7.5mA</td>
<td>7.5mA</td>
</tr>
</tbody>
</table>

To estimate the effect of running different subsystems, the target MCU was analysed to ascertain the contributing factors to the overall current drain. Different subsystems were turned on and operated at maximum and the change in current measured. These results appear in Table 3.11.

<table>
<thead>
<tr>
<th>System / Sub-system</th>
<th>@ 9.8304MHz</th>
<th>@ 1.2288MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller only.</td>
<td>7.2mA</td>
<td>2.9mA</td>
</tr>
<tr>
<td>ALU operating at maximum</td>
<td>add</td>
<td>2mA</td>
</tr>
<tr>
<td>A/D operating at maximum</td>
<td>add</td>
<td>1.0mA</td>
</tr>
<tr>
<td>Clock-Tick Interrupt Timer</td>
<td>add</td>
<td>0.3mA</td>
</tr>
</tbody>
</table>

MCU current drain was measured with a digital multimeter.
3.4.4 Scheduler Testing

Because this system was written in the C programming language with the RTOS using abstraction to minimise hardware dependencies, it was possible to test most of the functionality on a desktop system prior to cross-compiling for the target system. Due to variations in cross-compiler configuration and operation, it was still necessary to debug the RTOS on the target system. Several debug session outputs are reproduced below. These can be interpreted using the debug key of Table 3.12.

Table 3.12 Debug Output Key

<table>
<thead>
<tr>
<th>E</th>
<th>Clock-Tick Interrupt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>Real-Time Schedule Executing</td>
</tr>
<tr>
<td>a</td>
<td>adding a task to a schedule</td>
</tr>
<tr>
<td>h</td>
<td>adding a higher priority task</td>
</tr>
<tr>
<td>m</td>
<td>middle priority</td>
</tr>
<tr>
<td>f</td>
<td>task added to front of schedule</td>
</tr>
<tr>
<td>i</td>
<td>iterating through schedule</td>
</tr>
<tr>
<td>l</td>
<td>last in schedule</td>
</tr>
</tbody>
</table>

The debug output of Fig. 3.19 represents processing where the RTS has been set to execute on every clock-tick, regardless of whether there are any scheduled tasks. Tasks A,B,C and D have been added to the RTS and at a certain point a background task is added to the BGS. This background task required the outputting of a long text string, which takes long enough to print that the process is interrupted by the RTS several times. The sequence "EZz" represents the clock-tick interrupt, followed by the execution of the RTS, followed in turn be the execution of the BGS. The sequence "EZXA" indicates the clock-tick interrupt is followed by the RTS execution and then the execution of real-time task A.

Fig. 3.19 Debug output for RTS and BGS operation
The debug output of Fig. 3.20 represents normal RTS operation where the RTS only executes if a task is scheduled to execute on that clock-tick. This output includes debug output of the system start-up where a number of tasks are added to the schedule. This section shows the operation of the priority process. The output sequence "aiiiil" indicates a schedule item was added and that as the priorities were compared with existing items, the new schedule item iteratively worked down the linked list that forms the schedule, finally ending up as the last item in the list. The bolded "zEZ" sequence immediately after the second "START", indicates that the RTS executed even though there were no tasks scheduled. This is normal, as the RTS knew that tasks had been added and it needed to determine when the tasks were scheduled.

The bolded sequence "XafXDahzX" indicated that a real time task was executed and that this task added the first task to one of the schedules. By implication this must have been added to the BGS since the RTS already had tasks loaded. The RTS "D" task was executed and then another (un-named) task added a 'higher priority' task to the front of a schedule. In this case it was the BGS task designed to print out "I'm_a_queue_jumper...!". This task was inserted in front of the existing BGS task and therefore executed first.

These scheduler debug outputs, along with numerous other debug outputs, were used in the testing of the RTOS functionality.
3.4.5 Clock Reception and Scheduler Synchronisation

In these tests, various communications processes associated with clock messages and synchronisation were debugged.

**Communications State Machine - Clock Message Reception**

The content of Fig. 3.21 includes the debug messages from the communications state machine of Fig. 3.11. The various functions used by the communications state machine print their function name and in some cases the data they received, eg. "in rxV s". This indicates that the current function is "rxV" and it just received the data "s". These two test sequences cover scenarios such as, receiver noise prior to message reception, errored clock message and correct clock message. The incorrect clock was indicated by "CalcClockData SYNC FAIL" and the correct clock message by "CalcClockData Clock Data Success PABC".

```
SYNC MESSAGE RECEPTION  (TEST OUTPUT)
(sent data = "asV2sABCDPPC" followed by "V2sPABCDP")

Start-up
in rxV a  Invalid character stay waiting for V
in rxV s  Invalid character stay waiting for V
in rxV V  Waiting for V, V received
in rx2  2  Waiting for 2, 2 received
in rxS s  Waiting for s, s received
in rxClockData Receiving 1st byte of interval (MSB)
in rxClockData Receiving 2nd byte of interval
in rxClockData Receiving 3rd byte of interval
in rxClockData Receiving 4th byte of interval (LSB)
in rxClockData Receiving 1st byte of sub-interval (MSB)
in rxClockData Receiving 2nd byte of sub-interval (LSB)
in rxClockData Receiving 1st byte of check sum
CalcClockData SYNC FAIL
in rxV V
in rx2  2
in rxS s
in rxClockData
in rxClockData
in rxClockData
in rxClockData
in rxClockData
in rxClockData
CalcClockData Clock Data Success PABC <-interval value
```

Fig. 3.21  Debug output from state machine of Fig. 3.11 (some comments added).
Clock Synchronisation using Scheduled 1ms Corrections.

Clock drift between different units was discussed in Chapter 2 and multiple mechanisms discussed in section 3.3.11. In the following debug output, a slave unit had a 1ms clock correction task added to the scheduler using a periodic interval calculated manually from previous tests. At start-up the slave obtained the correct time from the master unit and updates the local clock value to the correct global time, "CLOCK: 15->600944". Every minute, or every 60000 milliseconds, the slave received a new clock message and printed both the local copy of the global clock and the received time. Three minutes after starting the test the master unit was disconnected and then reconnected after an interval of approximately 6 minutes.

Without the internally scheduled correction occurring, there should have been an error of approximately 3-4ms per minute or 20-24ms over the period the units were not communicating. Because of the internal correction, based on knowledge of the inter-unit drift, the slave unit maintained millisecond accuracy for a much longer period than previously possible.

This indicated that the units should be able to operate successfully in a TDMA wireless network, or in a network requiring data fusion using millisecond accuracy, provided the units obtain some initial synchronisation and some periodic corrections. This result means that wireless units can save power by reducing the frequency of synchronisation message reception.

<table>
<thead>
<tr>
<th>START:CLOCK:</th>
<th>15-&gt;600944</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOCK:</td>
<td>610003-&gt;610003</td>
</tr>
<tr>
<td>CLOCK:</td>
<td>670003-&gt;670003</td>
</tr>
<tr>
<td>CLOCK:</td>
<td>730003-&gt;730003</td>
</tr>
<tr>
<td>CLOCK:</td>
<td>790003-&gt;790003</td>
</tr>
<tr>
<td>CLOCK:</td>
<td>1150003-&gt;1150003</td>
</tr>
<tr>
<td>CLOCK:</td>
<td>1210003-&gt;1210003</td>
</tr>
</tbody>
</table>

Fig. 3.22 Local Clock and Remote Clock comparison used scheduled clock corrections.

In the above test, the slave unit used a wide receive window to ensure it captured the synchronisation message. Normally the window starts 10ms prior to the expected receive time and would continue to 10ms after the normal cut off time. With improved synchronisation the window width can be reduced and receiver power reduced.
3.5 RTOS Summary

This chapter has demonstrated a small footprint, low-power Real Time Operating System (RTOS) designed for embedded systems operating within a synchronised network. The RTOS has minimal functionality beyond the core necessities of:

- Real-time periodic task scheduling and execution.
- Real-time aperiodic task response and execution.
- Background task scheduling.
- Inter-process communications (IPC).
- External communications.
- Network clock synchronisation.

The system APIs and programming requirements are likewise small in number and non-complex in nature.

The core of the RTOS is easily portable to different hardware due to the use of a high-level programming language (C) and appropriate levels of abstraction. The conceptual RTOS is also small enough that a competent programmer could quickly duplicate the RTOS functionality without the source code.

Limitations of this RTOS are generally related to the limitations of the target systems. Context-switching is not used due to the limited available memory and the generally non-existent memory management hardware. Protections mechanisms such as locks and semaphores were not used due to the limited hardware support. Resources such as pipes and the system clock were not registered in an internal resource directory and therefore rely on the application programmer to track their locations and usage. This is not a problem where the entire RTOS and application may only take a few thousand lines of source code.

While none of the technologies, techniques or principles used in the development of this RTOS are unique, the combination presented here has been effective is producing an RTOS suitable for use in the novel environment of wireless monitoring of teams of athletes.
3.6 References


3-60


[41] 2001, Hitachi H8-366X Hardware Manual, Hitachi Ltd, Tokyo, Japan


4 Introduction to Athlete Accelerometer Data

4.1 Introduction

This chapter deals mainly with the limitations, processing and experimental results related to the interpretation of Athlete Data collected from triaxial accelerometers. Over the entire gamut of the Athlete Tracking Project, athlete data comes from a variety of sources such as accelerometers (uni-axial and tri-axial), Global Positioning Systems (GPS), magnetic sensors (triaxial), gyroscopic sensors (triaxial), in-sole sensors and a variety of medical sensors (heart rate, lactate levels and oxygen use).

The data and extracted information is useful to a variety of sports-science users such as physiologists, biomechanists, physiotherapists, elite coaches and athletes. While there is a variety of sensors and a range of extractable information, many sensors are specialised and only useful in limited circumstances. GPS, for instance, is not particularly useful for monitoring swimmers or for use in indoor activities, unless the venue has GPS pseudolites (pseudo-satellites) installed. For general-purpose athlete monitoring, the most rudimentary low-cost MEMS sensor package uses accelerometers therefore, the greatest benefit in analysing athlete data comes from extracting the maximum content from accelerometer data.

In this chapter, athlete data refers almost exclusively to the raw and processed data from triaxial accelerometers mounted on an athlete engaged in field or court based sporting activity. Subsequent chapters discuss specific uses for accelerometer data in physiological monitoring (Chapter 5) and biomechanical monitoring (Chapter 6). The processing associated these forms of monitoring is developed in this chapter. Sampling rate and sample size, with respect to various types of athlete data and the compressibility of that data, for logging and wireless transmission, is dealt with in Chapter 7.

This chapter is mainly experimental and investigates various interrelated subjects with respect to the use of accelerometers in the elite sporting environment such as:
4.2 **Operational environment and limitation of accelerometers.**

Accelerometers are a form of inertial sensor. In the examples of Fig. 4.1, a proof-mass is attached to the sensor body by a suspension system. When acceleration is applied to the sensor body along the axis of operation, the sensor body moves relative to the proof-mass. The change in position is a measure of the acceleration applied to the sensor. If the axis of operation is vertical, then even if the sensor body is stationary, the proof-mass will move to a position representing one-gravity acceleration.

Some Micro-Electro-Mechanical Systems (MEMS) accelerometers are based on this principal. In silicon MEMS implementations, it is usual that the sensor body is silicon, the proof-mass is silicon, and the suspension system is silicon, with the components etched in the silicon using similar techniques to semiconductor manufacture. The change in position of the proof-mass is detected as a change in capacitance between different components in the system. An image of an Analog-Devices MEMS accelerometer, similar in function to those used for collection of data discussed in this chapter is depicted in Fig. 4.2.

Accelerometers are widely used in the study of a variety of human and mechanical movement activities. In 1978 Rewick et al.[1], demonstrated the use of a head mounted accelerometer as an energy expenditure estimator and later, Montoye et al. (1983)[2], demonstrated a portable accelerometer used for the same purpose. Chapter 5 investigates this application in more detail. Accelerometers are used for monitoring the
activities of the elderly and the infirm\textsuperscript{26} and they are often used as estimators of position of both humans and machines\textsuperscript{27}. In these and other areas their use, the derivation of useful information is the subject of continuing study.

\textbf{Fig. 4.2} Image of Analog Devices ADXL150 accelerometers, photo (including labelling) extract from [3]. The tether is suspension system for the proof-mass and is in the form of a 'Folded Beam'. Capacitance is measured between the fixed and moving fingers.

The environment of elite athlete monitoring has a number of inherent limitations and impacts on the triaxial accelerometer data. Studies of the elderly and studies of daily activity cover activities that are generally of low intensity. Physically intense activities commonly associated with elite sport are not differentiated and are simply labelled 'intense'. Activities such as football are not only intense but there are factors not

\textsuperscript{26} Monitoring the elderly and the infirm is heavily studied as it relates to our ability to care for these members of society. An example of the use of accelerometers is a paper by Najafi et.al. [4], which investigates the use of wavelet transformations of accelerometer data to identify positional transitional activity of the elderly such as laying to sitting, sitting to standing etc.

\textsuperscript{27} Accelerometers are often used to estimate the position of humans or machines within some specific context. Without external references, these ‘dead reckoning’ systems are unable to estimate position with any degree of accuracy. Lee & Mase use accelerometers, in conjunction with other sensors, to estimate position within a known floor plan [5]. Randell, Djialis & Muller perform similar dead reckoning within the context of a street map [6]. These authors work within a low complexity environment where the subject is undergoing simple locomotion. Gianisanti et.al. used accelerometers to monitor limb position [7], this is a different problem since limbs have known degrees of freedom, limits and interrelationships. Doogue & Walsh combine accelerometers with other sensors and GPS to monitor the position of a racecar on a track [8]. The GPS provides constant resets of the cumulative error.

4-3
normally considered 'daily activity' such as collisions, tackles, jumps, swerving and scrums or mauls.

4.2.2 Accelerometer Factors & Limitations

Considering the number and variety of environmental factors and limitations affecting the collection and use of accelerometer data, many of them are simply enumerated below (in alphabetical order). These factors are those that affect the use of accelerometers in general, and affect the use of accelerometers in the monitoring of elite athletes in particular.

**Acceleration**: Many devices incorporate accelerometers rated to around ±2g (gravities). As the downward force of gravity accounts for 1g on the vertical axis, a ±2g device can only capture athlete activity between +1 and -3g. This is sufficient for activities such as swimming and rowing but inadequate for running and activities such as football.

The accelerometers used for the majority of experiments reported in this thesis were rated at ±2g but experimental results indicated that they operated linearly to around 7g (Fig. 4.3). Vertical forces from fast walking or running were observed to generate acceleration in excess of 6g (Fig. 4.4). These large accelerations were confirmed using ±10g accelerometers from the same manufacturer. The recovery time from overload is not identified in the manufacturer's data sheets. The combination of unknown peak acceleration and unknown recovery time, combine to generate an unknown error component. These very short and sharp acceleration peaks were observed to occur at a point at, or very shortly after, the initial foot contact. The acceleration spike was much larger than the acceleration experienced through the rest of the foot contact phase.
Fig. 4.3  Load response from Analog Devices ±2g Accelerometer. The response is linear until the limit is reached. Units used in experiments appeared to reach overload somewhere in excess of 6g. Acceleration obtained by installing the accelerometer in a spinning machine.

The situation regarding unknown peak acceleration is not clarified when combining accelerometer data with force plate data. Force plate data for running athletes (supplied by the AIS) indicates typical foot contact forces in the vertical plane of around 3-4g. The difference in acceleration could be due to a variety of factors. Possibly the force plates have limited bandwidth and cannot detect the higher
frequency contact impulse signal. This limited bandwidth may be mechanical and due to the required physical size and consequent inertia of the force plate itself. As one alternative, the force detected on the athlete's trunk may be a combination of ground reaction force and muscular forces anchored against the inertia of the lower limbs. This could spread the ground reaction force across a longer period while generating a shorter more intense acceleration on the trunk. This is the less likely scenario as, during the running action, the athlete's leg is straighter during the initial contact, then the muscles undergo eccentric, isometric and concentric phases as the downward movement of the body is resisted, stopped and finally reversed. This has the effect of cushioning the body contact resulting in a more gradual change of acceleration.

The intense acceleration may also be an artefact of the anchoring of the sensor itself. In this case, the sensor (and its packaging) is not arrested by the initial foot contact but continues in a downward direction until reaching the limit of the elasticity of the anchoring material. This would result in a sudden sharp deceleration.

While there has not been a specific attempt to identify the source of the intense acceleration and the discrepancy between it and the force plate data, the combination of sensors and athletes tested suggests that the large forces do exist but may be exacerbated by the attachment to the athlete's body. This also implies that the force plate data is subject to filtering.

It would appear that accelerometers rated at ±2g, while acceptable for some sports, are not appropriate for monitoring athletes engaged across the range of elite sports. While these ±2g accelerometers proved useful due to their ±6g response, there is no guarantee that other ±2g devices will exceed the rated specification by such a margin.

**Acceleration, Velocity & Displacement:** Acceleration has a physical relationship with velocity and displacement. By monitoring the displacement of an object as a function of time and differentiating the resultant data, the velocity can be determined. Similarly, by differentiating the velocity, the acceleration can be determined. In this situation, where the acceleration is monitored, the converse is required, that is the integral of the acceleration gives velocity and the integral of the velocity gives
displacement. This relationship is used in some dead reckoning systems, in conjunction with other sensors, to assist in identifying location.

When monitoring athletes with accelerometers the sensors detect acceleration from a combination of sources. This acceleration is a combination of the acceleration due to gravity, dynamic oscillatory action (in the case of running, rowing etc.) and the translational acceleration imparted to the body causing it to move in a particular direction. Due to the structure of the body and the biomechanics of performing a particular activity, the body is constantly twisting and turning, and is therefore dynamically spreading all signals across all channels. Extracting a particular signal, such as the horizontal across-the-ground acceleration, is therefore very complex, requiring an understanding of the entire system.

Because integration is necessary to generate estimates of velocity and displacement, small errors can quickly compound. This makes the extraction of across-the-ground velocity and displacement difficult. This does not preclude giving an estimate of dynamic, in-stride velocity and displacement. In fact, the extraction of in-stride velocity and displacement could assist in the modelling of the athlete's biomechanics and lead to a method of extracting translational acceleration.

**Athlete:** The mounting of a monitoring device on an elite athlete engaged in competition must be psychologically non-invasive. Either the athlete is psychologically conditioned to the presence of the sensor through repeated use in training, or the sensor is mounted or used in such a way that the athlete is not conscious of its presence. These preconditions rule out the precise mounting of sensors on an athlete. As a result, sensors are generally located at a position approximating the correct position and in an orientation approximating the correct orientation. Inserting the sensor within a pouch in the clothing of the athlete simplifies the attachment of the sensors, but the sensors then become subject to various short and long-term signal noise factors. This noise interferes with the ability to cleanly extract biomechanical or positional information.

**Calibration:** The accelerometers are calibrated for both offset and sensitivity. These parameters can vary considerably between devices and even between the two
components of a dual axis accelerometer. These parameters are primarily affected by the manufacture process tolerances and have long-term stability [42]. Input to manual calibration requires exposing each axis to full positive and negative gravity to allow the estimation of the offset and scaling factors. This is subject to error if the sensors are not exactly aligned to the gravity vector. Calibration can use iterative error minimisation techniques or manual calibration however the 6-point Newton-Raphson based calibration technique of Lai et al. [10] gives good results. This technique does not require the device to be aligned exactly with gravity but only requires six distinct stationary orientations. A visual estimation of the effectiveness of a calibration technique can be performed by graphing the magnitude of the acceleration vector through the calibration data. Poor calibration exhibits a signal that changes magnitude with each different orientation (as depicted in Fig. 4.5).

![Calibration Comparison](image)

**Fig. 4.5** Calibration comparison. The vector magnitude of triaxial data calibrated using the Lai method compared to the same data using default offset and scaling values.

**Errors:** Accelerometers are subject to a variety of errors from different sources. Noise error increases with increasing bandwidth; orientation of the device with respect to gravity and the direction of travel affects the input signal. Accelerometer outputs are directly subject to quantising error and subsequently subject to calibration error. Loose coupling between clothing mounted sensors and the athlete generate artefacts related to the mounting. Accelerometers used in intense physical activity can be subject to overload and recovery related errors. Using accelerometers as a technique for dead reckoning of position is subject to serious cumulative error. The error in simple straight-line movement systems is substantial; when dead reckoning is attempted for
triaxial systems with two or three degrees-of-freedom an accelerometer-only system cannot estimate position [11]. Lai et al., using GPS and accelerometer systems located on a rowing shell, quantified errors due to quantisation and noise [12] and recommended techniques to minimise (but not eliminate) these errors through the use of appropriate sample rate and sample size.

**FFT:** Fast Fourier Transforms are a standard tool in the signal processing tool kit. For analysis of an electronic or mechanically produced signal, the FFT can usually produce useful output such as repetition frequency. When analysing athlete data, the FFT is not the optimal tool. This is due to the inter-cycle and longer-term variability of human movement. The FFT requires a reasonable number of cycles of data to make an accurate estimate while an athlete may go from stationary, up to a four Hz step rate and back to stationary in a matter of ten steps or three seconds. Even on longer samples - such as for a 1500 metre runner (Fig. 4.6), the variability in the step rate is only discernable as a broadening of the FFT peak. FFT output is stepped and the shorter the sample sizes the fewer available output levels. The example of Fig. 4.6 compares step frequency estimation using zero crossing detection and step frequency estimated from an FFT. In this example the sample rate is 500Hz and the FFT is recalculated every second using 4096 samples (approximately 8 seconds ahead). Using 2048 samples (4 seconds), the FFT outputs at 3.05Hz and 3.3Hz disappear. At 1024 samples, the FFT only outputs the values of approximately 2.92Hz and 3.42Hz. This limitation does not preclude the FFT from other forms of analysis of athlete data.

![Fig. 4.6](https://example.com/figure46.png)  
*Fig. 4.6  Comparison of FFT and zero-crossing detection estimates of step frequency during a 1500m run.*
**Filtering:** Filtering is used to remove components of the signal considered to be noise and at the same time extract other components considered to be of interest. A running athlete has an accelerometer signal that combines gravity, stride-rate and step-rate and various artefacts related to the athlete's particular style, the location of the sensors etc. There is very little signal power below the stride-rate (half the step-rate)(refer Fig. 4.29 or Fig. 4.46); alternatively it could be considered that all the dynamic signal begins at the stride-rate. As the dynamic signal of a running athlete, consisting of athlete input and free-fall, is oscillatory, filtering out the dynamic signal leaves the longer-term orientation signal. Filtering can be applied in software or hardware. Filtering can be used to remove the overload signal from heel strikes.

**Free Fall:** While accelerometers attached to a rigid object resting on the ground will detect gravity, if the rigid object is in free fall and is subject to no external forces, the sum of accelerations will be zero. While this is useful information, athletes are not rigid bodies and are constantly exercising muscular forces to move one section of their body by leveraging off the inertia or momentum of another section. A runner enters free-fall during the flight phase of every step (Fig. 4.7). In sports such as snowboarding, an athlete can be in free-fall for a substantial period [13]. The duration of 'air' time for a snowboarder can be estimated from the loss of acceleration although the actual transition may be difficult to determine due to the complex environment. For a runner, the period of inter-step free-fall is not discernable in the acceleration data (Fig. 4.8).

![Fig. 4.7 Contact and flight phases through a running step. In the period between the toe-off and landing, the runner is in free fall, with no applied external forces other than gravity.](image)

**Gravity:** All accelerometers are subject to the effects of gravity. The orientation of a stationary triaxial accelerometer assembly can be identified using the application of trigonometrical rules. With the exception of periods of free-fall, at all other times, the output of a triaxial accelerometer assembly will include a vector of length one-gravity with an alignment indicating the orientation of the sensor assembly with respect to
gravity. This information can be used to generate a rotational tensor (matrix), which can be used to correct vertical axis misalignment of the resultant collected data. The gravity vector is one of the most useful components of the triaxial accelerometer output however; it cannot be cleanly extracted in most of the situations of interest.

By rotating a triaxial sensor unit around all its axes, a ±1g signal will be applied to each accelerometer and the resultant data used in calibration. If sufficient rotations are performed the resultant data set, after calibration, will, when reproduced graphically in two or three dimensions, describe a sphere. If the rotations are performed by hand there will be considerable noise in the signal. One such data set is reproduced in Fig. 4.9. In this case the data includes a vertical jump and then the sensor unit is rotated by hand in each plane. The resultant data has been calibrated and filtered (4Hz Hamming Windowed Filter) before being multiplied slightly to expand the graphical representation. This data is superimposed over a sphere of unit radius (one-gravity) to allow comparison of the accelerometer data with the theoretical output. Reducing the filter frequency (0.75Hz) improves the quality of the sphere but it still contains distortions, assumed to be caused by the mechanical process of generating the data.
Impacts: Athletes in various sports are subject to impacts, which are transmitted through to the sensors. Impacts are generated in a variety of ways such as foot contact with the ground, contact with another athlete, contact with an object (a ball, a balance beam etc), contact with the ground (falling, tackled etc). These contacts generate forces of varying intensity. The impact forces are combined with other forces and with any consequential orientation changes and are therefore difficult to extract.

Foot Contact Forces: As noted in Acceleration above, foot contact forces can be detected with back mounted accelerometers. These forces start from somewhere on the surface of the toes or foot and are transmitted via the ankle, knee and hip joints before being detected by the accelerometers. Despite the intermediate linkages, these forces can exceed the limits of the accelerometers. While many of the devices used in prior research (as discussed in Chapter 5) use ±2g accelerometers (or similar) this overloading did not appear to be discussed. In the case of foot contact with the ground, it is known that the ground is stationary and provides an absolute reference point. Analysis of signals from shoe-mounted accelerometers can utilise this as an absolute reference.
**Player Contact & Tackles:** Impact forces on a rigid body should be able to be resolved into both magnitude and direction however, with athletes, various complexities are involved. Athletes undergo rapid changes in orientation immediately prior, during and post impact. A football player arrested in their forward movement by an ankle tackle undergoes rapid forward deceleration combined with a rapid change in orientation and smaller amounts of rotational and centripetal acceleration. Player contact does not involve any fixed frame of reference, an impact that resolves as 45 degrees to the direction of travel may in fact be a direct head on impact, with the player turning slightly prior to impact. A combined impact, player-on-player followed by a change of orientation and impact with the ground may not be resolvable into the component impacts.

**Manufacture:** At the time of development of the athlete monitoring system, MEMS based accelerometers were available in single and dual axis packages. More recently (2004), triaxial packaging has become available. Previously, to create a triaxial accelerometer assembly required the combining of a dual axis accelerometer mounted flat on a circuit board, with a single or dual axis accelerometer mounted vertically. The vertically mounted device would either be mounted directly on the circuit board or flat on a small vertically mounted daughter board. In both cases, there is no specific control over the orientation of the vertical component and the manufacturing error is unknown. An alignment error of three degrees between axes could result in a spurious 0.5\(\text{ms}^{-2}\) acceleration signal (9.8\(\text{ms}^{-2}\) x \(\sin(3^\circ)\)) on a horizontal accelerometer while the athlete is stationary (Fig. 4.10). This misalignment will result in faulty calibration and other unrecoverable errors (due to the collection of non-orthogonal data).

![Diagram](image)

**Fig. 4.10** Error due to manufacturing misalignment of accelerometer axis. In the figure, the error is relative to the gravity. The introduced error is dependent on the orientation of the system.
Manufacturing is also a factor in the operation of the individual components. The sensors used for collection of athlete data are used worldwide in motor-vehicle air-bag deployment systems, where they function as crash sensors. These devices are not specifically designed for monitoring human movement and may not be optimal for this purpose.

**Positioning of System:** The location of the system on an athlete is dependent on a number of factors including:

- Comfort.
- Safety.
- Effect on performance.
- Sport being monitored.

For biomechanical monitoring in a sport such as canoeing, a sensor is required on the upper body, but the sensor must not interfere with the rowing action where the oar is pulled up against the chest. It also cannot impede the operation of the back muscles. In football, athletes are commonly involved in frontal impacts therefore a front mounted sensor may cause injury problems. For football players, both upper body and shoe-mounted sensors will not correctly detect a prone player. For most athletes, moving a sensor from the centre of the lower back across to the edge of the hip causes substantial changes to the acceleration patterns. Body mounted accelerometers are essentially useless for monitoring cyclists since the body is anchored and undergoes relatively little movement. An accelerometer-based sensor improperly mounted on a swimmer can cause problems with streamlining, for example, a belt mounted sensor package was found to lift from the swimmer's body during turns, causing significant drag on the swimmer.

The position of the sensors must be standardised for each sport and must be either firmly located in the correct position or, the interpretation of the data must be immune to the kind of positional shifts expected in the sport in question.

**Processing Power:** Factors such as the number of sensors, sampling rate, sample size, logging capacity or radio bandwidth determine the processing power required. System size, weight and battery life provide balancing factors. The 'expressive footwear' [14]
combined multiple sensors including general accelerometers, shock accelerometers, digital compass, rotational sensors and insole sensors. This system collected data using a 16MHz processor and passed it via a 20kbps wireless link for off-board signal processing and data fusion. The 'Incremental Motion-Based Location Recognition' [5] prototype system performed on-board processing using an Intel Pentium MMX 266MHz processor. Where on-board, high accuracy signal processing is required, high processing power is also required. If it is necessary to over-sample signals to improve resolution, this improved resolution must be maintained through the use of floating point calculations, high performance digital filters etc. This may require the use of appropriate multi-channel DSPs to reduce to processing load of the general-purpose microcontroller.

**Rotation about an axis:** Athletes engaged in field and court team sports engage in complex movements many of which cannot be differentiated by accelerometers. A body rotation occurring along an axis of the triaxial accelerometers will not be detected. As the axis of rotation moves away from the sensor axis, some rotational acceleration may be detected although it is difficult to differentiate rotation from the external force generating the rotation.

![Axis of Rotation Triaxial Accelerometer Axes](image)

No acceleration detected

![Axis of Rotation](image)

Tangential and Centripetal Acceleration

**Sample Rate & Sample Size:** The appropriate combination of sample rate and sample size can be used to reduce errors and thus improve the accuracy of an accelerometer based dead-reckoning system [12]. This improvement is offset by the additional cost in terms of processing load, wireless bandwidth and the dynamic and non-volatile (logging) memory requirements. Over sampling reduces the memory and network bandwidth issues but still requires high sampling rates. Lower sampling rates reduce
processor and bandwidth requirements but may miss useful detail. The quality of the reproduced acceleration signal may not necessarily be important if the parameter of interest can be extracted from a lower quality signal. Therefore the appropriate sample rate and sample size is application dependent and is determined by the type of information to be extracted. A more complete discussion on sampling rate, as it affects data content, is included in Chapter 7. Techniques for minimising sampling rate but maintaining the accuracy of the extracted date are discussed in [15].

4.2.3 Factors & Limitations: Summary

The combinations of limitations and factors enumerated above, limits the use of accelerometer sensor systems from use in some quantitative forms of analysis as well as some qualitative forms. This still leaves considerable scope for the use of accelerometer-only sensor packages monitoring elite athletes.

For example, if the existence of unquantifiable impact error prevents the use of the system in distance estimation based on integration, this then removes the necessity for the high sampling rates and sample bits used for error minimisation. This has a flow on effect, reducing the processing power required. Conversely, the existence of an overload signal itself may provide the basis for some form of simplified qualitative analysis.

Some forms of quantitative analysis can be performed accurately in the presence a combination of error factors. For instance, gross calibration errors combined with manufacturing misalignment, poor sensor attachment and poor sensor-athlete alignment do not prevent the accurate estimation of muscle contraction rates (such as step rates).

Despite the range of factors that appear deleterious to the usefulness of the accelerometer data, there appears to be many opportunities for exploiting this data, provided the limitations are understood.
4.3 Signal processing and visualisation of Athlete Data.

A variety of signal processing and visualisation techniques were implemented to assist in understanding the data obtained from accelerometer based athlete data. Data for this analysis was collected from athletes engaged in a variety of sporting activities including specifically directed test manoeuvres as well as athletes engaged in competition. Typical processing included filtering to obtain dynamic or orientation information, differentiation to extract velocity and displacement information, multi-channel Fast Fourier Transformations (FFT) to extract frequency as a function of channel and running speed.

Accelerometers are commonly used as estimators of energy expenditure using the RMS of the acceleration as the estimator, generating a signal power vs. time function similar to the signal power vs. frequency generated by the FFT. This is an estimator of the power in the signal at a point in time and has been shown to have a high correlation with human energy expenditure for some types of activities. Both the absolute magnitude and the square of the magnitude can be used, but as the square of the magnitude requires less processing, the normal technique is to use the square of the magnitude of the acceleration. By using the square of the signal, the resultant graph is a time-domain analogy of the frequency-domain Power Spectral Density and could be referred to as the *Power Temporal Density*. In this case the Power Temporal Density is unscaled but the magnitude is commonly referred to as 'counts' and is dependent on the manufacturer of accelerometer-based device.

Visualisation is predominated by two and three-dimensional representations and comparisons of processed data and also includes stride-rate normalised multi-speed comparisons as well other attempts to extract information of interest. Many graphical presentations in this chapter are for explanatory purposes. In practice the volume of data is so vast that the processing is performed simply to obtain some summary data for the purpose of documenting a training session: eg the training session lasted 1 hour 10 minutes, the athlete ran 13010 steps with a maximum step rate of 3.3Hz and an estimated energy expenditure of X kilojoules. In other cases, such as gait-cycle normalised multi-speed running data, the graphical representation is the output and record of interest.
The following processing and visualisations are discussed in this chapter. There are many interdependencies between various processes and these are discussed in the summary (Section 4.3.13).

- Filtering.
- Orientation of Sensors.
- Power Spectral Density (FFT).
- Power Temporal Density.
- Autocorrelation.
- Gait Cycle Normalisation of Acceleration Signal.
- Extracting In-Stride velocity and displacement.
- Combining accelerometer data with synchronously collected in-shoe data.
- Step Rate Estimations

### 4.3.1 Test Data

A variety of accelerometry test data was collected and analysed. Testing included two 10-by-10 data sets, where ten athletes, monitored by triaxial accelerometers\(^{28}\), walked and ran at ten speeds on a hi-speed motorised treadmill and for one 10-by-10 data set, the athletes were measured for oxygen consumption during the activity. Some test data was collected from athletes performing set test pieces\(^ {29}\), such as walking and running up and down a specific number of steps, walking or running in set patterns with known dimensions, engaging in specific game plays - such as performing football tackles or jumping for a basketball shot. Finally, data was also collected from athletes engaged in training and competition, such as 1500metre runners\(^ {30}\), football players, swimmers etc. A number of these data sets are discussed in this chapter while others are extensively investigated in subsequent chapters. One data set in particular is used as a reference data set for a number of processing techniques. This data set represents a range of activities and therefore there are potentially a number of processing techniques applicable. The data set also highlights some of the limits that affect accelerometry based monitoring.

---

28 Triaxial accelerometers and support electronics (approx 40 grams total) inserted into centre of back of trouser waistband. Sub ten-gram sensor platform connected by flexible wiring to other components.

29 Triaxial accelerometers and support electronics and batteries inside small belt mounted 50gram unit.

30 Two triaxial accelerometers (3 grams each) taped in position either side of the L3-4 vertebra.
Test 16 - Referred to as the three-tackle sequence.

In this test an athlete, monitored by a triaxial accelerometer mounted on the trouser waistband at the centre of the back, repeats a sequence consisting of run forward, tackle a dummy, fall to the ground, get up, jog backwards to the start and repeat. Prior to the test a standing vertical jump is performed. The test sequence is repeated three times. This test involves a combination of activities, which include running, impacts and changes in orientation. Some raw and processed data from this test is presented below (in Fig. 4.12, Fig. 4.13, and Fig. 4.14) and is referenced by several of the processing sections. Other figures relating to this test data are interspersed through this chapter. Together they assist in interpreting a complex 3D signal.

![Fig. 4.12 Raw triaxial acceleration data combined with acceleration vector magnitude and Power Temporal Density. The three tackles are clearly discernable, particularly in the vertical signal where the offset due to gravity disappears while the athlete is on the ground. The instant after the tackle is effected the athlete is stationary and this is discernable in the Power Temporal Density.](image)

Although this data is discussed in detail later, some aspects can be quickly identified here. In the power temporal density (bottom graph Fig. 4.12) the impact/fall sequences
were identifiable. This signal exceeded the power in the running signal. In Fig. 4.14 the different frequency signature of the mediolateral signal (stride-rate) and the vertical signal (step-rate) was apparent.

![Graph of 3D Low Pass Filter Output](image)

**Fig. 4.13** Magnitude of output of 1Hz low-pass filter.

![Magnitude vs Frequency](image)

**Fig. 4.14** Power Spectral Density for inter-tackle run phase. Step frequency dominates the vertical and anterior-posterior channels (~3.5Hz). Stride frequency is dominant in the mediolateral channel (~1.75Hz).
4.3.2 Filtering

Filtering is used for two purposes, (1) removing unwanted signal and, (2) extracting wanted signal. The ability to digitally extract or remove signal while not impacting relevant information is determined by the type and quality of the filter, which is in-turn impacted by the available processing power and system memory.

Unwanted signal exists in different forms, such as a small noise signal riding on a large main signal, or as large impulse spikes that distort the overall signal. Wanted signal includes the gravity component of a signal, the signal in the absence of gravity, the fundamental vertical axis signal, the biomechanical gait 'signature' et cetera. Filtering that removes unwanted signal may also remove some components of desirable signal. The chosen techniques represent trade-offs between the different requirements. The effect of the filtering should be considered in subsequent analysis.

Post-processing of data, using the power available to desktop processing, can utilise the best filtering options and minimise the trade-offs. Low-power embedded systems have neither the processing power or the memory resources to perform optimal filtering. A Hamming Windowed Finite Impulse Response (FIR) Filter with a length of 300 elements requires 300 floating-point multiplications and additions per sample per channel (270,000 floating-point operations per second for 3-channel, 150Hz-sampled data) plus the integer operations for array manipulation (refer Fig. 4.17). A 300 element rectangular windowed FIR filter can reduce this to two floating-point multiplications and additions per sample per channel along with a small amount of integer arithmetic (refer Fig. 4.18). In both cases the filtering process requires that the last 300 samples for each channel be held in memory. For some applications a first order lag function can be substituted for the filter function - at a cost to the quality of the output. The first-order lag function does not require as much memory. To further reduce the processing power requirements of embedded systems, some filtering can be performed in hardware. Unless the filtering is switchable, this limits the available output from the system.
Pre-filtering to remove impact overload.

The impact related sensor-overload discussed previously contributes to 'confusion' in downstream processing, affecting the extraction of gait patterns, step-frequency, orientation and other information. In the example of Fig. 4.15, acceleration from multiple strides has been overlaid to visually identify the core pattern. The impact forces generated during initial foot contact exceeded the limits of the accelerometer. For this athlete, on this sensor (operating with 500Hz bandwidth at 150Hz sample rate), the rise time is very fast and occurs at the start of the step signal. For other athletes monitored using similar sensors, the impulse may occur later in the cycle and with a slower rise time. From the sequence of Fig. 4.15, it appeared that the overload only occurred on one side eg. only the left step. Moving only a few steps earlier or later in the data set and the overload appeared on both the left and the right steps but the underlying or fundamental shape of the acceleration signal during foot contact appeared consistent. This suggested that a very small change in some biomechanical aspect of the running motion, which didn't noticeably affect the fundamental shape of the acceleration signal, caused the disproportionally large impact overload. This may be attributable to a slight change in leg stiffness on landing.

![Fig. 4.15 Overloaded vertical acceleration signal for treadmill running at a constant speed. Five successive strides are overlaid to identify consistent features. Data sampled at 150Hz with a 500Hz accelerometer bandwidth.](image_url)

Because this impact overload can contain significant signal power it can distort filtering used to extract orientation or processing used to estimate the power in the vertical acceleration. To remove this impact overload signal but retain the key shape of the acceleration signal, hardware or software filters are applied to the signal from the accelerometers. Unless specifically viewing data for impulse distortion, all data sets
have been filtered using a filter with a response as in Fig. 4.16. This filter has no noticeable effects on athlete data from many sporting activities such as swimming (low impact and consistent stroke rates) or rowing (low stroke rates < 1Hz).

![Graph showing frequency vs. attenuation in dB](image)

**Fig. 4.16** Single pole hardware filter or digital Infinite Impulse Filter (IIR) used to pre-filter data. Approximate bandwidth 3Hz.

**Noise Filtering**

High bandwidth sampling systems incur sensor noise, some of which is removed as part of over-sampling. On more limited bandwidth systems there is still considerable mechanical/biomechanical noise, either from the sensor mounting or from the small step-to-step variation in athlete activity. This noise can interfere with extracting the information of interest and is removed using a low pass filter operating at an appropriate frequency.

**Low Power Filtering.**

For embedded processing, low cost filters are required. While an FIR filter using a Hamming Window requires many floating-point multiplications and additions, a rectangular windowed filter is vastly simplified.
Fig. 4.17 The 99 sample Hamming Windowed FIR Filter. The filter elements are stored in ROM, the sample history is stored in RAM. Each element of the sample history must be multiplied by the corresponding element of the filter and the multiplication results summed to give the filter output. The array pointers must track the array head (or tail) and the midpoint. Multiple filters could be applied concurrently and the outputs combined to generate high-pass, low-pass or band-pass signals as desired.

Fig. 4.18 The Rectangular Windowed FIR Filter. Theoretically this filter operates the same as for the FIR filter except that each element of the filter is identical. This provides a simplified implementation, as the weighting given to each sample is identical. The output can be generated from the previous output by removing a weighted value of the oldest sample and replacing it with a weighted value of the current sample. This reduces floating-point calculations; array index tracking and associated calculations are still required.

Fig. 4.19 First Order Lag. Processing for first order lag requires very little memory and processing is further reduced, as there are no array-pointer manipulations required for generating the filter output (but see text for proviso).

The perceived benefit of reduced memory requirements for 1st Order Lag Based filtering is illusionary. The Lag Filter output must be correctly aligned with the source data in order to handle subsequent processing. This requires that the source data is held in memory. For the FIR filters, the alignment point is the midpoint of the filter, or half the filter length back from the current sample. Considering a weighting factor \( m \) in a Lag based filter is the same as for a Rectangular FIR filter of length \( m \) (in both cases...
the sample is divided by \( m \), it could be assumed that the required offset is \( m/2 \). This is not the case in practice, from Fig. 4.20(a), as the weighting factor increases, the lag increases. When this is offset corrected by \( m/2 \) samples, the output is still delayed (Fig. 4.20(b)). For this sample rate (150Hz), it would appear that the required offset (and therefore array size in memory) is approximately \( m/2 + 15 \) samples. This additional memory cost reduces the benefits of the Lag based low pass filter compared to the Rectangular Window FIR Filter.

![Fig. 4.20 Comparison of Off-Normal Angles generated from 1st Order Lag based Filter Outputs and Hamming Windowed FIR Filter Output (Offset Corrected). (a) As the Lag weighting factor increases, the delay increases. (b) With the Lag based filter outputs offset corrected, the output is still delayed (approximately 0.1s or 15 samples).](image)

**Comparison of Filter Outputs.**

The effectiveness of different filter outputs is determined by the use of the output. This is a function of system type: eg. a low power embedded system performing on-board processing may have different output requirements to a system that requires complex post-processing. Empirical measures of the quality of a filter from a technical analysis of the filter or from a qualitative assessment of the output of the filter may be misleading when applied against the requirements. Different filtering techniques are analysed in subsequent sections and, in particular, an analysis of the effectiveness of complex information extraction.
4.3.3 Orientation of Sensors - Extracting Angle

The orientation of the sensors is concerned with the orientation of the sensors with respect to gravity and hence relates to giving an estimate of the orientation of the athlete to which the sensors are attached. However, unless the sensors are clinically attached (carefully located and firmly attached), this is at best only an approximate orientation. Assuming the sensors exhibit an approximate alignment with the athlete, monitoring the sensor alignment with respect to gravity will give some information regarding the athlete's approximate orientation. By comparing long-term orientation with short-term orientation, various aspects of athlete activity may be discernable. The periods designated long-term and short-term are sport and/or activity specific. An athlete's orientation could be calibrated by conducting some specific exercises, such as standing the athlete against a wall or accelerating them forward on a moving belt.

Only the orientation of the sensors with respect to gravity can be determined. There is no external reference signal that allows the determination of the orientation of the sensor within the horizontal plane. Determination of the orientation within the horizontal plane may be estimated by reviewing the biomechanical activity signals for cross-talk between channels but any apparent cross-talk may itself be an indicator of a biomechanical peculiarity of the specific athlete.

Various 'orientation' types may be required, either individually or in combination, to extract the information of interest. These orientation types are all variations of the same process where the major differentiator is time, or the period over which orientation is estimated. Orientation types include Reference Orientation, Stationary Orientation and Average Orientation (various periods).

The extracted orientation may be used to extract further temporal information by comparison between the instantaneous orientation and the reference orientation. Alternatively, the orientation information may be utilised to rotate the data set to align the nominal 'vertical' accelerometer with gravity.
4.3.3.1 Extracting Orientation

Triaxial accelerometers are mounted on the athlete to align individual accelerometer axes with a normal bodily frame of reference. Eg. The 'vertical' accelerometer aligns with the alignment of the spine, the mediolateral accelerometer is mounted transversely (across the body) and the third is aligned with the normal direction of travel (anterior-posterior) (Fig. 4.21(a)). The orientation of the sensors is read directly from the calibrated triaxial accelerometers. If the athlete is stationary the vector magnitude of the orientation signal will be one gravity. Fig. 4.21 (b) gives an example of extracting the sensor orientation. In this case the readings from the accelerometers are, V=0.91g, ML=0.37g and AP=0.21g. The trigonometrical sum of these values is 1g with the angles resolving to 25 degrees tilt out of vertical, at 30 degrees forward (or back) of the mediolateral axis. Filtering or averaging is used to remove noise from the accelerometer signal prior to estimating orientation.

A reference orientation may be required for some sports. This is, as indicated previously, a calibration of the athlete/sensor combination and may require the athlete to take up one or more reference positions. Alternatively the reference orientation may simply be a selected stationary orientation, for instance, when a footballer is standing upright and still.
Stationary orientation can be determined by trigonometrical analysis of the sensor signal during periods when the athlete is stationary. Stationary periods can be identified where the signal power drops to zero or near zero. In the power temporal density graph of Fig. 4.12, there are long periods where the athlete is stationary, as well as a few periods of inactivity in amongst the main test sequence. The stationary orientation may become a reference orientation, depending on the sport being monitored.

Average Orientation can be estimated from the average value of the signal over periods of dynamic activity. The 'average' orientation includes variations due to the athlete's stance during the activity, eg. an athlete may lean slightly forward while running. Average orientation may change over time, due to the changing location of the sensor relative to the athlete. The average orientation can be estimated from a filtered version of the acceleration data. The filter frequency is determined by the frequency of the dominant activity. In the case of low-speed running, step rates of 2.5Hz are common (Refer Fig. 4.14 and Fig. 4.28 and Chapter 5) therefore the mediolateral channel dominant frequency is 1.25Hz. Using a low pass filter of 1Hz appears sufficient to extract a signal for determining orientation.

4.3.3.2 Rotation of a Data Set.

If it is important to view or analyse a data set with a specific orientation, it may be necessary to rotate the data set. As the data is collected as orthogonal data, rotation of the data from one Frame Of Reference (FOR) to another does not change the underlying form of the data.\(^{31}\)

The primary use for this manipulation is the investigation of biomechanical activity. In the example of Fig. 4.22, an athlete is running at 17km h\(^{-1}\) on a motorised treadmill. The calibrated data from the accelerometers is plotted in two dimensions along with

---

\(^{31}\) Rotation of the data is performed using a rotational tensor or matrix based on Euler Angles (named for Leonhard Euler, 1707-1783.) By identifying the current orientation of the sensors and defining the required orientation, three angles representing rotations about three axes are identified. The rotational tensor is a 3x3 matrix that combines all the trigonometry necessary to rotate a point from one FOR to another. By multiplying all the triaxial accelerometer data by the appropriate rotational tensor, the data can be described within a different FOR while completely preserving the relationships within the data.
the vector representing gravity (extracted using a 1Hz filter). The data set is rotated to align the orientation vector with the vertical accelerometer and replotted.

Fig. 4.22 Raw and rotated 2-dimensional views of dynamic running acceleration. The heavy line represents the orientation of the vertical accelerometer. In the two pre-rotations plots, the vertical accelerometer (and hence all the data) is tilted to one side and leaning forward. After rotation the data set has been corrected for tilt and lean.

### 4.3.3.3 Angle between Reference and Instantaneous Orientation

Estimating the angle between two orientations is performed using a standard trigonometric relationship known as the Cosine Rule (Eqn. 4.1 and Fig. 4.23). If one orientation is a reference orientation, the instantaneous signal angle can be calculated at any time. In the example of Fig. 4.24, the three tackle sequence test data is 1Hz filtered to generate an orientation signal and the angle between this signal and a reference orientation is then generated.

![Diagram](https://via.placeholder.com/150)

**Fig. 4.23** Resolving angles with the cosine rule.
**Equation 4.1  Cosine Rule.** \( a^2 = b^2 + c^2 - 2bc \cos(A) \)

Where (with reference to Fig. 4.23);

- \( a \) = length of side a,
- \( b \) = length of side b,
- \( c \) = length of side c

\( A \) is the angle subtended by sides \( b \) & \( c \).

![Chart](image)

Fig. 4.24 Instantaneous orientation angle (with Power Temporal Density overlay)

Estimating the angle takes considerable processing power. If angle estimation is important for on-board processing, this processing may be simplified depending on application. In Fig. 4.25, the 1Hz filtered **vertical** acceleration has been rescaled and compared directly to the calculated angle of Fig. 4.24. Provided the vertical accelerometer stays reasonably vertical, for some applications the off-normal angle can be estimated directly from the vertical acceleration. Extending this application, the orientation of a prone player, identifying if the player is on their front, back, left or right side, can be directly estimated from the acceleration signal without resorting to complex trigonometric calculations.
4.3.3.4 Visualisation of Orientation Data

Orientation data exists either as a point reference or as a continuum, as is the case with low pass filter output. This data is of little visual interest for many activities but for some, such as the three-tackle test data (and any activity that results in gross orientation shifts), 2 and 3-dimensional representations can be used effectively.

While the graphical representations of the changes in orientation in Fig. 4.26 and Fig. 4.27 are useful in understanding the changes occurring to the athlete, there is sufficient orientation information in the signal to generate an animated representation of the orientation of the athlete.
Fig. 4.26 2-Dimensional views of the orientation signal during the 3-tackle sequence. The origin is marked with a cross. From the origin a line 1g in length is drawn indicating the normal (vertical) orientation of the sensor. A second line 1g in length goes from the origin to the point where the orientation was mathematically furthest from the normal vertical orientation.

![Rotated Low Pass (1Hz) Filter Output](image)

Fig. 4.27 3-Dimensional view of the orientation data during the 3-tackle sequence. The three tackles appear as three looping curves. During the tackles the orientation changed from vertical to near horizontal as shown by the two straight lines.

### 4.3.3.5 Orientation and Angle Summary

The orientation of the triaxial accelerometer (and by implication the athlete) with respect to gravity can be derived trigonometrically. Orientation can be extracted from static and dynamic data and the angle between various orientations derived. Extracted orientation data can be used to rotate the acceleration data to align the triaxial sensor frame-of-reference with gravity. For some applications, the value of the low pass filtered vertical acceleration signal is a sufficient estimator of angle.
4.3.4 Power Spectral Density (FFT)

Identifying the frequencies present in data is an assistive technique that can be used to identify the differences in the signals across orthogonal sensors or to obtain an approximate starting point for selecting filtering frequencies. Because of the limitations outlined above (Section 4.2.2) the FFT output lacks the precision necessary for some applications. Lack of frequency resolution also prevents meaningful interpretation of the FFT output magnitude. In Fig. 4.28, FFT is used to analyse an athlete running at 7 different speeds, the first plots are generated using 1024 samples (6.8 seconds of data @ 150Hz sample rate) and the second group using 2048 samples (13.7s). Not only are extracted frequencies different (depending on the FFT sample length), but the relative magnitudes are also affected by the FFT sample length.

![Fig. 4.28](image)

**Fig. 4.28** Comparison of multispeed FFT analysis using different FFT sample lengths.

The Power Spectral Density is a useful tool for comparing the signal across the three orthogonal sensors as well as comparing data between athletes. Fig. 4.29 shows both these scenarios, the data from orthogonal triaxial accelerometers from two athletes running at 21kmh\(^{-1}\) on a motorised treadmill is compared. For both athletes, the vertical axis is dominant with the signal centred on the step frequency. The mediolateral axis has most of the signal centred on the stride frequency (half the step frequency) but this signal is much smaller than the vertical signal. Both athletes have anterior-posterior signal at the step frequency and at a side-band or harmonic frequency. The first athlete has a larger signal at the harmonic, where the second athlete has the larger signal at the fundamental (step frequency). This form of analysis...
and comparison may be useful, in conjunction with other forms of analysis, in determining various parameters of biomechanical activity. For instance, this FFT analysis may indicate that athlete 1 does not have as much anterior-posterior movement as athlete 2, and/or that they have more 'core-stability' than athlete 2 (indicated by fewer and smaller sidebands). Conversely the FFT comparisons may indicate that athlete 2 moves backward and forward on each step (relative to the treadmill frame) and/or has less core-stability. In this context 'core-stability' refers to the strength and symmetry of muscular activity such that the hips are firm anchors for both upper body activity and leg functioning.

Fig. 4.29 Power Spectral Density - cross channel and inter athlete comparison for running at 21kmh⁻¹.

4.3.4.2 FFT Summary

The use of Power Spectral Density, in conjunction with other forms of analysis, may be a useful tool in analysing athlete biomechanical activity. The limitations of the FFT producing the Power Spectral Density must be considered. FFT is not an accurate estimator of frequency of either short bursts of activity or longer-term variable-frequency activity. Even a comparison of relative magnitudes of signals may be affected by the lack of precision in frequency identification.
4.3.5 Power Temporal Density

4.3.5.1 Generating a Power Temporal Density.

A Power Temporal Density should indicate the power in the signal at a particular instant of time. There are a number of possible alternative methods of generating this signal. If it is assumed that a signal is composed of a dynamic component (such as the running signal) and a static component (such as an orientation signal) then dynamic power could be calibrated from the high-pass (dynamic) content. This may be an acceptable approximation for running but, as Fig. 4.13 shows, it is not true for complex activities such as tackles where substantial power exists in the low pass signal. If the assumption used to generate the one-gravity length orientation vector (previous section above) has a basis in physics, then subtracting a one-gravity length orientation vector should leave pure dynamic data for generating a power spectrum. A third and simpler method is to generate the magnitude of the acceleration vector and then subtract one-gravity. The resultant data is then used in the generation of the Power Temporal Density. This method also uses assumptions that are not valid.

Without knowing the instantaneous orientation of the sensors, no method can provide an accurate estimate of the dynamic signal power. The shortcomings of each method are discussed below, example outputs from each method appear in Fig. 4.31. The outputs are generated by averaging the square of the acceleration magnitude over 0.25 seconds.

1. Estimating dynamic signal power by taking the high pass (1Hz) data.

This signal is generated by subtracting the low pass filter output from the calibrated acceleration signal. While the low pass signal may give the average orientation of the sensors, during each stride there are continuous changes in the dynamic orientation. If the situation depicted in Fig. 4.30 is applied in this scenario, the applied force of 1.05g @ 85 degrees is correctly detected and measured by the sensors. When the 1Hz low pass output (average orientation) is subtracted from the dynamic signal the result does not relate to the actual situation. If (for this example) the average orientation has the vertical accelerometer positioned vertically, the equation [Dynamic Values] - [Orientation Values] = [0.681(V) 0.803(H)] - [1.0(V) 0.0(H)] = [-0.319(V) 0.803(H)],
giving a vector magnitude of 0.86g (instead of 0.1g), an error of 760%. The output from this method appears in comparison with the output from other techniques in Fig. 4.31.

(2) Estimating dynamic signal power by subtracting a magnitude corrected orientation signal from the acceleration signal. This method is a straightforward modification of the previous method. In this case the low pass data is still assumed to contain orientation data. As Fig. 4.13 indicated, there is power in the low pass data. By maintaining the orientation of the low pass signal but fixing the magnitude at one gravity this signal leakage is returned to the dynamic signal. This suffers from the same errors as the previous method but includes additional errors. In the test data used in this example, the athlete undergoes several sharp impacts. While the impacts are of short duration they are of high intensity and spread spectrally through the signal. Because of the spectral spreading they appear in the low pass data incorrectly as orientation, therefore distorting the orientation signal.

(3) Estimating dynamic signal power by subtracting one gravity from dynamic signal vector magnitude. No matter what orientation the sensors have, the measured vector magnitude is the trigonometrical vector sum of gravity plus applied acceleration (force). In the example of Fig. 4.30, the resultant vector has a magnitude of 1.05g. Subtracting one-gravity leaves a magnitude of 0.05g, representing an applied acceleration of 0.1g, or an error of 50% in this example. The amount of error is a function of the angle of the applied force. If the applied force is in the same direction as gravity the error is zero.

![Diagram of acceleration vectors](image-url)
4.3.5.2 Period of Signal Averaging

Using the Power Temporal Density for generating energy expenditure usually uses the sum of the square of the acceleration signal counted over some finite period. The period is determined by the required usage. Typically, per-second counts would be stored and later analysed to determine total activity at a range of activity levels, periods of activity and inactivity, etc. When monitoring athletes such as football players, these factors are important for physiological monitoring. For activity identification some of the important content exists as very small detail. Using the longer period, the graphical representation of the signal in Fig. 4.32 (same period of activity as for Fig. 4.31) shows this lack of detail. The abrupt change from impact to stationary has been smoothed over (small detail lost) and the relative magnitude of the impact and running signal has changed.
4.3.5.3 Power Temporal Density Summary.

In the absence of complete knowledge of the sensor orientation, any estimate of dynamic signal power is subject to unknown error. From Fig. 4.31, it would appear that, despite the ability of the orientation-based methods to vastly overestimate the signal power, and the magnitude-only method to under estimate the signal power, all methods give comparative results. As would be expected the two variations of the orientation-based method have very similar results except at the moment of impact. In practice these dissimilar approaches have similar outputs for two reasons; (1) the predominate running signal is vertical (particularly if the athlete is running at a constant speed) and (2) the dynamic orientation does not usually change by as much as is given in the example of Fig. 4.21.

The magnitude-only method offers two benefits compared to the other methods, it is simpler to process and it better represents the points when the athlete is momentarily stationary (as in Fig. 4.31 & Fig. 4.12). This is important when attempting to ascertain activity type. The duration of the 'counts' period is determined by the needs of the application. Some applications may require different periods to detect and differentiate between impact and steady state signals. While extracting orientation is not necessary for estimating signal power, it is still important for other purposes.

4.3.6 Autocorrelation

Autocorrelation is a useful technique for extracting a regular signal in the presence of noise or for detecting relationships in apparently random signals. As the accelerometer
signal from an athlete engaged in sporting activity sometimes appears as a regular signal with considerable noise, the use of autocorrelation may identify some useful parameters. Moe-Nilssen & Helbostad [16] combine the autocorrelation of accelerometry data with timed walks over set distances to estimate parameters such as speed, step length, cadence (step frequency) and symmetry. This is primarily used for people with injuries as a method of comparing data from different cadences and speeds.

4.3.6.1 Detecting Symmetry and Consistent Cadence

Applying autocorrelation to test data from a group of 10 athletes walking and running at ten speeds provides some assessment of the gait characteristics. If an athlete has not settled at a fixed step frequency, the envelope containing the autocorrelation output shows a distinct decay (as in Fig. 4.33(a)). Despite the decaying envelope of Fig. 4.33(a), the autocorrelation output shows a typical symmetrical ML signal, unlike the asymmetrical ML signal in Fig. 4.33(b) where the left-step / right-step components are distinctly different. The fundamental period of the ML signal is the stride period while the fundamental period of the Vertical and AP signal is based on the step period. The asymmetry of an AP signal appears in Fig. 4.33(c) where the left and right steps have distinctly different magnitudes.

Fig. 4.33  Autocorrelation Output (a) Decaying envelope, (b) ML Asymmetry, (c) AP Asymmetry
4.3.6.2 Detecting Speed Related Changes

Comparing the autocorrelation coefficients from different running speeds may provide some insight into the changing biomechanical action as speed increases. In the example of Fig. 4.34, the autocorrelation coefficients were plotted along the x-axis as a function of the step period. This allows the direct comparison of the autocorrelation coefficients from different step frequencies.

![Comparison of autocorrelation coefficients from different running speeds.](image)

In this case, a distinctive speed dependent change can be observed and therefore this application may be beneficial.

4.3.6.3 Extracting step frequency from noisy signals.

For some subjects, autocorrelation can detect step frequency at low walking speeds where visual pattern recognition, zero crossing detection and in some cases, FFT, fail to detect the step frequency. Raw acceleration and resultant autocorrelation coefficients for an athlete at very low walking speed (3kmh⁻¹) are compared in Fig. 4.35, the FFT output for the same period appears in Fig. 4.36.

![FFT output for the same period.](image)

Noting that the normal channel for extracting step frequency is the vertical channel, the complexity of the walking signal is such that step detection directly from the acceleration signal, the autocorrelation output or the FFT all fail. In this particular example, the AP channel is the best indicator of step frequency, using both autocorrelation and FFT.
Fig. 4.35 Comparison of raw acceleration signal and resultant autocorrelation coefficients for low speed walking.

In this example, the complexity of the signal, the small amplitude of the signal, the noise on the signal and the very low frequency (relative to other ambulatory activities) make filtering problematic. Provided the step frequency is consistent over the period under test, the autocorrelation process assists in identifying the existence of a consistent ambulatory gait.

Fig. 4.36 Power Spectral Density (FFT) of low speed (3km\(^{-1}\)) walking signal.

Autocorrelation appears useful, in this case, for extracting low amplitude repetitious signals in the presence of noise.
4.3.6.4 Autocorrelation Summary

Autocorrelation of acceleration data may assist in determining some aspects of the biomechanics of athlete gait. In particular, identifying the consistency of the step frequency and the left-right symmetry. Comparing the output of the autocorrelation process at different running speeds may also be advantageous. The existence of low intensity repetitive activity can be determined using autocorrelation, this is not particularly useful in the case of field athletes, but may be useful when using trunk-mounted accelerometers in monitoring cyclists. Autocorrelation is limited in that human movement, even in trained athletes, is not as precise or consistent as repetitive mechanical or electrical signals. It is also not immediately obvious how the shape of the autocorrelation output relates to the movement of the athlete. Inter-channel cross-correlation is a possible investigative technique however it produces results that are less useful than autocorrelation.

4.3.7 Gait Cycle Normalisation of Acceleration Signal

In the context of this discussion, Gait Cycle identifies one complete stride or a complete left/right (or right/left) step combination.

Various difficulties exist when attempting to categorise the acceleration signals from the trunk-mounted accelerometers. Which component of the signal is a parameter defining the gait of a particular individual? Is the rise time of the vertical acceleration a function of the athlete's gait or a function of system filtering? Which humps and bumps of the acceleration signal are gait related and which are simply noise? Of those that are gait related, what aspect of the gait do they represent?

Some of these questions may be answered with straight empirical measures, such as using anterior-posterior peak-to-peak acceleration as an indicator of the braking and re-acceleration occurring on each step. Other questions have no distinct empirical answer.

To gain a better understanding of the components of the gait cycle, merging the acceleration data from a number of gait cycles can be used to obtain a representative gait cycle. This removes many transient noise signals and cycle-to-cycle irregularity. The representative gait cycle acceleration signal from one step-frequency can now be
normalised in the time domain and directly compared with representative signals from other step-frequencies. This combination signal reveals the athlete's underlying gait-cycle signal. The importance of comparing the representative signals, as opposed to comparing any acceleration sample, is that the random inter-step fluctuations are sufficient to hide any inter-speed commonality.

This process is also applicable to any cyclical athletic activity, such as rowing, etc.

4.3.7.1 Representative Gait Cycle Processing

Assuming the gait cycles are distinctive signals, the accelerometer data is filtered first to remove high-frequency noise and then filtered to remove the offset due to gravity. This results in triaxial acceleration signals oscillating about zero. As the vertical acceleration typically has a sharp rise time during brisk walking and running, the negative to positive zero crossing of the vertical acceleration is used as the cycle marker. To align the left-right step combinations, the alternating phase of the mediolateral signal is used to detect matching step pairs. Due to the coarseness of the zero-crossing identification at low sample rates (100-150Hz), the data is interpolated, using linear interpolation, to a much higher sample rate.

Using this processing, step-pairs in the data can be automatically detected and overlaid to create a visual representative step. The graph in Fig. 4.37 is a result of averaging pairs of successive steps. Inspection of the bottom of the curves shows that the step-pairs have been correctly aligned as the asymmetry exhibited between left and right steps is preserved. Note that even within the short sequence of six running steps, noticeable variation in the period or duration of a step exists as one of the sequences moves away from the grouping and then merges back in.
Fig. 4.37 Creating a representative gait cycle signal.

To generate the above figure, noise is filtered using a 10.5Hz Hamming Windowed FIR filter, gravity is removed using a 1Hz filter and the 150Hz-sampled data is interpolated to 450Hz. To differentiate between left and right steps, the area under the mediolateral curve is summed for successive vertical channel positive half-cycles as in Fig. 4.38. For virtually all athletes, the mediolateral summation delivers values that clearly distinguish between the left and right step. Which step (left or right) gives the greater value depends on the orientation of the sensor.

Fig. 4.38 Processing mediolateral channel data to distinguish between left and right steps. The integral of the shaded regions is positive for one step and negative for the alternate.
4.3.7.2 Normalising the Representative Gait Cycle

Normalising the Gait Cycle refers to the process of normalising all Gait Cycles against their own stride period. This results in the set of representative Gait Cycle data being measured in the 'Gait' domain rather than the time domain. In the following graphs, one complete Gait Cycle (left/right step combination) is represented as 360 degrees. The Gait Cycle data requires normalisation to allow the direct comparison of accelerometer data from different running speeds and step frequencies. As shown in Fig. 4.39, it is difficult to directly compare signals from different step frequencies. In Fig. 4.40, the same acceleration signals that appeared in Fig. 4.39 have been Gait Cycle normalised and the common gait characteristics begin to appear.

Fig. 4.39  Comparison of AP acceleration from different running speeds.

Fig. 4.40  Comparison of Gait Cycle normalised AP acceleration from different running speeds.
4.3.7.3 Gait Cycle Normalisation Summary

The ability to extract a signal pattern that is representative of a succession of similar patterns, and then compare patterns produced by the same athlete at different activity speeds, appears to have some potential for use as a tool for investigating biomechanical activity. This technique is not as robust as autocorrelation but preserves the acceleration shape and inter-channel relationships. The use of this technique as a method of categorising running biomechanics is explored in Chapter-6.

4.3.8 Using Integration to Extract Velocity and Displacement

As noted in Section 4.2.2 Factors and Limitations, under the title Acceleration, Velocity and Displacement, while the extraction of across-the-ground velocity and displacement for field athletes is desirable, it is not feasible. Slight misalignments in conjunction with noise, manufacturing error and calibration error could result in an athlete appearing to reach unrealistic velocity magnitudes over the duration of a football match. Filtering which is capable of removing offset errors due to misalignment will also remove the relatively small and short lateral accelerations that result in the lateral velocity and displacement. Using orientation information to constantly keep the sensor output aligned with gravity also will not generate accurate results. The short bursts of acceleration that bring an athlete up to running speed, also feed into the orientation information, resulting in a distorted orientation signal. Constant resetting may assist in estimating velocity and displacement. In this case, whenever a field athlete's activity drops below a threshold, it can be assumed that the athlete is stationary - therefore velocity is zero, therefore displacement does not change.

While the extraction of lateral velocity and positional information may not feasible, it is still possible to extract the in-stride dynamic velocity and displacement (Fig. 4.41). This processing, in conjunction with other processing may assist in defining biomechanical activity.
To generate this data, the original acceleration data filtered to remove noise (15Hz) and orientation (1Hz), then the resultant signal is integrated to generate velocity. Typically the velocity data will require high-pass filtering (1Hz) to remove a growing offset. This data is then integrated again to generate displacement. This signal may also require high-pass filtering (1Hz) to remove the increasing offset. After this processing, the acceleration, velocity and displacement data sets are stationary about the origin and can be graphed.

In isolation these graphs don't appear to have much value in developing an understanding of the biomechanical action, however, in conjunction with a number of the other processing techniques, they can provide assistance in interpreting the accelerometer data. Because this is an oscillatory signal, the graphs have a phase relationship, where the acceleration and displacement are separated by 180 degrees and the velocity lies between them. An acceleration maximum equates to a velocity zero and a displacement minimum.

### 4.3.9 Combining External Reference Data with Accelerometry

The in-sole sensor developed by Billing et al. as part of the 'Athlete Tracking' projects was used to synchronously collect in-sole forces and centre-of-mass accelerometer data. This data was primarily used to develop a wearable 'force-plate' where the primary interest was the foot contact forces [17]. In that research the accelerometer data augmented the in-sole sensor data. In the study of the accelerometer data, the foot contact information was used to provide absolute external reference points. This data was linked to the time series acceleration series as in Fig. 4.8 and Fig. 4.42 and combined with the 3D views (Fig. 4.43).
In the above figure, the toe-off (last foot contact) occurs approximately at the point that the vertical acceleration is equivalent to gravity, indicating zero athlete input at that point. The mediolateral and anterior-posterior acceleration is also close to zero input. At the point of initial foot contact all the acceleration signals are approaching zero. This is the last moment of free-fall. These two key points are at best only approximate and the acceleration data is not sufficiently consistent to extract contact or flight timing.

The unmarked 3D velocity graph of Fig. 4.43(a) has enhanced value when the markers were added (Fig. 4.43(b)). The components that represent contact phases are now differentiated from the flight phases. With the plot rotated to obtain a side view (Fig. 4.43(c)).
4.43(c)), the initial contact was observed to occur prior to maximum negative velocity, the trajectory of increasing velocity follows a straight line until toe-off occurs approximately at peak velocity. The ability to analyse other figures is similarly enhanced. The use of reference points on some figures enhances the ability to orientate and interpret figures without the foot contact reference data.

These aspects are further explored in Chapter 6.

4.3.10 Miscellaneous Processing

This section deals with various other minor processes used in analysing the data gathered from accelerometers.

4.3.10.1 Basic and Interpolated Zero Crossing Detection

For low power embedded sensor systems, step and step-frequency detection can be performed through implementing a zero-crossing detector operating on the vertical axis. Steps are detected as the interval between two consecutive positive-going transitions of the vertical acceleration. Step frequency is estimated from the integer count of samples between transitions. During brisk walking or running the vertical signal is sharp and false positives due to noise are limited. Various signal processing or software programming techniques can be used to further reduce the likelihood of false positives. During low intensity activity the step detector is disabled due to both the high noise to signal ratio and the ambiguity of the step patterns.

Digital sampling systems operate asynchronously in relation to monitored analogue systems (such as athletes) and therefore samples do not necessarily coincide with a specific point of interest (as in Fig. 4.44). On a 150Hz sample-rate, a one-sample error in timing results in an error of 0.007 seconds, converting a 3Hz step frequency to 2.94Hz or 3.06Hz. When the sample rate reduces to 15Hz, a one-sample timing error approaches 0.07 seconds, converting a 3Hz step frequency to 2.5Hz or 3.75Hz. The resultant step-frequency can oscillate between these values but over a period will give an average that is approximately correct.
Because the timing jitter increases as the sample rate reduces and the number of samples between zero crossing is small, estimating the step frequency from the integer count of samples between zero crossing results in large step changes in the detectable step frequency. At the processing cost of performing a small number of floating-point calculations, estimating the step frequency using a linear interpolation of the zero crossing can produce results similar to those obtained from a much higher sampling rate. The comparisons of Fig. 4.45 are based on the raw estimated step frequencies. The timing jitter that results in one step being measured as one sample longer will subsequently cause the following step to appear one sample shorter. Thus the step frequency estimation is improved if the estimated step frequency is averaged with the raw estimate from the previous step. This also assists where there is small inter-step biomechanical variability. Using two-step averages would improve the integer-based estimate in Fig. 4.45(b) but this is a function of the step frequency in the example being approximately in the middle of two distinguishable levels (2.5 & 3Hz). If the rate were closer to either level, estimating using an average would require more than two successive steps.
Fig. 4.45  Step Frequency Estimation: Comparing 15Hz linear interpolation with (a) 150Hz estimates and (b) 15Hz integer based estimates.

The interpolated step frequency estimates in Fig. 4.45(a) & (b) appear to occur slightly earlier in time than the integer based estimates. The example of Fig. 4.44 indicates that for some combinations of signal and sampling rate, the interpolated estimate of the positive zero crossing at a low sampling rate will precede the integer based estimate associated with a higher sampling rate. This is irrelevant to the actual results.

4.3.10.2  Trimming Orientation Signals at One Gravity

The use of filters to extract average sensor orientation assumes the signal is predominately high frequency dynamic signal superimposed on a static, or slowly changing orientation signal. For high intensity activities, such as running, the power in the dynamic signal component is several orders of magnitude larger than the power in the extracted orientation component. For the signal power comparison of Fig. 4.46, the orientation signal was extracted from the total signal using a Hamming Windowed FIR digital filter with a cut-off of 1.5Hz and a filter length of 4 seconds.
The cut-off frequency was chosen as the highest frequency below any significant spectral component identified in a multi-channel Power Spectral Density (Fig. 4.47) at the lowest step frequency for this test data.

Although the filter frequency is very close to the activity frequency, the power in the orientation signal is still less than 0.3% of the power in the dynamic component. The vector magnitude of the triaxial low-pass filter output remained close to the expected value of 1g. The signal was not consistently 1g due to the constant small fluctuations in runner's action, such as drifting very slightly to one side or the other over the space of several steps.
This result appears to confirm the use of low-pass filtering to extract orientation in certain circumstances. This no longer applies when contact sports are involved. The impact forces in contact sports can be very large and they result in high-power broad-spectrum signals. In this case, there is considerable signal leakage into what is considered the orientation signal. The 1Hz low pass output from the three-tackle test sequence (Fig. 4.13) includes a peak-to-peak acceleration signal approaching \(7\text{ms}^{-2}\).

There are a number of consequences of this high level signal in the orientation signal. (1) Estimates of power temporal density using dynamic signal assume no leakage. (2) An orientation signal implies or assumes a one-gravity magnitude and the orientation should be mathematically describable as a point somewhere on the surface of a unit (1g) sphere. (3) The orientation signal is corrupted and represents a combination of orientation (gravity) and an additional force at an unknown angle.

For this form of activity, a filter frequency of 0.8Hz to 1Hz appears effective in extracting a signal that represents the changes in the athlete's orientation. Increasing the filter frequency causes more leakage of activity signal into the orientation signal,
reducing the frequency results in a signal that doesn't track the fast orientation changes (this is a qualitative assessment).

The assumption of point (2) could be enforced by mathematically fixing the orientation vector at a length of 1g. This will ensure that there is zero signal power in the orientation signal and that all orientation is represented by a coordinate on the unit-gravity sphere.

To extract a fixed length orientation signal it is assumed that the 1Hz signal is in the general direction of the orientation of the sensor. This is a false assumption, but is used as a method to remove power from the orientation signal. The orientation signal is processed and; where it is longer than one gravity in magnitude it is truncated, where it is shorter than one gravity in magnitude it is lengthened. The magnitude is fixed at one gravity but the orientation of the signal is preserved. The new orientation vector is graphed in multi-dimensional views in Fig. 4.48.

---

**Fig. 4.48** Views of the orientation vector. In the first figure, the three-dimensional view has been rotated to only display the ML and V axes. Dotted lines indicate the relationship of the signal to the origin. In the second figure, the fixed length of the orientation vector causes it to scribe arcs on the surface of a sphere (all lines on these graphs are on the surface of a sphere).
4.3.10.3 Cross Channel Comparisons

Various cross channel comparisons can be performed to derive information on the activity of the athlete. The cross-channel Power Spectral Density examples of Fig. 4.14, Fig. 4.29, Fig. 4.36, and Fig. 4.47, each indicate some factor related to the gait of the athlete. Comparing Power Temporal Density across channels also offers some information. In the case of field sport athletes engaged in contact sports, the relative magnitudes of the signal on the horizontal axes can be used to give an indication of the direction of an impact force. An athlete swerving sharply or changing direction at high speed generates detectable changes across the horizontal axes.

4.3.11 Interdependencies of Processing Techniques and Applications

The processing techniques discussed in this section have a range of interdependencies. The particular implementation is application dependent with desktop processing targeting complex analysis such as gait identification through expensive filtering and processing, with low-power embedded systems extracting activity rates, activity intensity and sensor orientation using simplified processing.

Table 4.1 Signal Processing Techniques and Application

<table>
<thead>
<tr>
<th>Processing</th>
<th>Desktop Processing</th>
<th>Embedded Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>High Power</td>
</tr>
<tr>
<td>Hamming Window FIR Filter</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Rectangular Window FIR Filter</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>1st Order Lag based Filter</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Orientation Extraction</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Off-Normal Angle Estimation</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Data Set Rotation</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Auto/Cross correlation</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Gait Cycle Normalisation</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Power Spectral Density (FFT)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Power Temporal Density</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Zero Crossing Rate Extraction</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Pattern Recognition</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Cross Channel Power</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Combining with in-sole data</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

* In-sole data could substitute for accelerometer data for step detection.
Table 4.2 Interdependencies of Signal Processing Steps

Interdependencies of Processing Techniques.

**Example:** FFT will work perfectly well using raw acceleration data.

**Example:** Orientation extraction requires filtered data and the preferred source would be an appropriate Hamming FIR Filter.

<table>
<thead>
<tr>
<th>Subsequent Processing Phase</th>
<th>Raw Data</th>
<th>Calibrated Data</th>
<th>Hamming Window FIR Filter</th>
<th>Rectangular Window FIR Filter</th>
<th>1st Order Lag based Filter</th>
<th>Orientation Extraction</th>
<th>Off-Normal Angle Estimation</th>
<th>Data Set Rotation</th>
<th>Auto/Cross correlation</th>
<th>Gait Cycle Normalisation</th>
<th>Power Spectral Density (FFT)</th>
<th>Power Temporal Density</th>
<th>Zero Crossing Rate Extraction</th>
<th>Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hamming Window FIR Filter</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rectangular Window FIR Filter</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order Lag based Filter</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orientation Extraction</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Off-Normal Angle Estimation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Set Rotation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto/Cross correlation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gait Cycle Normalisation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Spectral Density (FFT)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Temporal Density</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zero Crossing Rate Extraction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pattern Recognition</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Energy Expenditure Estimates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2D &amp; 3D Visualisations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>In-stride Velocity &amp; Displacement</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X indicated dependency of the row-process on the output of the column-process.

1, 2 & 3 Indicate that the row-process can utilise output from any of the column-processes but the preference is in the order 1st, 2nd, 3rd. Actual choice is determined by the required output, preference ordering is by most-correct to least-correct output from a technical perspective.
4.3.12 Example System - Embedded Football Athlete Monitoring.

An athlete monitoring system would be a multi-purpose device capable of operating in a number of different modes. Each mode would monitor and categorise activity based on the assumption that the appropriate mode has been selected. The output would be in a format determined by the mode. Some possible modes of operation:

- Raw Data Output (logged to memory)
- Swimming Mode (detect stroke, time and count laps and monitor stroke rate)
- Jogging Mode (count steps, estimate energy expenditure and distance, monitor session length)
- Football Mode (estimate energy expenditure & distance, track step rates, orientation, tackles, hit-ups etc).
- Other Modes as required.

In this example, the system required second-by-second summary data for a number of parameters. The output format and signal processing latency are not a concern in this discussion. Activity of interest includes:

- Step Counts.
- Step Rates for steps detected.
- Average activity intensity for the current period.
- Orientation
- Athlete and ground impacts and probable tackles.

The extraction of orientation immediately indicates the need for low pass filtering. Step detection using zero-crossing detection requires the high pass filtering of the vertical acceleration. Activity intensity requires the calculation of the Power Temporal Density. Since the system has no knowledge of who is carrying the ball, tackle detection should be defined. In this case, impact followed by significant change in orientation and momentary inactivity would be sufficient. Impacts can be defined as signal power substantially above the normal running level. At this point the duration of an impact is unknown.

A model of the processing used to generate the outputs for desktop processing appears in Fig. 4.49, in this case, Hamming Window FIR filters are used to extract orientation, produce the high-pass vertical acceleration signal and to remove noise from the
vertical acceleration to reduce false triggering of the step detection. The embedded version, with simplifications and approximate processing costs (as clock cycles) appears in Fig. 4.50.

Fig. 4.49 Desktop Processing for Football Athlete Monitoring. The output from the various processing steps can be used directly or fed to a lookup table to detect likely tackle events (combination of high-intensity and high angle followed by very low intensity and high angle). Step-detection uses activity intensity as an off switch during low intensity activity.

Fig. 4.50 Embedded Processing for Rugby-football Athlete Monitoring (with processing cycles ~x). Overall sample rate is lowered, simplified filtering for orientation and step detection. For a minimalist system, the system can function with only the vertical channel, some orientation data is lost and the activity intensity has a lower correlation with energy expenditure. Alternate processing for the Power Spectral Density reduces cost by eliminating the Square-Root calculation.
Overall Processing
The embedded system algorithms are performed entirely in integer arithmetic, predominately using 8-bit and 16-bit values. Data is not converted to SI units as this causes a loss of resolution. The 10-bit sampled calibrated accelerometer data can be held in signed 8-bit integer variables which limits the maximum acceleration to approximately ±2g or ±4g if the least significant bit is dropped. Chapter 7 discusses the value of the various bits to the quality of the signal. The important factor is the available resolution and its impact on integer calculations. For the ADXL202 accelerometer, read by a 10-bit ADC, the resolution for 3V operation is 59 units per g.

Step Frequency Calculation
As previously indicated (Section 4.3.10.1), step frequency at low sample rates can be better approximated using linear interpolation. Restricting calculations to integer arithmetic does not interfere with this process. Using 8-bit unsigned integers, step-frequencies from 0 to 6.4Hz can be represented to 0.025Hz accuracy.

Power Temporal Density
Section 4.3.5 discusses the generation of power temporal density and the use of the square of the acceleration signal. For daily activities, simply calculating the square of the signal and subtracting the square of gravity is sufficient as an estimator of signal power. For rugby football, orientation at instances of inactivity is of interest. This requires a better performing signal power estimator. Poor calibration of triaxial systems can result in the appearance of large error components during changes in orientation (see Fig. 4.5). Small signals are also magnified due to the presence of the gravity signal eg. Signal = 10ms$^{-2}$ comprising gravity (9.8ms$^{-2}$) and athlete signal (0.2ms$^2$). $\text{Signal}^2-\text{gravity}^2=3.96(\text{ms}^2)^2$ but $\text{athlete \_signal}^2 = 0.04(\text{ms}^2)^2$. Using a simple estimator, small errors are magnified preventing correct detection of inactivity. For a system trying to correctly detect inactivity during changes in orientation the square root function is required. Output from both methods is compared in Fig. 4.51.

Square Root Function
An integer square-root function can be performed in a number of ways: (1) iterative root finding (Newton-Raphson), (2) look-up tables and (3) manipulation using logical shifts and adds. There are variations of the each of the above methods. The Newton-
Raphson method requires multiplication and division, which are costly, but the method can converge quickly if the starting point is a good approximation. Considering that the maximum value of the vector magnitude is $3 \times 128^2$, or 49152, the square root is limited to a range of 0-223. A binary search of a look-up table can converge in 7 iterations. Both the look-up table and logical manipulation can obtain a result in less than 256 clock cycles on the Hitachi H8 processor.

Table 4.3 Processing Cost to Generate Outputs (Hitachi H8 Processor)

<table>
<thead>
<tr>
<th>Process</th>
<th>Cost (processor cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Accelerometers</td>
<td>18</td>
</tr>
<tr>
<td>Subtract Offset</td>
<td>18</td>
</tr>
<tr>
<td>Calibrate 0, 1 or 2 channels</td>
<td>0, 40 or 80</td>
</tr>
<tr>
<td>Create Vector$^2$ (ie. $a^2+b^2+c^2$)</td>
<td>60</td>
</tr>
<tr>
<td>Square Root.</td>
<td>256</td>
</tr>
<tr>
<td>Subtract gravity</td>
<td>4</td>
</tr>
<tr>
<td>Square result.</td>
<td>20</td>
</tr>
<tr>
<td>Orientation Filter + array pointer manipulation</td>
<td>72+8</td>
</tr>
<tr>
<td>Impact Filter + array pointer manipulation</td>
<td>24+14</td>
</tr>
<tr>
<td>Activity Intensity Filter + array pointer manipulation</td>
<td>24+10</td>
</tr>
<tr>
<td>Step Detection</td>
<td>120</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>728</strong></td>
</tr>
</tbody>
</table>

For a 1MHz processor, 25Hz sampling @ 728cycles per sampling equates to 1.82% of available processing cycles, with the square-root function contributing about one-third of the load.

**Output:**

As a measure of the effectiveness of the proposed embedded processing, output of integer-only processing, 25Hz sampled data - with simplified filtering is compared with the same output from 150Hz sampled data where the processing uses floating point calculations and complex filters. Figures on the following pages indicate that the outputs generated from the embedded system processing closely match the outputs from desktop processing. Processing cost estimates from Table 4.3 indicate that the proposed system can be implemented without exceeding the available processing power. In this application, the processor clock speed could be reduced.
Fig. 4.51  Comparison of two methods of generating Power Temporal Density. The first utilises a square root calculation, the second avoids the square root calculation. Generally the shapes are similar but the second method suffers by increasing the error at low signal levels (Compare the marked areas on the two graphs). Both graphs generated from 25Hz integer only calculations with averaging over 0.166s.

Fig. 4.52  Comparison of Power Temporal Density (0.33s averaging) for 150Hz floating-point data vs. 25Hz integer-only calculations. At this resolution the traces are almost exactly overlaid and there is virtually no discernable difference in the 25Hz and 150Hz sample-rate data. A close up of the peak at 20.3 seconds appears in Fig. 4.53 below.

Fig. 4.53  Close up of Power Temporal Density for 150Hz vs. 25Hz from Fig. 4.52 above. 25Hz sampling with integer calculations does not appear to be detrimental to the output.
Fig. 4.54 Comparison of Angles generated from 150Hz orientation signal extracted with 1Hz Hamming Windowed FIR filter (300 samples long) and orientation extracted from 25Hz data using 21 sample rectangular window FIR filter. Angles align closely; the 25Hz data has more noise at low angle due to simplified filtering.

## 4.3.13 Processing Summary

A number of processing techniques were discussed. These techniques fell into interdependent groups with some processes appropriate for desktop processing and others appropriate to embedded systems. Some techniques, such as cross-channel FFT comparisons, autocorrelation, gait-cycle normalisation, in-stride velocity and in-stride displacement estimation can be combined for analysis of biomechanical activity. These processing techniques would be supported by FIR filtering, orientation extraction and data set rotation. The use of these techniques for categorising running biomechanics is discussed in Chapter 6.

Different techniques for the extraction of the Power Temporal Density were discussed in the context of a particular application. The validity of the Power Temporal Density as an estimator of energy expenditure was not covered in this chapter (see Chapter 5).

Processing for a low-power embedded system application was investigated and the outputs compared to results from desktop processing. The embedded system processing was viable both in terms of quality of output and processing load.
4.4 References


[5] Lee S.W., Mase K., 2001, Incremental Motion-Based Location Recognition, Fifth International Symposium on Wearable Computers, pages 123-130, Zurich, Switzerland, October.


5 Estimated Athlete Energy Expenditure

5.1 Introduction
This chapter deals with estimating athlete energy expenditure (EE) using accelerometers. It starts with a general introduction into the concepts and some of the techniques currently in use. It reports the history and basis for using accelerometers as estimators of EE and identifies some of the limitations. The design, conduct and results from several experiments are reported followed by a proposed accelerometer based EE estimation technique. The key points of this chapter are collated in the summary.

5.2 Background
Athletes undergo long periods of training with the intention of improving their competition performance. Training focuses on developing some aspect of stamina, strength, speed and technique. For elite athletes, training is monitored in varying degrees, by physiologists, coaches, biomechanists and the athlete themself. Training activities include many hours of self-monitored training sessions, coach monitored training sessions and comparatively few laboratory sessions. Self-monitoring is traditionally done using an appropriate diary system such as that described by Moytoye et.al. (1996) [1]. A compendium by Ainsworth et al. (1993) [2], matching physical activities to EE, allowed reliable estimates of EE to be made from the activities recorded in the diary. The diary system could be unreliable if the interval between recordings is too long, or an irritation to the athlete if the recording interval is too short. In some circumstances, diary monitoring has been shown to be unreliable ([3][4]) and electronic monitoring is the preferred option.

The introduction of a Micro-Electro Mechanical Systems (MEMS) based athlete monitoring system is intended to simplify athlete monitoring by providing an innocuous device that records athlete activity and provides a meaningful output. Existing monitoring systems are used to estimate EE ([3][4][5][6]), count steps or estimate distance ([7][8]) and monitor acceleration patterns ([8]). Some of these can be used by elite athletes but are generally intended for recreational athletes, for personal navigation ([9][10][11][12]) or for the elderly or infirm ([13]).
The primary output of any athlete monitoring system, including the diary system, is the measure of intensity and duration of a training activity and subsequently the estimated EE. Training intensity may be inferred by lap times (swimming, running), treadmill speeds or heart rate etc. This information is important to the design and implementation of training schemes and the usual audience for this information is the coaches, physiologists and the athlete.

It should be noted that, for fit athletes, there is a high correlation between ambulatory speed, either walking or sub-maximal running, and EE. Laboratory based techniques for estimating EE, including results reported later in this chapter, give high correlations between athlete speed, athlete mass and estimated EE. Because of this, in some circumstances speed can be utilised as a proxy estimator for EE.

5.2.1 Current Energy Expenditure Estimating Techniques

A variety of techniques are used to estimate the EE of an athlete. A common technique is the use of commercial accelerometer-logger units ([4][5][6]) that estimate EE from the magnitude of the acceleration. Other methods include tracking the on-field player using wireless technology\(^{32}\) ([14][15]) or camera technology\(^{33}\) ([16][17]) and estimating EE from the resultant velocity and distance data. In some circumstances, such as open-air sports, Global Positioning System (GPS) based techniques can be used. Another common estimating technique is a combined manual/machine technique with an observer manually tracking the actions of the athlete and recording them into a computer. This is time consuming and error prone.

An existing system for estimating EE of football players, used internally by the AIS, requires an observer to manually track the activity of a player by monitoring video data, and then enter an activity code into a computer representing one of the activity categories of Table 5.1. The system outputs this information as a movement-category

---

\(^{32}\) Orad [14] use a combination of athlete mounted passive microwave diodes and microwave radar to detect locations of players on a field. Trakus [15] use athlete mounted active transponders and directional receivers to detect locations on the field.

\(^{33}\) Amisco [16] uses instrumented fields where cameras are mounted to cover the entire playing area. Computers using vision recognition systems track the players. The Amisco system requires post processing and data is not immediately available. SportX [17] is a similar concept but requires the athletes to wear a camera recognisable symbol.
per-second. This output is subsequently used as input to another system that tracks physiological function. Monitoring a single player in both training sessions and in competition is time consuming. Using an accelerometer-based logging system to generate a per-second EE estimate will eliminate the restrictive manual data entry step.

<table>
<thead>
<tr>
<th>Activity Category</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Stationary</td>
</tr>
<tr>
<td>2</td>
<td>Walking</td>
</tr>
<tr>
<td>3</td>
<td>Jogging</td>
</tr>
<tr>
<td>4</td>
<td>Striding</td>
</tr>
<tr>
<td>5</td>
<td>Sprinting</td>
</tr>
</tbody>
</table>

The activity categorisation of Table 5.1 includes three categories that could be defined as running. Existing categorisation of accelerometer output is limited to lower intensity activities ([6][18][19]) and activities beyond this are simply labelled 'intense' or 'vigorous'. Treuth et.al. (2004) [18] labelled brisk walking as moderate and anything beyond this as vigorous. The categorization of the output from existing accelerometer systems is limited and not able to discriminate between different running speeds and therefore cannot discriminate between the different intensities of the activity. Brage et.al. (2003) [20] attempted to adjust the output of the CSA accelerometer system by applying a step-frequency based filter correction. Due to the wide variation in output, the resultant data still could not be used as an EE estimator for intense activity.

5.3 **History: Energy Estimation Using Accelerometers**

In 1960 Coates & Meade [21] published findings of experiments that correlated energy used by an individual walking with the vertical distance of that individual's trunk movement. Their technique involved a wire attached to the subject's waist marking out the vertical displacement. The experimental results gave a high correlation (r = 0.98) between volume of oxygen consumed (\(\overline{V}O_2\)) and the vertical displacement, step frequency & mass product. More recent work, such as that of Duff-Raffaele et.al. (1996) [22], attempted to apportion energy use to different components in the
biomechanical subsystem however the correlation identified by Coates & Meade is still relevant.

In 1978 Rewick et al. [23], using a head mounted accelerometer, found that during walking, the integral of the vertical acceleration correlated with the lift per step and subsequently with $\overline{VO_2}$.

Montoye et.al. (1983) [24] developed a small portable uniaxial accelerometer and undertook a comparison between the accelerometer, oxygen use and mercury switches. Subjects randomly worked through a series of physical activities chosen for their similarity to daily activities. The accelerometer results proved to be highly reproducible and, across the combined activities, to have reasonable to good individual and group correlations with $\overline{VO_2}$ (n=21, individual $r = 0.63-0.89$, mean $r = 0.79$, group $r = 0.74$).

When commercial devices became available, the vertically mounted uniaxial accelerometer appeared to become almost ubiquitous in studies of EE in free-living subjects. The more recent availability of triaxial accelerometers has resulted in a range of studies comparing tri and uni-axial accelerometers and various electronic pedometers and evaluating their effectiveness as EE estimators ([25][26][27]). For these devices the EE estimator is referred to as 'counts' and is calculated using techniques closely related to those discussed in Chapter 4, section 4.35.

### 5.4 Limitations of Accelerometers as Energy Estimators.

As noted previously, accelerometer-based EE estimates for running are categorised simply as *vigorous* and attempts to adjust the accelerometer output to differentiate estimation across a range of running speeds have not been successful (Brage, [20]). The original studies investigated walking movement and Coates & Meade referred to the product of vertical displacement, step frequency and mass as the *lift power*. The walking movement has been modelled as an inverted pendulum [28] but when considering only the *vertical* movement it becomes a relatively simple oscillatory action as in Fig. 5.1.
The magnitude of the oscillation is governed by the step length and leg length. Example data from a treadmill study was used to generate a simplified walking model\(^{34}\) with data tabled below in Table 5.2. The model estimates peak positive vertical displacement as occurring with the leg vertical, and minimum vertical displacement at mid stride (both feet on the ground, forming an isosceles triangle). Step Rate by Vertical Displacement is substituted for Lift Power. This simple model demonstrates a key characteristic of walking i.e., the measured acceleration is a function of step rate and step length.

Table 5.2 Correlation coefficients for kinematic parameters during walking, estimated from simple triangle model.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Speed (kmh(^{-1}))</th>
<th>Correlation Coefficient Parameter vs. Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.00</td>
<td>5.00</td>
</tr>
<tr>
<td>Step Rate (Hz) (measured)</td>
<td>1.47</td>
<td>1.97</td>
</tr>
<tr>
<td>Step Length (m) (calculated)</td>
<td>0.57</td>
<td>0.71</td>
</tr>
<tr>
<td>Min Height (m) (estimate)</td>
<td>0.90</td>
<td>0.87</td>
</tr>
<tr>
<td>Vertical Displacement (m) (estimate)</td>
<td>0.04</td>
<td>0.07</td>
</tr>
<tr>
<td>Step Rate * Vertical Displacement</td>
<td>0.06</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Leg Length = 93.9 cm.

Note: the vertical oscillation uses an estimate that is the height of the isosceles triangle formed by the step length and leg length, foot flexure and other factors are ignored.

\(^{34}\) This is a simple model treating the legs of a walker as the sides of a triangle and the step length as the base of the triangle. As a result, as speed is increased by increasing the step length, the variation in height of the triangle increases. In reality, human walking gait has a number determinates some of which tend to limit the change in height (see Ayyappa [29]), however Gard & Childress (2000) [30] (using a 'rocker' model) contend that vertical movement during walking is only a function of leg length and step length.
The running action is fundamentally different to the walking action with the vertical displacement being determined by a variety of unknown factors. Experiments were conducted to determine how the accelerometers responded to athletes running at various sub-maximal speeds and if anthropometric factors or individual technique could account for variation in results.

Other accelerometer limitations are identified in Chapter 4. In particular the limitation of the accelerometer to respond to high intensity impact signals and the necessity to filter to remove these impact signals placed a limitation on the usefulness of some forms of accelerometer based estimates.

5.5 10x10 Treadmill Trial

This trial was one of several trials conducted to collect accelerometer data from different athletes running at different control speeds. It was originally hypothesised that the non-correlation of accelerometer counts and speed was due to some form of individual running efficiency factor. The data collected from this trial was analysed in a number of ways in an attempt to identify any relevant factors. As a side effect of this analysis, various attributes of the acceleration waveforms were identified. As the waveforms are the result of biomechanical activity, the analysis identified similarities and differences between the various athletes. This analysis forms the basis for the biomechanical categorisation discussed in Chapter 6.

While this trial is labelled 10x10 (ten athletes at 10 speeds), one athlete failed to appear for this particular test. To maintain alignment with other conducted tests using the same subjects, the data is retained in the 10x10 format. Data from this test appears in this chapter's Appendix.

5.5.1 Method

Nine male athletes walked and ran on a motorised treadmill at 10 speeds, from 3 kmh\(^{-1}\) to 21 kmh\(^{-1}\). Athletes spent approximately 20 seconds at each speed with a pause between speeds. Speeds incremented in 2 kmh\(^{-1}\) intervals with the change from walking to running self-selected by the athlete. The entire session was recorded using a triaxial accelerometer - data logger unit [31] and a commercial video camera. The accelerometer was constructed from two dual-axis Analog Devices ADXL202E
accelerometer units [32]. The accelerometer and data logger was inserted into the centre of the back of the trouser waistband. The accelerometer was connected to the analogue inputs of a Hitachi H8 microcontroller. Data was sampled at 150 Hz using the H8's 10 bit analogue to digital converters (ADC) and logged to an on-board 32 MByte SD Flash Ram card. The ADXL202 accelerometer was rated at +/- 2 gravities (g). These devices were tested and found to have a linear response to over 6g. The accelerometers were calibrated for offset and sensitivity. The hardware included a hardware single pole 3Hz filter with a response as shown in Chapter 4, Fig. 4.16.

Data from the accelerometers was processed using a low pass 25 Hz Hamming Windowed Finite Impulse Response (FIR) filter (filter length 1 second) to remove noise, and a high pass 0.9Hz Hamming Windowed FIR filter (length 4 seconds) to remove orientation (gravity) signal. At the lowest walking speed, the signal removed by the 0.9Hz filter accounted for less than 2% of the signal power, dropping to less than 0.002% for running speeds.

Uniaxial and triaxial accelerometer counts (AC) were generated at each speed using 10 seconds of data at a point where the athlete had settled into a steady rhythm. For uniaxial ACs, the acceleration data for the vertical axis was squared and averaged over one second. For triaxial results, the data from all three axes was squared and summed and then averaged over one second. Step frequency (SF) was calculated by manually counting the number of steps occurring in 10 seconds in the acceleration data at each speed. The use of 10 seconds of data allowed more accurate estimates of SF and at the same time averaged out second-to-second fluctuations in ACs.

Data was analysed using regression analysis with particular interest in relationships between SF, AC, leg-length, athlete mass and speed. The following subsections deal with specific regression results where it appears that a potentially useful relationship between various inputs and outputs may exist.

5.5.2 Results: Accelerometer Counts vs. Speed

Fig. 5.2(a) shows the variation of Uniaxial ACs and Triaxial ACs as a function of speed for the 9 athletes. The error bars show one standard deviation. For comparison, uniaxial AC results from Brage [20] are given in Fig. 5.2(b).
**Uniaxial ACs** show a linear correlation across all speeds of $R^2=0.61$. The walking component showed reasonable correlation between AC and speed ($R^2=0.62$) but in the running action, AC had little correlation with speed ($R^2=0.19$) and the AC output reduced at higher speeds. There was a considerable discontinuity between running and walking. There was a large variation between individuals. These results appear similar in form to those recorded elsewhere ([20][25]).

![Graph of Uniaxial & Triaxial Accelerometer Counts from Treadmill Trial and results from Brage et al (Fig. 2 in [20]).](image)

The results in (a) have been separated into walking and running. The scale for Accelerometer Counts is system dependent. Graphs give the mean AC + and/or -1 standard deviation.

Results for **Triaxial AC** versus speed had improved correlation at walking speeds ($R^2=0.83$) with a similarly large discontinuity in the ACs. Again, there was less correlation between ACs and running speed ($R^2=0.33$). The mean running ACs showed an asymptotic increase (Fig. 5.2(a)). This curve is not useful for estimating energy from speed. Similar to the uniaxial results, the triaxial results showed a large variation between individuals both in magnitude and slope. Maximum and minimum individual triaxial AC results are graphed in Fig. 5.3. While the trendlines indicate that the results are close to a straight line, because the line is flat in many cases, the correlation coefficient is approximately zero.

Uniaxial ACs in their current form do not appear to provide useful running information, either at an individual or group level. Similarly, Triaxial ACs, while appearing to provide a 'better' result, do not provide a more useful result.
5.5.3 Results: Step Frequency vs. Speed

Step frequency results (Fig. 5.4(a)) gave two distinct SF/speed relationships. In contrast to the AC results, SF had a strong group correlation to speed for both walking ($R^2 = 0.92$) and running ($R^2 = 0.74$). Individual correlations were also strong with individual results for all speeds giving $R^2=0.82-0.91$ (mean 0.87), walking speeds $R^2=0.96-1.0$ (mean 0.99) and running speeds $R^2=0.93-0.98$ (mean 0.95). These results are similar to the results of Brage, which are reproduced in Fig. 5.4(b) for comparison.

From analysis, the magnitude of the triaxial AC in Fig. 5.3 appeared to be influenced by SF. As the predominate component of triaxial AC was the vertical acceleration, the uniaxial (vertical) AC results were analysed against the SF. Comparing the individual's 9km$h^{-1}$ SF with their average uniaxial AC (for 9-21km$h^{-1}$) (Fig. 5.5(d)) gave a strong
positive relationship \((r = 0.76)\). This suggested that the magnitude of the individual's AC results was strongly influenced by the \(9\text{kmh}^{-1}\) SF. \(9\text{kmh}^{-1}\) is used as a reference speed as this is the lowest speed where all athletes are in a full running mode.

### 5.5.4 Results: Anthropometric Effects

Viewing individual triaxial ACs for running, such as the example of Fig. 5.3, it was noted that there were distinct variations in the magnitude and slope of individual results. Regression analysis on the slope of the triaxial AC resulted in the identification of a negative correlation between the slope of an individual's AC graph and their leg length \((r = -0.76)\)\(\text{ speeds 11 to 21kmh}^{-1}\). The correlation for full leg-length was stronger than the correlation for the leg component parts. This result is depicted graphically in Fig. 5.3 where runners with longer legs (athletes 1 & 7) have flatter graphs than those with shorter legs (athletes 4 & 9).

Longer leg-length had a consistent negative correlation to SF at all speeds, ranging from \(r = -0.59 \text{ @ 17kmh}^{-1}\) to \(r = -0.75 \text{ @ 13kmh}^{-1}\). Mass appeared to have no direct correlation with AC, with the slope of the AC graph or with SF. Mass appeared to have a correlation with the slope of the SF graph \((r = 0.57)\), i.e. greater mass may affect the rate at which the SF increases with speed.

### 5.5.5 Discussion: Source of Accelerometer Count Variation

It is clear from the wide variation in individual AC results from this system that estimating running speed from raw AC is not feasible. Other research results discussed previously also suggested that raw AC were not a viable estimator of speed. The biggest obstacle to utilising AC is the large variation between individuals. Since the anthropometric factor of leg-length impacts both SF and AC this factor may be useful in normalising the results.

An inspection of acceleration signals for running athletes indicated that the vertical acceleration, the predominate signal, is fundamentally sinusoidal. By assuming a sinusoidal trunk movement, the time varying in-stride displacement \(s\) at time \(t\) can be estimated by Eqn.5.1 where \(A\) is the amplitude of the displacement and \(f\) is the SF in Hertz (Hz). Eqn.5.2 is the first derivative of Eqn.5.1 and represents the in-stride velocity \(v\). Eqn.5.3 is the second derivative of Eqn.5.1 and represents the time varying
in-stride acceleration $a$. The magnitude of the in-stride acceleration is a function of both $A$ and $f^2$ therefore higher SF tends to increase the AC.

Normalizing each athlete's 9km$^{-1}$ acceleration signal by the average 9km$^{-1}$ SF allows a comparison of the relative displacements. This involves multiplying the original 9km$^{-1}$ acceleration signal by a factor of $(SF_{\text{average}}/SF_{\text{individual}})^2$. These appear plotted against the 9km$^{-1}$ AC in figure Fig. 5.5(a), and against leg-length in Fig. 5.5(c). Plotting the relative 9 km$^{-1}$ displacements against the AC resulted in a correlation coefficient of $r = +0.82$. This appeared reasonable since the two contributing factors to acceleration (and hence to AC) were SF and displacement. The overall magnitude of an individual's AC appeared to be governed by the 9km$^{-1}$ SF while the AC at 9km$^{-1}$ was strongly influenced by the athlete's trunk displacement.

Equation 5.1. $s_i = A \sin(2\pi f t)$
Equation 5.2. $v_i = A(2\pi f) \cos(2\pi f t)$
Equation 5.3. $a_i = -A(2\pi f)^2 \sin(2\pi f t)$

From the foregoing analysis, two main factors were shown to determine the basic AC vs. speed characteristic for running. The magnitude of the individual's AC curve appeared to be determined both by the chosen starting step-rate (Fig. 5.5(d)), and the initial relative vertical displacement Fig. 5.5(a). The relative vertical displacement appeared to be influenced by leg-length Fig. 5.5(a), which also negatively influenced SF in Fig. 5.5(b). From previously identified results, the slope of the curve was determined, to a certain extent, by the athlete's leg-length.

---

Fig. 5.5  Miscellaneous regression analysis (a) relative vertical displacements at 9km$^{-1}$ (after normalising for step frequency) vs. accelerometer counts, (b) step frequency at 9km$^{-1}$ vs. leg-length (c) relative vertical displacement at 9km$^{-1}$ (after normalising for step frequency) vs. leg-length and (d) step rate at 9km$^{-1}$ vs. average running accelerometer counts.
In an attempt to visualise the changes to displacement magnitude with increasing speed, the triaxial AC for each individual were normalised by their own 9kmh\(^{-1}\) SF. The results of Fig. 5.6 give an indication of how an athlete's displacement amplitude may change with speed. For this group, the peak SF normalised triaxial AC occurred in the range 9-15 km\(h^{-1}\) with some ACs at 21 km\(h^{-1}\) dropping to as little as 50\% of the peak value. If this is a valid indicator of changes in displacement magnitude then the individual's displacement vs. speed relationship is an added complexity in the speed independent appearance of the AC results.

![Fig. 5.6 Triaxial accelerometer counts normalised by the 9kmh\(^{-1}\) step-frequency. Removing changes in counts due to increasing step frequency gives an indication of changes in displacement amplitude.](image)

### 5.5.6 Discussion: Triaxial vs. Uniaxial Results

The triaxial AC appears to have improved correlation as well as higher overall values when compared to uniaxial results. Comparing Fig. 5.3 (triaxial ACs for athletes 1,4,7,9) and Fig. 5.7 (vertical acceleration for athletes 1,4,7,9), it appeared that the magnitude of the vertical acceleration corresponded well with the triaxial AC, athletes 1 & 7 had consistent vertical acceleration magnitude across the speeds and consistent triaxial AC. Athletes 4 & 9 had increasing vertical acceleration magnitudes and increasing triaxial AC.
Since the triaxial AC appeared to correspond to the vertical acceleration magnitude, the reducing AC shown in the uniaxial results must be attributable to some leakage of ACs to other axes as speed increased. Athletes 4 & 9 must also have had proportionally more signal in the other axes since in Fig. 5.3 athlete 4 was consistently lower than athlete 1, and athlete 9 was generally lower than athlete 7, yet in Fig. 5.7, athletes 4 and 9 exceeded athletes 1 and 7 respectively. Further investigation of these phenomena is reported in Chapter 6.

Of interest, from the aspect of identifying the source of variation in AC, is the fact that the slope of the individual triaxial AC vs. speed regression line appears to be directly linked to the vertical force exerted by the athlete, which is itself related to leg length.

5.5.7 Discussion: System Filtering

Any interpretation of signal magnitudes must be considered in relation to the impact of the system filtering. In this system, the post processing FIR filtering was at frequencies that do not significantly impact the dynamic signal magnitudes. The hardware 3Hz filter, made necessary by the overload impact signal, will impact signal magnitudes. Although the SF for running covers a small band from around 2.5Hz to 3.5Hz, the acceleration signal will be attenuated more at the higher frequency. This impacts all
AC results. A possible rule of thumb for reinterpretation is that high speed equates to higher SF and therefore increased attenuation, therefore acceleration and AC results at higher speed should be greater than they appear.

A mathematical analysis similar to Brage's [20] could be performed however this is considered unnecessary for the following reasons:

- Higher AC results already positively correlate with initial SF.
- The slopes of the individual AC results appear to be linked to SF via leg-length, such that greater slopes will tend to increase more than the lesser slopes.

While an increasing AC at higher speed will improve the correlation between AC and speed for the whole group, both of the effects described above will work to increase the difference between individuals. This analysis may be incorrect if the higher frequency harmonics contribute differently for different anthropometric factors. This is not considered a factor as investigation of the biomechanical activity (in Chapter 6) strongly suggests that considerable channel specific harmonic power is related to small orientation changes affecting the horizontal accelerometers (see Fig. 6.17 & Fig. 6.18).

### 5.5.8 Summary: 10x10 Treadmill Trial

The analysis of this data identifies that ACs had poor or no correlation to speed when running and therefore did not appear to be suitable for use as an EE estimator for athletes involved in running activities. ACs for running had wide variations between individual results. AC based output was affected by anthropometric and related factors, as well as technical factors such as filtering (and from Ch. 4, also affected by overloads, alignment error, calibration error etc).

The SF gave very strong individual correlation to speed as well as strong group correlation to speed. The SF appeared to be directly influenced by the subject's leg-length.

As the running SF had a strong correlation with speed (as shown in this and other studies), then SF, in conjunction with anthropometric measures of leg-length and mass,
may be a more appropriate estimator of speed and hence EE. The following experiment is designed to test this hypothesis.

5.6 **Treadmill Trial for Energy vs. Step Frequency**

The previously discussed treadmill trial and other previously referenced research identified a linear relationship of speed to SF for the tested running speeds. It is appropriate therefore, to identify if SF has any correlation with estimated EE.

5.6.1 **Method:**

Ten recreational runners performed walking and running at speeds between 4km h\(^{-1}\) and 20km h\(^{-1}\) on a custom-built motorised treadmill. The accelerometer devices described previously (Section 5.5) were firmly attached to the athletes, located in the centre of lower back at waistband height. Accelerometer data for this trial was processed for AC and SF as for the previous trial. The runner's starting speed, walk-to-run changeover speed and final speed were all self selected. After a four minute normalisation, \(O_2\) consumption was measured for one minute at each speed and converted to kilojoules (kJ) using appropriate analysis techniques and conversion algorithms.

The gas analysis method is described in Saunders et al. (2004) [33].

"Respiratory gases were analysed on a custom designed and built open-circuit indirect calorimetric system with associated in-house software (Australian Institute of Sport, Belconnen, Australia). This automated system uses the Douglas bag principle [34] to collect all expire into one of two 150 L aluminised bags. While one bag is being filled, the other has the expired volume and gas fractions determined. Standard algorithms are employed to compute minute values of E, expired CO\(_2\), O\(_2\) and RER from the sum of two consecutive 30 s samples. The O\(_2\) and CO\(_2\) gas analysers (AEI Technologies, Pittsburgh, PA) were calibrated before each test with three precision gas mixtures, with an acceptable calibration being within \(\pm 0.03\%\) of all target values. Volume was measured with a precision-calibrated linear displacement piston coupled to real-time measurement of temperature and pressure inside the piston. The typical error of measurement (TEM)[35] or standard deviation of the differences divided by square root of two for \(O_2\) in the [AIS] laboratory is 2.4% for the pooled data for
running at 14, 16 and 18 km.h⁻¹. The TEM was established from duplicate trials conducted on 11 subjects prior to the start of the [this] study.

5.6.2 Results

One runner achieved 20km h⁻¹ and seven runners achieved 16km h⁻¹. Results for AC and SF were similar to the previous trial and are given in Table 5.3. AC correlation to speed was improved but this appeared to be due to the athletes not attaining the higher speeds. The individual results were similar with a wide spread of ACs, uniaxial ACs peaked and decreased, and triaxial ACs decreased, flattened or increased depending on the individual. EE estimated using VO₂ results gave strong correlations with speed. Adjusting the EE estimate and SF results using predictor-based adjustments for mass and leg-length improved the results further.

Table 5.3 Correlation Coefficients for EE, SF & ACs vs. Treadmill Speed

<table>
<thead>
<tr>
<th>Variable</th>
<th>Activity</th>
<th>Individual R²</th>
<th>Mean R²</th>
<th>Group R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE (from VO₂)</td>
<td>Walk &amp; Run</td>
<td>0.95-1.00</td>
<td>0.98</td>
<td>0.87 (mass adjusted 0.96)</td>
</tr>
<tr>
<td></td>
<td>Run only</td>
<td>0.92-1.00</td>
<td>0.97</td>
<td>0.73 (mass adjusted 0.96)</td>
</tr>
<tr>
<td>SF</td>
<td>Walk &amp; Run</td>
<td>0.65-0.96</td>
<td>0.84</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.76 (Adjusted for Leg-Length)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.77 (Adjusted for LL &amp; Mass)</td>
</tr>
<tr>
<td>SF</td>
<td>Run only</td>
<td>0.90-1.00</td>
<td>0.95</td>
<td>0.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.66 (Adjusted for Leg-Length)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.73 (Adjusted for LL &amp; Mass)</td>
</tr>
<tr>
<td>Triaxial AC</td>
<td>Walk &amp; Run</td>
<td>0.77-0.90</td>
<td>0.86</td>
<td>0.74</td>
</tr>
<tr>
<td>Triaxial AC</td>
<td>Run only</td>
<td>0.74-1.00</td>
<td>0.88</td>
<td>0.45</td>
</tr>
</tbody>
</table>

While speed can be used as a substitute estimator for EE, this experiment recorded both the athlete's kinematic activity and oxygen consumption. This allowed a direct comparison between kinematic data and the EE estimated from VO₂. The regression analysis for EE and other variables gave varying amounts of correlation with results listed in Table 5.4. Individual results gave strong correlations between estimated EE and most variables including SF and AC. Group correlations were weaker with running-only correlations of EE to SF dropping to R²=0.29 and for EE to AC the correlation coefficient dropped to R²=0.17. SF is plotted against EE in Fig. 5.8.
Table 5.4 Correlation coefficients for SF and AC vs. EE estimated from \( \dot{V}_O_2 \).

<table>
<thead>
<tr>
<th>Variables</th>
<th>Correlation Coefficients as ( R^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Individual</td>
</tr>
<tr>
<td>SF (R+W)</td>
<td>0.57*-0.96</td>
</tr>
<tr>
<td>SF (R only)</td>
<td>0.80-0.99</td>
</tr>
<tr>
<td>Triaxial AC (R+W)</td>
<td>0.74-0.96</td>
</tr>
<tr>
<td>Triaxial AC (R only)</td>
<td>0.73-1.00</td>
</tr>
<tr>
<td>Uniaxial AC (R+W)</td>
<td></td>
</tr>
</tbody>
</table>

*0.57 is an outlying value, next lowest 0.80, average value after removing outlying 0.87

Group correlation results on (a) SF alone, (b) SF adjusted for leg length, (c) SF adjusted for leg-length and mass, (d) SF adjusted for mass.

As for the previous experiment, SF recorded a negative correlation to leg-length at all speeds with correlation coefficients for running ranging from \( r = -0.53 @ 10 \text{ kmh}^{-1} \) to \( r = -0.83 @ 16 \text{ kmh}^{-1} \) (Average \( r = -0.72 \))

![Graph showing Step Frequency vs. Estimated EE](image)

**5.6.3 Discussion:**

The key area of interest in this study was the relationship between SF and EE during running. The initial results gave strong individual correlations between SF and EE (\( R^2=0.80-0.99 \)) but weak group results (\( R^2=0.29 \)) as indicated in Fig. 5.8. From the graph, it can be seen that a SF of 2.75Hz could represent a range of EE between 35kJ and 90kJ. As EE is affected by the athlete's mass, and SF has been shown to be affected by the athlete's leg-length, linear predictions based on these values were attempted. Further, the previous study indicated that mass might have some impact on the change in SF (\( \Delta SF \)) with speed. A number of linear and non-linear SF prediction
models incorporating mass were tested. The simple linear model of Eqn.5.4 appeared to give adequate results, improving group correlation results from $R^2=0.29$ to $R^2=0.81$.

**Equation 5.4:** \( EE = \beta_1 \times SF \text{ (Hz)} + \beta_2 \times \text{Leg-Length (cm)} + \beta_3 \times \text{MASS (kg)} + C \text{ (kJ)} \)

(Combined walking/running coefficients: \( \beta_1=56.4, \beta_2=1.82, \beta_3=0.35, C=-282 \text{kJ} \))

The results depicted in Fig. 5.8 were adjusted for anthropometric factors of leg-length and mass and re-graphed in Fig. 5.9. To allow a comparison between the original and modified results, the graphical output of Fig. 5.9 was generated using a slightly less optimal arrangement of converting both EE (from $\dot{VO}_2$) and SF to speed, normalising the results, and plotting these against each other. In this case EE was adjusted by mass and SF was adjusted by Leg-length.

For this group, at these sub-maximal running speeds, the SF to EE correlation for the combination of raw $\dot{VO}_2$ EE figures and Leg-Length adjusted SF ($R^2=0.79$) is almost as strong as the correlation between EE and Leg-Length-Mass adjusted SF ($R^2=0.81$). This suggests that the effect of mass on EE and on SF follows a similar pattern and that an adequate estimator can be generated from knowledge of leg-length and SF alone.

For monitoring elite athletes, a direct EE estimator can be obtained by calibration of the individual athlete. Treadmill training on a motorised treadmill requires no sensors as the treadmill speed and athlete mass can be used as a direct estimator for EE. In over-ground running, using sensors to record SF and adjusting for leg-length and mass can give an approximate solution but based on the individual results, calibrating the athlete for SF at two known speeds would provide a more accurate estimate. SF at two speeds allows the generation of the SF/Speed regression line and these were shown to have strong individual correlations. EE also has strong individual correlation to speed, and, when mass adjusted, shows strong group correlation.
5.6.4 Summary:

For individuals, SF during sub-maximal walking and running exhibited a strong relationship to EE estimated from $\dot{V}_O_2$. Group SF and $\dot{V}_O_2$ estimated EE results exhibited a much weaker relationship. Combining measured SF with an individual's leg-length and mass in a linear predictor equation resulted in a strong correlation. This result, while good, and a vast improvement on AC based estimates, can be improved on by calibrating individual athletes. This calibration requires measuring their SF at two running speeds. This gives the individual SF to speed relationship and hence an improved individual SF based estimated EE could be obtained. For athletes involved in running as part of their training regime, monitoring SF would provide a highly reliable, non-invasive EE estimate.

5.7 MEMS based Per-Second Energy Expenditure Estimator

To create a MEMS based per-second EE estimator to replace the manual system of section 5.2.1 that can generate the classifications of Table 5.1, requires that the system can discriminate between the different classifications of activities.

The results from the treadmill trials of sections 5.5 and 5.6 and previously discussed research indicated that, while ACs gave good correlation with low to moderate intensity activities, they were not suitable as an EE estimator for intense activities or for discriminating between different levels of intensity for running. The treadmill trial
results also indicated that SF, when modified by specific anthropometric factors, was a good estimator for speed and EE at both walking and sub-maximal running speeds.

The analysis of Chapter 4, Section 4.3.6.3, identified that at lower walking speeds, identifying steps in acceleration signals such as those of Fig. 5.10 was problematic. The use of zero-crossing detection as a step-detector was unreliable. The use of FFTs to detect steps was also unreliable. Autocorrelation could be used to identify the SF in some cases but this is a processing and memory intensive function. The study of pedometers by Bassett, Ainsworth & Leggert (1996) [7], found that at low walking speeds none of the brands of pedometers tested could reliably detect the stepping action. Because of the difficulty in reliably detecting steps at lower walking speeds, SF cannot be used as an estimator of EE at these speeds. This limits the use of SF as an energy estimator to the moderate to higher intensity ambulatory activities.

Given the restrictions of step detection at low walking speed and the non-reliability of ACs at higher speeds, a combination approach using ACs for low intensity EE estimating and SF for high intensity ambulatory EE estimating appears appropriate.

![Comparison of vertical acceleration for walking.](image)

**Fig. 5.10** Comparison of vertical acceleration for walking. Of the three, Athlete A5 has the clearest detectable step signal with a large pulse on each step. For Athlete A10, the asymmetry, with the acceleration from one step greater than the alternative, makes automatic detection more difficult. There is no discernable pattern to Athlete A8's vertical acceleration.

### 5.7.1 Combined AC - SF Categorisation

Table 5.5 below identifies a decision matrix for combining the AC and SF outputs to generate the required activity banding. For this to be generated on-board a triaxial accelerometer based monitoring device, the individual athlete's activity threshold characteristics must be pre-loaded. An alternative mechanism, requiring no individual knowledge of the athlete, treats the AC and SF outputs as a continuum. A per-second
output represents a point on the continuum and this value can be used to estimate EE by post processing. This removes the necessity to load values into the monitoring device and allows greater flexibility. This flexibility exists because the post-processing step can be used to apply individual SF/Speed regression line calibration, or anthropometric adjustments based on group characteristics for the age/gender/fitness grouping of the athlete.

Table 5.5 Combined AC - SF Decision Table

<table>
<thead>
<tr>
<th>Activity</th>
<th>3D Accelerometer Count (AC)</th>
<th>Step Frequency (SF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stationary</td>
<td>AC &lt; Stationary Threshold</td>
<td>Not Required</td>
</tr>
<tr>
<td>Walking</td>
<td>Stationary &lt; AC &lt; Walking Max.</td>
<td>Not Required</td>
</tr>
<tr>
<td>Slow Jog-shuffle</td>
<td>Walking Max &lt; AC &lt; Run Min.</td>
<td>SF</td>
</tr>
<tr>
<td>Running</td>
<td>Running Min &lt; AC</td>
<td>SF &lt; Sprint Threshold</td>
</tr>
<tr>
<td>Sprinting</td>
<td>Running Min. &lt; AC</td>
<td>SF &gt; Sprint Threshold</td>
</tr>
</tbody>
</table>

Note: Using this decision table, it is only necessary to set the appropriate thresholds to get the banded output. Alternatively, the bolded items can represent a continuum.

To output this continuum, it is necessary to extract both the AC and SF values from the acceleration signal. The data used in the preceding treadmill trials was sampled using 10bit sampling at 150Hz. In line with the system requirements identified in earlier chapters, it was necessary to minimise the processing requirements of the sensor device. SF & AC extraction at low sampling frequencies is discussed below. The method of combining results to generate the continuum is depicted in Fig. 5.11.

![Fig. 5.11 Generating combined AC-SF categorisation. Rescaled mean walking triaxial AC from Fig. 5.2 (a) is rescaled to match the mean walking SF from Table 5.4. The crossover from AC to SF occurs at 2Hz. The combined output is generated in Hz.](image-url)
5.7.1.2 Low Sampling Rate SF Detection.

To extract an oscillatory signal using a sampling system, it is necessary to sample at a frequency at least twice the maximum frequency of the system being monitored. This sampling frequency is known as the Nyquist Rate. Analysis of 10 metre split timing from 100 metre sprint data provided by the AIS, identified maximum sprint speeds in excess of 40 km h\(^{-1}\) for an international level sprinter. Extrapolating SF data from the 10x10 Treadmill Trial gave a maximum 41 km h\(^{-1}\) SF of approximately 5 Hz. This is only an approximation as the research of Hunter, Marshall & McNair (2004)[36] suggests that as an athlete approaches maximal speed, the rate of increase in step frequency increases while step length plateaus or decreases. The fastest sprinter in the Hunter et.al. study, reached 31.7 km h\(^{-1}\) at a SF of 4.45 Hz. Elliott & Blanksby (1976)[37] reported that at higher running speeds, SF is higher in treadmill running than overground running. Taking these factors into account, a maximum SF of 5-5.5 Hz could be expected, requiring a minimum sample rate of 11Hz. A higher sample rate is preferred.

The extraction of SF at low sample rates using linear interpolation was discussed in Chapter 4, Section 4.3.10.1. Generating a per-second estimate requires the averaging of the SF from several successive steps, with the number of steps depending on the actual SF. Using a sample rate of 25 Hz at a SF of 5Hz, on average there will only be 5 samples per step.

5.7.1.3 Low Sampling Rate AC Extraction.

To determine if the correlation between ACs and walking speed exists for low sample rates, the data collected from the 10x10 treadmill trial was interpolated at different sample rates and with different numbers of bits per sample. This data was reprocessed to generate uniaxial and triaxial ACs and then analysed using linear regression. The results (Table 5.6) from the regression analysis indicated that the reduced sample rate and sample bits had minimal effect compared to the correlation coefficients given by the regression analysis of the original 10-bit, 150Hz samples, where \(R^2=0.83\) for triaxial ACs and \(R^2=0.62\) for uniaxial ACs. Based on this result, ACs extracted from low-sample-rate data appeared to maintain the required relationship.
### Table 5.6 Correlation Coefficient as $R^2$ for reduced Samplerate ACs vs. Walking Speed

<table>
<thead>
<tr>
<th>Samplerate</th>
<th>10 Bit Samples</th>
<th>8 Bit Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Triaxial</td>
<td>Uniaxial</td>
</tr>
<tr>
<td>15Hz</td>
<td>0.80</td>
<td>0.58</td>
</tr>
<tr>
<td>20Hz</td>
<td>0.83</td>
<td>0.62</td>
</tr>
<tr>
<td>25Hz</td>
<td>0.82</td>
<td>0.58</td>
</tr>
<tr>
<td>30Hz</td>
<td>0.80</td>
<td>0.57</td>
</tr>
</tbody>
</table>

### 5.7.2 Results

The banded output from the combined system was generated and visually compared to the outputs from manually categorised data for the same activity (football matches). From this visual comparison many small discrepancies were identified. These small discrepancies generally involved short durations of 2-3 seconds where the activity wasn’t recorded or was incorrectly categorised. Reviewing the accelerometer data identified that the activity appeared to be appropriately categorised. Discussion with the person responsible for the manual data input identified that in general; changes in type of activity of short duration were often ignored and incorrect categorisation was not retrospectively corrected. For example, the operator may enter one code, realise the mistake and enter the correct code 2-3 seconds later. These short mis-categorisations were not considered to be a serious impediment to the overall results. To further ascertain the effectiveness of the automatic generation of per-second categorisation additional comparisons and study may be required. A potential qualitative technique would be the combining of the per-second output with the video of the athlete activity.

One of the overall system outputs is histograms showing the overall distribution of intensity. For comparison, a histogram based on the AC system output was generated for Fig. 5.12 and a combined AC-SF system output used to generate Fig. 5.13. Both these systems use a continuum output rather than a banded output. These figures represent the intensity of activity of one player for one-half of a football match. The figures divide the output into bands representing the percentage time spent at a particular intensity level. The first band represents approximately 50% of the overall time while subsequent bands represent approximately 10% each. The AC based output
(Fig. 5.12) was rescaled to generate an output on a similar scale to the AC-SF based output (Fig. 5.13).

The above figures appear similar however, as ACs have a discontinuity between running and walking, and the weighting of running ACs is higher relative to walking ACs, the output of the AC based system undervalues lower intensity activities. Using intensity-by-time as a substitute for energy, the AC system has the top 10% of activity consuming more energy than the AC-SF system (1067 units vs. 900 units). Overall, the
AC-SF based system suggested higher energy expenditure compared with that indicated by the AC only system (3202 units vs. 2768 units). This is predominately due to the matching of the weighting of the walking AC to the walking SF.

The shape of the graph of Fig. 5.13 is more amenable to interpretation than that of Fig. 5.12. The preferred running SF of the athlete appeared to be centred on 2.6-2.7Hz, with the athlete running for over a total of 400 seconds at between 2.5 and 2.8Hz. This information is not available from an AC based system due to the non-linear AC response.

5.7.3 Summary

A combined AC+SF approach appeared to improve the MEMS based per-second estimate of EE. The assessment of the system was limited in that the manual-categorisation data used in comparisons was itself subject to considerable error. The scaling of ACs to match the equivalent SF rates appeared to capture information, particularly low intensity activity that would otherwise be lost.

5.8 Chapter Summary

The use of accelerometers in estimating EE in daily activities is a common technique that has been in widespread use for some time. Data presented here showed that due to the biomechanics and kinematics of running, the traditional AC output from the accelerometers is not suitable for estimating EE when the athlete is running. Treadmill trials and other published research indicated that SF strongly correlated with speed and that speed, adjusted for the athlete's mass, strongly correlated with EE.

Experimental results indicated that an individual's ambulatory SF strongly correlated with \( \dot{V}O_2 \) based EE estimates. Experimental results indicated that the group correlation for SF to estimated EE was not sufficient to use SF as a general estimator of EE. Investigation of anthropometric factors indicated that an athlete's leg-length, and to a lesser extent the athlete's mass, affected the athlete's SF to speed relationship and ultimately the athlete's SF to estimated EE correlation. Using a linear predictor equation incorporating leg-length, mass and SF, a strong group correlation (\( R^2=0.81 \)) was obtained for SF to \( \dot{V}O_2 \) estimated EE.
While ACs were not a useful tool as an EE estimator for high intensity ambulatory activity, SF was difficult to extract from low intensity ambulatory activity. A combined AC-SF based system was proposed, trialed and the results analysed. A low sample rate MEMS based system was used to generate per-second activity banding for football activities and this system compared to a manual categorisation system. The MEMS based system appeared to generate a more reliable output than the manual system.

The results of experiments and trials reported in this chapter indicated that the use of a combined AC-SF based activity identifier (Table 5.5) was more reliable than a manual system. The use of a combined AC-SF output continuum provided a better basis for EE estimation than either AC or SF alone.

This system is further described in *Signal Processing for Estimating Energy Expenditure of Elite Athletes using Triaxial Accelerometers*, Wixted et.al. (2005)[38] and *Measurement of energy expenditure in elite athletes using MEMS based inertial sensors*, Wixted et al (2007)[39]

Future Research: While this system is useful for documenting the EE during steady state activities, such as steady state walking and running, game activity involves many additional moves and activities. Further research is required to quantify the EE that occurs in game specific activity. Where this activity specific EE proves to be significant to the overall EE, appropriate signal processing should be developed to extract the activity and estimate the EE.
5.9 References


6 Biomechanical Analysis of Athlete Accelerometer Data

6.1 Introduction

The use of accelerometers for monitoring human and even animal biomechanical activity is common and the data obtained is used in a variety of ways such as the monitoring of ambulatory activity of the elderly or assessing injury. This chapter investigates one small fragment of the universe of biomechanical monitoring, namely:

Q.1. Is there useful biomechanical information that can be extracted from Centre-of-Mass (CoM) triaxial accelerometer data collected from elite athletes engaged in running?

The introduction of this chapter briefly describes the term biomechanics, the use of 'biomechanics' in the context of sports science, a selection of biomechanical monitoring activities relating to sports science and elite athletes in particular. It then describes the context and limitations of accelerometers used in biomechanical monitoring.

The remainder of the chapter covers:

- The data sets used for analysis.
- Previously identified and potentially useful analysis techniques.
- Information that appears to be contained in the data.
- The combination of processing steps used in the analysis of the data.
- Categorisation of movement based on this information.

Finally the results of this chapter are summarised.

6.1.1 Biomechanics

The term 'Biomechanics' is used to describe a continuum of investigative studies into the mechanics of biological structures. This continuum operates from the smallest biological level to the greatest, for instance, studying the mechanical operations of living cells, the flow of blood through capillaries, the operation of a muscle undergoing contraction, the interaction of muscles, bones and ligaments to move a limb and finally - the total integrated mechanical system that results in (say) human locomotion. Biomechanics of humans at the macro-level, that of the interaction of
limbs and the trunk etc., may often be described under the heading of Human Movements Studies. The study of the human movement of athlete's within the context of 'Sports Science' is often referred to simply as biomechanics.

6.1.2 Biomechanics & Sports Science

Within sport's science, biomechanics is studied across a wide range of sports using a range of techniques. For instance, within the AIS athlete-tracking project, biomechanics studies have formed a substantial component. The biomechanics of rowing has been modelled [17]. A separate project was created to build an Australian standard rowing biomechanics system [18]. Sporting activities such as swimming [19], snowboarding [20] and a wide variety of other sports have been analysed. Some biomechanical tracking systems use high-speed cameras to capture athlete activity and to convert the pictorial information into computer extracted biomechanical parameters. This technology can be used in the monitoring of many different activities. Other systems, such as the Force-Plate, capture information on the time-series vector forces applied by the foot to the ground during ambulatory activity, particularly running.

Numerous papers have been written investigating biomechanics with relation to running athletes or with relation to the technologies discussed here, a small sampling follows. Elliott & Blanksby (1976)[21] analysed the differences in overground and treadmill running and identified a number of biomechanical relevant differences such as changes in the stance phase. Kivi et.al. (2001)[22] measured lower extremity

---

35 Human Movement Studies is not a new field of study, Aristotle (384 BC - 322 BC) wrote "On the Progression of Animals" [16] and covered many fundamentals of Human Movement & biomechanics.

36 Numerous camera based biomechanics analysis systems exist, a small sampling of biomechanics papers will identify several. This technology is used for the assessment of injury, disability, aging, sporting activity etc.

37 Similar to the camera technology several suppliers of force plate exist and the technology is used for similar purposes to the camera technology. Use of this technology even extends to assessing animal injury. AIS 'Athlete Tracking' projects extend to the development of force-plate replacement technology using in-sole sensor systems [23][24].

38 Stance is the time when a particular foot is on the ground. It is usually measured as a percentage of the gait-cycle ie. a 25% stance phase indicates that the left foot is on the ground for 25% of a stride. Assuming this refers to running, then during one stride the body is supported for 50% of the stride.
kinematics of subjects sprinting on a high speed motorised treadmill. Hunter et.al (2004)[25] studied the interaction of step length and step rate during sprinting. This study monitored a number of lower extremity points as well as CoM factors such as horizontal and vertical velocity at specific points of the gait cycle, CoM angle of takeoff and various other CoM variables. Mann & Hagy (1980)[26], using a combination of elite athletes and experienced recreational runners, tracked a number of biomechanical parameters as well as muscle activity using electromyography\textsuperscript{39}. They identified the reducing length of the stance phase as running speed increased as well as a lowering of the centre-of-gravity due to changes in the muscle and joint actions as speed increased. Karamanidis et.al. (2003)[27] investigated the symmetry\textsuperscript{40} and reproducibility of lower extremity kinematics during running at different step rates at the same and at different speeds. These, and many other papers researching running biomechanics, particularly in relation to elite athletes, are an attempt to answer a short list of questions:

1. What are the key biomechanical parameters of running?
2. With the knowledge of (1) above, can an athlete be trained to run faster?
3. Can an athlete run faster and at the same time reduce the risk of injury?

These questions are almost a paraphrase of the statements of Williams (1985)[28].

6.1.3 Kinematics

The analysis of athlete accelerometer data to uncover the underlying biomechanical activity falls into a subdivision of classical mechanics\textsuperscript{41} called kinematics. In this case kinematics uses the captured acceleration\textsuperscript{42} data to derive velocity and displacement and together these components are used in the describing of the movement of the sensor system within a Frame of Reference (FOR) that is directly related to the athlete.

\textsuperscript{39} Electromyography is a technique for detecting muscle activity through the electrical activity of the muscle cells.

\textsuperscript{40} Moe-Nilssen & Helbostad (2004)[16] used autocorrelation of CoM accelerometer data to detect asymmetry (processing described in Chapter 4).

\textsuperscript{41} Classical mechanics (also known as Newtonian Mechanics, named after Sir Issac Newton, 1642-1727), deals with Statics, Dynamics, Kinetics and Kinematics. Contributors to Classical Mechanics include Galileo, Kepler, Newton, Lagrange, Hamilton and others.

\textsuperscript{42} Because acceleration is captured, kinematics is used to derive velocity and displacement. If a positional transducer were used, the velocity and acceleration would be the derived components.
Further, this study seeks to relate the dynamic orientation of the athlete FOR within a particular FOR, discoverable through the effect of gravity on the sensor system. Much, but not all, of this data is without any external reference except gravity. Using gravity, information about the orientation of the sensors is discoverable and this information then used to rotate the data from one FOR to a more suitable FOR.

By synchronous collection of this data in combination with data from other systems, such as in-sole sensors, a general approach to resolving the kinematic data into biomechanical information of interest can be made. Using systematically collected data, the different forms of kinematic activity can be categorised and translated into the categorisation of an individual's ambulatory biomechanical activity.

The fundamental area of study covered in this chapter is the study of accelerometer data in isolation, and in conjunction with data from other sensors, to describe and categorise athlete biomechanical activity when running.

### 6.1.4 Accelerometer and Frame of Reference (FOR) Limitations

A human body engaged in sporting activity can and does move within a three-dimensional (3-D) Cartesian space. Since the athlete can, and often does, undergo rotation about one or more axes, the CoM accelerometers are attached to a system that has 6 Degrees of Freedom (6-DOF). To accurately describe these movements they must be related to a specific known FOR.

For accurate analysis of biomechanical activity, an external FOR is required. Traditionally biomechanical monitoring using an external FOR is performed using camera technology with the resultant picture frames analysed manually, by the operator manually marking specific points on the picture, or automatically, with a computer tracking location markers placed on the athlete prior to the testing. Combining the output from orthogonally located cameras creates a 3-D time-series record of the activity. This system requires complex equipment and requires the testing to be performed in a specific location.

Rotation of the data is performed using a rotational tensor or matrix (refer Chapter 4 Section 4.3.3).
Using triaxial MEMS accelerometers alone, the only external FOR is gravity. Accelerometers alone cannot orientate (except to know which way is up) or locate an athlete within three-space. A knowledge of the starting point and the terrain (such as knowing the athlete is running clockwise on an oval track) may be combined with accelerometer data, such as step count and step rate to generate indicative locations [29]. Accelerometers alone cannot detect rotation about the vertical axis although rotation about either of the horizontal axes is detectable due to the effect of gravity.

As previously discussed [Chapter 4], it is not possible to extract absolute orientation in the presence of activity, particularly that of fast moving athletes. To overcome this, studies have been performed using a combination of MEMS sensors to locate [30] or orientate a subject [31]. By combining triaxial MEMS accelerometers with triaxial MEMS gyroscopes and triaxial MEMS compasses, a system can be developed that tracks orientation with respect to gravity and the earth's magnetic field. This allows an approximate 3-D image of biomechanical activity to be generated. Where multiple limbs are tracked there are further complexities depending on the data required [32].

For the data gathered in the described experiments there are a limited number of external inputs. For the purpose of biomechanical monitoring of athletes, particularly athletes running, these external inputs provide useful reference data when analysing data from a CoM triaxial accelerometer.

6.1.4.1 Test Data & External Inputs

Three sets of test data are used in the describing and categorising of accelerometer based biomechanical information. This test data has been previously described (Chapters 4 & 5) but is reviewed briefly described here for completeness. This data included various types of external inputs, which are described below.

**Ten x Ten (10x10) Trial:**

In this test ten Australian Rules Football players, monitored by a 10 gram triaxial accelerometer unit (and 30 grams of support electronics and batteries) fixed in place by an elastic belt, walked and ran at 10 speeds on a motorised treadmill. All the subjects were experienced with treadmill running. The speeds went from 3 kmh$^{-1}$ to 21 kmh$^{-1}$ in 2 kmh$^{-1}$ increments. The change over from walking to running was self-selected. No
data was collected from one subject. To preserve the relationships between data collected from different tests, the subjects are referred to as A1 to A10.

**Treadmill trial with in-sole pressure transducers:**
In this trial a single subject ran on a high speed motorised treadmill at five speeds from 10 km\( \text{h}^{-1} \) to 18 km\( \text{h}^{-1} \) (2 km\( \text{h}^{-1} \) increments). The subject was monitored by a 10 gram CoM triaxial accelerometer firmly taped in place over the L3-4 vertebra and by in-sole pressure transducers in the left shoe. Data was collected synchronously from the triaxial accelerometer and the in-sole sensors.

**1500m Track Trial.**
In this trial, two National level competitive athletes separately ran a 1500m trial race. Each athlete was monitored by two triaxial accelerometers (taped in place), one either side of the L3-4 vertebra, and two sets of in-sole pressure sensors (one set in either shoe). Race and lap timing was collected. As data from both triaxial accelerometers was essentially identical for the purposes of biomechanical assessment, data from only one sensor was used here. The accelerometers were mounted on their own very small circuit board and were of very low mass (approx 3 grams).

**External Inputs:**
These are inputs to the accelerometry analysis that assist in interpreting the acceleration signal and its derivatives.

- Motorised Treadmill.
- In-sole sensors.
- Timed track running.
Treadmill Speed:
Tests on a motorised treadmill provided information on the running speed and hence on the stride length since stride rate was captured from the sensors. Changes in detected acceleration were analysed against running speed and various speed dependent changes identified.

In-sole sensors: Synchronously collected data from insole sensors provided direct reference to timing of foot contact with the ground that was then directly related to forces applied on the accelerometer. Foot contacts could not be directly identified from the CoM accelerometers due to the mechanical complexity of the leg and the decoupling due to foot, ankle, knee and hip joints. This, in itself provides information about an athlete regarding factors such as leg stiffness and other biomechanical factors. The in-sole sensors provided information on accelerometer orientation and reference points for analysing braking and forward acceleration phases.

Timed track running:
Runners engaged in track work and wearing multiple sensors were timed as they raced on a track. The effect of running on the straight verses running on the curve was analysed. The running speed, step-rate and acceleration was compared for two runners to identify factors that may play a part in determining the runner's performance.

6.2 Biomechanics Signal Categorisation
Categorisation of the acceleration signal requires knowledge of what the acceleration signal represents and at what points in the continuum of biomechanical analysis, the acceleration signal information and the needs of biomechanists intersect. A common mode of data presentation is required and in this case, most of the acceleration signal based analysis, or its derivatives, is presented graphically as gait-cycle normalised data. In this there is a small variation from the normal mode of presentation in that due to the cyclical nature of the signals under investigation, gait cycle is measured in degrees instead of the more common percentage measure.

The following subsection picks out a few points from various studies or texts where different potentially useful variables are discussed. Some of those variables, where
they relate to trunk kinematics or possibly foot contact, may be extractable from the accelerometer data. There are other variables that the author considers may be of potential interest in the study of athlete biomechanics. These points are generally identified in the form of questions, that later analysis will attempt to verify or discard.

6.2.1 Biomechanics Variables and Variable Quantification.

The categorisation or identification of biomechanical activity from athlete mounted, CoM located, triaxial accelerometers is complex\(^{44}\). Lenhoff et.al. (1998) [33] identified the complexity of quantifying a single continuous parameter (such as knee flexure angle) over a complete gait cycle and developed techniques such as bootstrapping and confidence bands. These techniques were developed to assist in characterising a particular parameter so that a subject could be identified as belonging within the group of acceptable responses. Where multiple variables of interest are involved the complexity increases accordingly, for example in [25], 23 variables across 12 body segments were recorded. The number of variables, the bands representing normal range of the variable and possible interactions necessitates the development of techniques for visualising the data such as [34]. Generally these techniques are related to subjects walking as this is the area of greatest need in the wider community.

As has been identified in the previous chapter, when athletes run, a different mechanism, usually represented as a spring-loaded weight [35], comes into effect. From Chapter 5, this mechanism appeared to be strongly influenced by the athlete's leg-length and mass, which directly or indirectly (via step-rate) affects the recorded acceleration. This leads to an acceleration signal that, across a group of subjects, varies widely in magnitude and shape. In clinical biomechanics, various parameters are normalised in various ways [36] and plotted against the gait-cycle for use in identifying subjects inside and outside a particular group. Duhamel et.al (2004) [37]

\(^{44}\) The language alone is complex and technical, Ounpuu (1994)[38] attempted to standardise some of the terminology. The number of biomechanical variables that can be measured is vast and studies may need to define their own measures (as in [25]). In this chapter, technical language is kept to a minimum except for those terms that are readily understandable.
identifies three different statistical tools used to (a) measure the reliability of the gait curves, (b) classify a subject as belonging to a group and (c) compare two populations.

**Q.2. Can the CoM acceleration signal be normalised such that a group of like-subjects will produce similar graphical results, and therefore allow subjects to be measured within or outside a standardised group?**

**Q.3. Further, regardless of the ability to normalise the CoM acceleration, can this acceleration, and its biomechanical affect, be categorised such that the CoM acceleration signal can be used to partially define a subject's running style?**

Ayyappa (1997)[39] stated that the primary goal [of gait] was 'energy efficiency in progression'. The mechanism of achieving energy efficiency during walking is described as the inverted pendulum model where energy is transferred back and forth between kinetic energy and potential energy[35]. During running the mechanism changes from the inverted pendulum model to a bouncing gait model where energy efficiency is determined by the muscle's ability to capture, store and reuse energy. That this mechanism is effective is visible in the linear relationship between speed and energy expenditure during walking and sub-maximal running recorded in Chapter-5. Farley & Ferris (1998)[35] identified that in the spring loaded weight model, as running speed increased, vertical movement reduced due to increased leg sweep angle. At the same time leg stiffness reduces as running speed increases but reducing leg stiffness increases vertical movement. Together this implies that as speed increases AP-movement increases with a consequent increase in AP-acceleration.

**Q.4. Does the CoM acceleration indicate a reducing V-displacement and increasing AP-displacement as running speed increases? Further, is there any indicator in the CoM acceleration to suggest a relationship to leg-stiffness?**

Elliot & Blanksby [21] compared treadmill and overground running of both male and female subjects using cinematography. Whilst in this particular study, the same subjects performed both overground and treadmill running, the results can be compared with results obtained from the available accelerometer studies. The most notable results occurred at higher running speeds where treadmill and overground
results diverged. Compared to overground running, treadmill running at 12 km h\(^{-1}\) and greater resulted in lower vertical and horizontal in-stride velocities, at 17.6 km h\(^{-1}\) and greater there was a reduced stride length (and consequent increase in stride rate) and reduced flight time. At 23 km h\(^{-1}\), there were longer periods of support as well as less variability in horizontal velocity. Mann & Hagy [26] identified sequentially reducing stance period amongst track runners as they progressed from walking (62%) to running (31%) and sprinting (22%).

**Q.5. Do the accelerometer results indicate greater stance time for treadmill running compared to overground running? Do the treadmill results indicate any difference in step-rate compared to overground results? Is there any indication of lower vertical and horizontal in-stride velocities for treadmill running?**

While these questions may not be directly answerable (since the accelerometer trial does not include the same subjects undergoing both forms of running), the results may be indicative.

Ayyappa (1997)[38] suggested that for a person of normal height walking, the CoM moves within a square ±2.5cm vertically and ±2.5cm in the mediolateral direction. Assuming a sinusoidal oscillation, if these values existed for an athlete running with a step-rate of 3.2Hz, they would correspond to an acceleration of ±10.1 m s\(^{-2}\) on the V-axis and 2.5 m s\(^{-2}\) on the ML-axis (Fig. 6.1). For non-sinusoidal oscillation the acceleration would be greater. Further Ayyappa suggests for a person of normal height walking, a pelvic rotation of ±4 degrees and a pelvic tilt of ±5 degrees. For triaxial accelerometers, these values correspond to approximately 7% and 9% leakage of the instantaneous acceleration in the anterior-posterior and vertical directions into the mediolateral accelerometer. There is a corresponding loss of approximately 0.2% and 0.4% of signal from accelerometers on the affected axes.

These values are used for comparative purposes.
6.3 Biomechanical Analysis using CoM Accelerometers

Accelerometer analysis treats the acceleration data in several different ways:

1\textsuperscript{st}, there was extraction of data that was more or less impervious to calibration and orientation errors, such as step-rate. This was covered in previous chapters.

2\textsuperscript{nd}, there was extraction of orientation information where as much of the dynamic content as possible was ignored. Speed dependent orientation changes, which are a biomechanical function of running, were documented in this section.

3\textsuperscript{rd}, there was the dynamic in-stride acceleration signal itself and its kinematic relatives, velocity and displacement. This analysis was performed in rather more depth than other forms. In-stride physical displacement and velocity were used in various studies; the relationships of stance and flight phases to acceleration, velocity and displacement also came under scrutiny. This led back to one of the original problems associated with the use of accelerometer data without any frame of reference. The recent study of Pfau, Witte & Wilson (2005)[31] answered this question with relation to the accelerometer-derived displacement. In that study the authors monitored horses engaged in running with multi-sensor devices (MEMS gyroscopes, accelerometers, and magnetometers) and external FOR cinematic systems. The accelerometer data was corrected for dynamic in-stride tilt and lean using information from the gyroscopes and magnetometers and displacement was derived. The accelerometer-derived displacement closely matched the values obtained from the external system (depending on the axis - worst case approximately ± 6%).

Fig. 6.1 Acceleration vs. Frequency of Oscillation for a ±2.5cm sinusoidal displacement. For V or AP-acceleration use the step rate, for ML-acceleration, use the stride rate as the frequency of oscillation.
In the data presented here, in-stride acceleration, velocity and displacement were extracted from the raw acceleration by the use of Hamming-Windowed Finite Impulse Response (FIR) digital filters. The Pfau study utilised Butterworth filters but for the purpose of this chapter, this difference was of no consequence.

The analysis in this chapter is predominately a narrative description of graphically formatted data. Although there are numerous methods for normalising biomechanical data, such as normalising by weight, leg-length, step-rate etc (see [37]), the wide range of outputs from runners monitored by CoM accelerometry means that these techniques may not be appropriate in this analysis. Using the multi-speed peak-to-peak vertical acceleration as an example, attempting to normalise this data by step-rate, leg-length, mass or various combinations of these parameters resulted in some marginal reductions in the group standard deviation at some points of the curve. Some combinations of normalising factors resulted in increased divergence from the mean.

Content in this section covered the following areas:

- Investigation of relationship of accelerometer derived acceleration, velocity and displacement and various rules of thumb that could be applied when interpreting this data. This section also identified if any distortion had been introduced by the applied signal processing.
- Consistency of running action at different speeds.
- Analysis of changes to orientation due to treadmill running speed.
- Analysis of acceleration for a single subject (various processing steps).
- Cross-subject analysis of acceleration data.
- Analysis of acceleration data combined with in-sole sensor data
- Comparison between overground runners and between overground and treadmill running.

### 6.3.1 In-stride acceleration, velocity and displacement relationship

The output of the triaxial accelerometers represents acceleration therefore the numerical integration of accelerometer output should represent velocity. Similarly, the numerical integration of the velocity output should represent displacement. Due to the inability to extract pure dynamic information from a static FOR, any attempt to extract
longitudinal acceleration, velocity or displacement, will be subject to large cumulative error (other error sources also contribute to overall error). By integrating the high-pass component of the acceleration signal, a dynamic in-stride velocity can be extracted. This signal is still subject to cumulative error with the magnitude of the offset changing over time. In practice, it is necessary to high-pass filter the signal at each stage.

The offset removal filter would generally be around 0.9-1.0 Hz although higher filter frequencies improved the overlaying of a sequence of steps due to the removal of small inter-step lateral drift. As shown in section 4.2.10.2, during running there was very little signal below the stride rate. In two of the examples in Sections 6.3.4.4 and 6.3.4.5, a 0.9 Hz filter was used, in two other examples a 1.3Hz filter was used (subject stride-rate 1.55 Hz). Provided the stride rate was known, the filter frequency could be chosen to match.

For some analysis, the use of acceleration alone may be sufficient. Assuming a sinusoidal acceleration, some simple rules of thumb are available, eg. velocity is 90 degrees out of phase with acceleration and displacement 180 degrees out of phase with acceleration. This leads to other more 'biomechanical' rules of thumb, eg.:

- If toe-off occurs when V-acceleration is zero, V-velocity is a maximum.
- If acceleration is a minimum, displacement is a maximum, etc.

This approximation is close but, as Fig. 6.2 indicates, not sufficiently accurate in some cases. In this example the acceleration peak does not exactly correspond with the related features in the velocity and displacement values.

To generate displacement data of Fig. 6.2 (or such as Fig. 6.27), the original acceleration was filtered (1.3Hz) to remove the orientation signal, integrated to generate velocity, re-filtered (1.3Hz) to remove the increasing velocity offset, re-integrated to generate the displacement and finally re-filtered (1.3Hz) to remove the growing displacement offset.
Has this processing significantly distorted the signal?

To estimate the effect of the repeated filtering, the displacement signal was differentiated twice and then added to the orientation offset removed in the first step. The original acceleration and the reconstituted acceleration were graphed together in Fig. 6.3. Despite discarding the output from two filter stages, the alignment of the two signals was so close that it was necessary to graph the reconstituted signal as a dashed line to make the original signal visible underneath. Small discrepancies were visible when studying fine detail in the signals. This was of no consequence for the analysis of the data.

The processing required to extract in-stride velocity and displacement does not appear to affect the signal detrimentally.
6.3.2 Consistency of Running Action.

In generating gait-cycle normalised multi-speed graphical data (Chapter 4 Section 4.3.7), an intermediate step required the averaging of a number of successive strides. This 'average' stride was then used to represent the particular running speed for which the data was collected. For some athletes, the consistency of running action varied across the speeds. In Fig. 6.4 following, four successive strides (left/right step pairs) at each sample speed were used to generate the representative signal. Across the running speeds this subject exhibited good consistency of movement although at 21kmh\(^{-1}\) a number of small discrepancies were beginning to appear, notably, there was a small drift in step-duration. Kivi et.al [22], in a study of sprinting on a treadmill, identified the breakdown in action as the subject approached their maximum sprint speed. In that study the speeds attained approached 35 kmh\(^{-1}\), whereas in this study the maximum speed was 21 kmh\(^{-1}\). Considering the generally consistent action across the tested running speeds, small variations at 21 kmh\(^{-1}\) may simply indicate that this particular speed does not fit with the subject's natural cadence. In some cases the entire action changed at 21 kmh\(^{-1}\), this may be indicative of the change over from running to
sprinting for that particular subject. Some athletes appeared to have had difficulty holding a consistent action or step-rate at one speed but were more consistent at a higher speed. In some cases it was necessary to try different sampling points as the variability in step-rate made it difficult to generate an appropriate average signal.

Analysis of the consistency of running action would be conducive to mathematical reduction to a defining parameter such as an error number or other statistical value. Graphical output such as Fig. 6.4 requires an appropriate number of successive strides to generate the 'average' stride for a particular speed. Too many stride cycles and the average stride retains few features. Too few cycles and the average stride does not adequately represent the running action at that speed.

To allow cross subject comparisons, both the gait cycle and acceleration magnitude could be normalised.

**Potential biomechanical measures:**
- Step-rate variability
- Error measure of variability of acceleration over gait cycle.

![Graphs showing acceleration over time for different speeds](image)

**Fig. 6.4** Averaging of sample strides. In this example each of the four graphs represent a treadmill running speed and each graph includes four successive strides aligned and overlaid. The fifth (dark) trace represents the average of the four strides. Generally the successive strides are consistent in shape and magnitude but at 21kmh$^{-1}$ some inconsistency in successive strides is occurring. (Subject A5).
6.3.3 Angle of Lean

To move data from a FOR directly related to major axes of the athlete body (Fig. 6.5(a)) and convert this to a FOR related to the orientation of gravity (Fig. 6.5(b)), required the extraction of the current orientation of the sensor platform. In visual monitoring during multi-speed treadmill running data\(^{45}\), it was observed that some runners leaned forward as they ran, (as in Fig. 6.5(a)), while other runners ran with an upright action. From sensor orientation analysis of the multi-speed treadmill data it was observed that for most, but not all subjects, this angle was speed dependent. Although this data is from treadmill running, the angle of lean can be extracted from any data. The systematic nature of the treadmill data sets allowed multi-speed analysis that would be difficult to replicate in overground running. Because the shape of the subject's body and the subject's stationary posture (during the trial) affected the quiescent angle of the accelerometer, it is not possible to estimate actual lean angle unless the subject was calibrated for upright stationary posture.

![Fig. 6.5 Accelerometer alignment: A body-mounted accelerometer may align as in (a) resulting in both the AP and V axes mixing their signal. Rotation of the triaxial accelerometer data moves the accelerometer data away from direct alignment with the body (a) and back into alignment with gravity (b). This assists in isolating the vertical signal from the anterior-posterior signal.](image)

![Fig. 6.6 (a) Angle Y, Rotation about ML axis. (b) Angle Z, Rotation about AP axis.](image)

\(^{45}\) Treadmill angle was not changed at any time during the tests.
**Extracting Angle:**

In the following examples the angle Y refers to the rotation about the mediolateral axis of the triaxial accelerometer (Fig. 6.6(a)) and the angle Z refers to the rotation about the Anterior-posterior axis (Fig. 6.6(b)). In Fig. 6.5, the difference between (a) and (b) is a change in angle Y. The Y & Z angles were extracted from the low-pass orientation signal. To obtain the change in angle as running speed increases, the low pass data was sampled at the different running speeds. Trunk lean angles (both Y & Z), correlation of lean to running speed and change in angle from 9kmh\(^{-1}\) to 21kmh\(^{-1}\) were recorded in Table 9.16.

![Image](image.png)

Fig. 6.7  Change in off-normal angles with change in running speed (Subject A-5).

Generally the Y lean angle had a strong correlation with speed (Fig. 6.8 & Table 9.16) although one subject (A3 in Fig. 6.8 and Table 9.16) recorded no angle/speed correlation. This subject maintained a consistent orientation across all speeds. The Z angle also recorded some good angle/speed correlation although the smaller change in angle and the mix of positive, negative and weak and strong correlations suggested that the change in Z was probably a function of the change in Y and the general orientation of the sensor system.

![Image](image.png)

Fig. 6.8  Correlation coefficients (as R\(^2\)) for Y & Z angle compared to running speed.

6-18
Potential biomechanical measures

- Equation of Angle vs. Speed regression line.
- Regression line error value.

The equation for the regression line and the associated error provide information about how the subject adjusts lean as speed changes.
6.3.4 Treadmill Acceleration & Derived Information Analysis

This section traces the processing of the data from a single subject during treadmill running. Where this data appeared significant it was compared across the test group. For graphical analysis, comparisons were made between different athletes engaged in treadmill running. To assist in identifying the stages of the gait cycle, data from synchronously collected in-sole and CoM accelerometers was also analysed.

6.3.4.1 Output from Processing Steps.

Considering the large quiescent Y angle (Table 9.16) that existed in most cases, it was apparent that the V & AP acceleration data was intermixed. To assess the impact of this intermixing, the data from one subject was processed to remove the Y & Z tilt. Where it appeared that the re-orientation of the data changed the form of the extractable information, data from other subjects in the trial data was analysed. This data appears below. The sequence of graphical representations is as follows:

- Calibrated triaxial accelerometer output for ten speeds Fig. 6.9.
- 1Hz low pass filter output (orientation signal) Fig. 6.10.
- 1Hz high pass filter output (dynamic signal - with signal envelope) Fig. 6.11.
- Rotated dynamic signal (with signal envelope) Fig. 6.12.
- Detailed comparison rotated and non-rotated acceleration data Fig. 6.13.
- Multi-speed Peak-to-Peak (p-p) acceleration all axes Fig. 6.14.
- Correlation coefficients for p-p acceleration vs. speed (all subjects, all axes) (non-rotated data Fig. 6.15) (for rotated data Fig. 6.16).
- Gait-cycle normalised multi-speed data, non-rotated Fig. 6.17, rotated Fig. 6.18.
- Effect of pelvic-tilt on V-acceleration magnitude Fig. 6.19.
- Gait-cycle normalised multi-speed data from different subjects for comparison (Subject A-3 Fig. 6.20 and Subject A-9 Fig. 6.21).
Fig. 6.9  Calibrated acceleration data (Subject A-5) for each accelerometer channel. Walking and running speeds marked against the AP acceleration.

Fig. 6.10  Low pass filter (1.0Hz FIR Hamming Window) output from acceleration data (Subject A-5). Note the leakage of ML walking signal (around 20-30 seconds) into the low pass acceleration.
Fig. 6.11 Acceleration signal from high pass output of 1Hz FIR filter. Envelopes have been added to the acceleration signal. The envelopes start growing as speed increases but then level out, so some acceleration magnitude vs. speed correlation would appear to exist (Refer Fig. 6.15 for actual correlation).

Fig. 6.12 After Rotation: Acceleration signal from high pass output of 1 Hz FIR filter (Subject A-5). Envelopes around the ML and AP acceleration suggest a correlation between acceleration magnitude and speed may exist (Refer Fig. 6.16 for actual correlation).
Fig. 6.13  Raw data vs. rotated data. The mediolateral channel is virtually unchanged; the vertical channel has some small changes but is relatively unchanged. The anterior-posterior channel is substantially different after removal of the influence of the vertical signal component. This is reflected in the changes in the envelope in Fig. 6.11 compared to Fig. 6.12.

Fig. 6.14  Peak-Peak Acceleration after rotation (from averaged waves) (Subject A-5) for all three axes. Compared to pre-rotation results, the V acceleration lost some correlation with speed but the AP axis increased correlation.
Most athletes had at least one axis showing good to strong correlation with speed, although there was no consistency in which axis showed the most correlation.

The graphical and tabulated results suggest that the rotation to realign the FOR such that the 'vertical' axis is aligned with gravity, improves the interpretability or correctness of the data. Where some sort of correlation between speed and acceleration may have been expected, the raw data was such that there was no consistency in results. After rotation, the group of subjects in this particular trial show a consistent acceleration/speed correlation in the AP axis.

This adjustment to the FOR resulted in substantial differences in the AP signal (Fig. 6.13), which were reflected in changes in the multi-speed gait-cycle normalised representation of the acceleration (compare Fig. 6.17 & Fig. 6.18).
In Fig. 6.17, leakage of AP acceleration into the V axis resulted in the systematic increase in acceleration shown in the V axis. The AP signal for this subject, and for most others in the treadmill trials, showed an underlying sawtooth waveform with some superimposed speed dependent changes. From analysis it appeared that this sawtooth waveform was predominately the trigonometrical 'leakage' component of the vertical acceleration - combined with the generally smaller AP acceleration. For this subject, the negative-going speed related changes that occurred at approximately 50 and 230 degrees are actually the AP acceleration superimposed on the sawtooth waveform. After FOR rotation to align with gravity (Fig. 6.18), it is apparent that these points coincided with the peak AP acceleration.

*Does realigning the FOR assist in interpreting the acceleration from a biomechanical viewpoint?*
Rotated Data Biomechanical Analysis:

In the rotated data of Fig. 6.18, the *vertical acceleration* was consistent in magnitude and shape across the different speeds with a small amount of left/right asymmetry visible (variation between the first cycle and the second cycle).

The *mediolateral acceleration* displayed the left/right asymmetry that is typical of the mediolateral sensor, since this sensor operated with a fundamental frequency of the stride-rate whereas the other two sensors had their fundamental frequency at the step-rate. Despite the asymmetrical nature of this acceleration, it was observed that the peak-positive ML-acceleration occurred at approximately 25-30 degrees into the gait-cycle and the peak negative ML-acceleration at approximately 200-210 degrees. Both of these acceleration peaks occurred immediately prior to the positive peak V-acceleration. There were subsidiary peaks at 150 and 330 degrees - both corresponding to immediately after the negative peak V-acceleration.

The *anterior-posterior acceleration* showed very distinct speed dependent behaviour. The AP-acceleration (at 50 and 230 degrees) peaked immediately after the peak V-acceleration.
acceleration. As speed increased the forward AP-acceleration increased. Between each propulsive peak, there is a period where the AP-acceleration from all speeds follows the same pattern.

In Pfau [31], the accelerometer data was corrected for tilt along various axes to allow the accurate extraction of displacement along the 3 axes. In Fig. 6.19 the ML-acceleration and the V-acceleration have been trigonometrically combined and compared to V-acceleration. Despite the magnitude of the ML-acceleration, it only makes a very small difference, if any, to the V-acceleration magnitude. To estimate if accelerometer tilt (and hence athlete pelvic-tilt) is a feasible source of ML-acceleration, Fig. 6.19 includes the calculated angle between the V and ML acceleration during the V-acceleration peaks. Note that on every second step the angle has been flipped so that all angles appear positive. Angles during the peak sometimes reach to nearly 30 degrees. As a rule of thumb - the angle is only relevant if the V+ML vector magnitude is substantially different to the V magnitude.

Combined, these acceleration patterns described a combination of forces applied to the triaxial accelerometer. The sensors oscillated in the vertical plane, accelerated forward in short-sharp pulses, presumably as a result of forces applied to the trunk by the power of the legs during contact with the ground. From Fig. 6.1 and Fig. 6.19, the ML-acceleration can be explained by a combination ML-displacement and pelvic-tilt. For this subject the ML-acceleration was approximately half of the value of the V or AP acceleration. It was considered that the most likely source was tilting of the hips (observed in video footage) causing the ML accelerometer to detect and record a proportion of both gravity and V-acceleration. At the point of maximum ML-
acceleration, the V-acceleration was approximately $20\text{ms}^{-2}$ (gravity plus approximately $10\text{ms}^{-2}$ V-acceleration). A 30-degree pelvic-tilt, in the absence of any other force would be sufficient to account for the ML-acceleration.

At $21\text{kmh}^{-1}$, the subject of Fig. 6.18 was running with a step rate of 3.2 Hz and a stride rate of 1.6 Hz. From Fig. 6.1, a small lateral movement of $\pm 2.5\text{cm}$ would result in an acceleration of $\pm 2.5\text{ms}^{-2}$ on the ML-axis (assuming sinusoidal operation (refer Fig. 6.2)). The peak-to-peak ML-acceleration is more than three times that value, indicating a value that does not appear reasonable ($\pm 7.5\text{cm}$ or $15\text{cm}$ peak-to-peak). In the absence of any additional information on this issue, it can only be assumed that the ML-acceleration is a combination of pelvic-tilt and lateral movement. It is possible that for some, or all, athletes, pelvic tilt and lateral movement counter each other at some point in the gait cycle.

**Non-Rotated Data Analysis:**

The non-rotated data exhibited some close similarity to the rotated data, as well as some striking differences.

The V-acceleration was very similar to the rotated date except that the peak acceleration exhibited some speed dependent behaviour. The left/right symmetry was maintained.

The ML-acceleration was essentially unchanged compared to the rotated date, maintaining the same left/right behaviour in the same relationship with the vertical acceleration.

The AP-acceleration of the non-rotated data bore no resemblance to the AP-acceleration of the rotated data. Speed dependent behaviour was observed but not in the same places as previously. The apparently propulsive spike was missing and where the rotated data had a maximum acceleration, the non-rotated data was around zero. There was a different phase relationship between the V & AP acceleration and the peak acceleration was considerably less.

**Discussion:**

The first point to note is the disclaimer that neither FOR is consistent within the stride cycle. The orientation of the sensors was extracted by a 1Hz filter. This filtering could only extract the average orientation leaving a signal that consisted of both applied
forces and dynamic orientation changes. Similarly, the rotated data could only bring the FOR into average vertical/lateral orientation. Dynamic orientation changes still occur. Both types of graphs (non-rotated vs. rotated) describe the same orthogonal data.

**Mediolateral Acceleration:** Given that in both sets of data the ML-acceleration remains consistent, in conjunction with the approximately zero value in the 1Hz filter output (Fig. 6.10), it could be assumed that the ML alignment of the accelerometers was very close to the horizontal and at the same time normal to the vertical and AP movement (both before and after rotation).

**Anterior-Posterior Acceleration:** Using the orientation of Fig. 6.5(b), a sensor rotated about the ML axis (change in angle Y) would record a combination of negative vertical acceleration superimposed on the AP-acceleration. Again, depending on the angle and relative magnitudes of the acceleration, the resultant recorded acceleration may not exhibit the expected behaviour. The recorded V-acceleration, while marginally reduced due to leakage to the AP-accelerometer, would gain extra acceleration during the forward acceleration peaks.

Depending on the extent, additional complexity in the signal would occur due to the combination of pelvic-tilt and pelvic-rotation. In the graphs of raw (non-gravity aligned) data, the combination of AP & ML-acceleration for some subjects, suggested that complex pelvic rotation and tilt information might be extractable.

**6.3.4.2 Comparisons of Different Subjects**

To identify if there was any biomechanical information of interest it was necessary to identify differences and similarities in the acceleration signals. For comparison Fig. 6.20 & Fig. 6.21 were produced using the same processing as Fig. 6.18. These figures were accompanied by analysis similar to that above.
Fig. 6.20 Rotated data: Gait Cycle Normalised Acceleration (Subject A3) for treadmill speeds 9-21 km/h. Systematically increasing acceleration marked with arrows. (Bold trace = 21 km/h)

**V-acceleration:** Asymmetric vertical acceleration with the first step (left step) indicating a lower maximum acceleration, a decrease in acceleration (around 60 degrees) as running speed increased and a delayed curve as speed increased.

**ML-acceleration:** Left-right symmetry, distinctive shape, peak magnitudes similar to vertical acceleration however the left peak had a distinctive speed dependency and phase shift as speed increased, this was not apparent on the right contact.

**AP-acceleration:** Asymmetric with the left step (60 degrees) not reaching the same magnitude as the right step (240 degrees).

**Combined:** Vertical peak occurred first (approx 30 & 210 degrees) followed by the ML peak (30-45 degrees & 210-225 degrees) and the AP peak (60 & 240 degrees). Viewed in isolation, the interaction between the V, ML and AP for the left step suggested that the increasing ML signal was due to increasing pelvic tilt resulting in more ML acceleration and less V acceleration. This may be true, however comparing the left step to the right step; the right step has larger ML acceleration, larger and more consistent V acceleration and a systematically increasing AP acceleration. The combination may indicate a weakness in the left step.
V-acceleration: Symmetric V-acceleration with distinctive speed related changes. At low speed the V-acceleration is large (suggesting a large V-displacement) followed by a plateau. As speed increased the V-acceleration became more sinusoidal. The phase of the peak acceleration changed with the increase in speed.

ML-acceleration: Small signal with no apparent left/right symmetry. This signal suggests that any left/right hip movement or tilt is small. Reviews of video of this subject strongly suggested that the peak between 90-105 degrees was due to a distinct lift in the hip during or just after toe-off.

AP-acceleration: Generally left/right symmetrical acceleration with distinctive speed related changes. In particular, at lower speeds the positive AP-acceleration is spread over approximately 90 degrees but shortens to approximately 60 degrees as speed increases.

Combined: The V and AP acceleration peaks appear to occur together, starting at around 30 degrees at low speed and sliding back to around 45 degrees as speed increases. The unusual shape of the AP & V acceleration at low speed may be due to this subject ‘riding’ the treadmill, that is, the subject appears to slide back and then leap forward.
Cross-subject comparison:

A number of parameters of possible interest were identified in Table 6.1.

Table 6.1 Multi-subject Acceleration Comparison

<table>
<thead>
<tr>
<th>Subject</th>
<th>First Peak (Degrees)</th>
<th>Peak Order (degrees)</th>
<th>Magnitude min, max, (total) ms⁻²</th>
<th>Left-Right Symmetry</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>V</td>
<td>ML</td>
</tr>
<tr>
<td>A5</td>
<td>25</td>
<td>ML-V-AP</td>
<td>-16 +13 (29)</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-7 +9 (16)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-5 +10 (15)</td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td>25</td>
<td>V-ML-AP</td>
<td>-14 +14 (28)</td>
<td>Mag. Shape</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-10 +10 (20)</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-5 +8 (13)</td>
<td>Mag.</td>
</tr>
<tr>
<td>A9</td>
<td>30-45</td>
<td>(V-AP)</td>
<td>-15 +20 (35)</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-7 +7 (14)</td>
<td>Shape</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-7 +11 (18)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Note:** For subject A9, the first peak changes position depending on running speed, in the peak order, V & AP occur together. **Symmetry:** waves either show left-right symmetry or are not symmetrical due to a variation in magnitude (Mag.) or due to a variation in shape, or both. Subject A5 has some left-right symmetry but at low speeds the signal is small and indistinct.

From this small sample, variability in acceleration patterns is visible and the identified variations occur throughout the whole sample set. Some subjects exhibited good left/right symmetry, some subjects exhibited consistent acceleration patterns across the entire speed range where all the vertical acceleration traces overlayed each other (as in Fig. 6.18).

**6.3.4.3 Comparison Tri-Axial Acceleration & In-sole Pressure**

As yet these patterns do not appear to indicate any particularly useful biomechanical activity. Larger acceleration can indicate larger displacement and in-step velocities however acceleration includes a step-rate squared component. Mediolateral acceleration could equate to either pelvic tilting or lateral movement or both. Video of some subjects indicates substantial pelvic tilting. In isolation, CoM triaxial accelerometry is suggestive of possible biomechanical activity. By comparing the acceleration directly with an external reference such as in-sole pressure sensors, the acceleration can be directly referenced to known biomechanical analysis. Fig. 6.22
includes multi-speed gait-cycle normalised CoM triaxial accelerometry and in-sole pressure. The in-sole pressure is for the left foot only.

Fig. 6.22 Gait cycle normalised multi-speed acceleration & in-sole pressure (speeds 12km\(^{-1}\) - 18km\(^{-1}\)) (gravity aligned)

**Analysis of Fig. 6.22:**

There was an approximate delay of 15 degrees between initial foot contact and the V-acceleration positive-going zero-crossing (330°-360°). Immediately after the heel contact there was a sharp negative-going AP-acceleration suggesting braking on heel-strike. This was apparent in left and right steps at all running speeds. Reviewing Fig. 6.20 & Fig. 6.21, there was a suggestion of a sharp negative AP-acceleration immediately before the AP peak. In the light of the analysis of Fig. 6.22, it was possible that this negative AP-acceleration was due to the heel-strike causing AP braking.

Toe-off or final contact appeared to occur on or about the V-acceleration negative-going zero-crossing. Using a rule of thumb - this equates approximately to peak V-velocity (refer Fig. 6.2).
This subject had a very distinctive double peak in the vertical acceleration. The near alignment of the large ML-acceleration peak (40°, 215° & 400°) to the trough in the V-acceleration peak suggested that this could be attributed to pelvic tilt. While this might be true, analysis revealed that large tilting, sufficient to generate the entire ML-acceleration spike, was insufficient (trigonometrically) to account for the trough in the V-acceleration. Tilting may account for the trough through some biomechanical change such as stiffness reduction or some lever action. As the in-sole pressure did not indicate the double peak and trigonometry couldn't account for it, some form of mechanical damping action between the foot and trunk was indicated. From the in-sole information, the stance phase was around 30% of the gait cycle. This was consistent across the speeds 12-18 km$h^{-1}$.

**6.3.4.4 In-Stride Velocity by Integration**

By processing the acceleration from the various data sets reviewed, velocity data was generated. This data was graphed as 2-dimensional representations of the in-stride velocity viewed along each of the three axes (V-axis is gravity aligned). Where insoles sensors indicated stance phase this was included in the graph.

![Graph of CoM trunk velocity with left-foot contacts marked. Initial contact identified by circle marker. Treadmill running speed 18km/h. Noise filter 25Hz. 1.0Hz offset removal filters. SR=3.05Hz, Period = 0.328s, contact = 0.192s, flight = 0.135s. Contact estimated as 2 x left contact period. Stance = 29%. For this subject at all tested speeds, initial contact occurred just prior to maximum negative vertical velocity and toe-off occurred at, or immediately after peak positive vertical velocity.](image)
Analysis of Fig. 6.23 derived velocity with contact markers:
The contact marking in Fig. 6.23 gave the figure a definitive orientation with respect to the stance phase. By identifying the initial foot contact (marked with an asterisk in a circle), the sequence or time series of the contact phase was identifiable.

- This single-speed figure was representative of the 2D velocity across different speeds for this subject (consistent shape with some small variations in magnitude).
- Initial contact always occurred before the maximum negative V-velocity.
- Toe-off occurred at maximum positive V-velocity or just after maximum V-velocity at the lower speeds.
- This athlete exhibited relatively little forward and back movement.

Analysis of Derived Velocity (no contact markers) Fig. 6.24:

Fig. 6.24 2-Dimensional views of velocity @ 17kmh

- At a similar running speed to the previous example, this subject reached higher velocities in each dimension.
- This figure was generally representative of the in-stride velocities at different running speeds for this subject (both in magnitude and shape).
- It was difficult to identify the contact sequence without contact markers on the graph. For the V vs. ML graph, an approximation could be made but the other graphs were far less obvious.

The contact markers of Fig. 6.23 assisted in assimilating the information associated with the graphs however in-sole sensors are not widely used. As an alternative to
synchronously collected in-sole sensor data, and for the purpose of tracing time sequence on velocity or displacement graphs, the stance period can be estimated from the acceleration graph. Using an estimated initial contact point at 15-30 degrees prior to the V-acceleration positive-going zero-crossing, and a final contact point at the V-acceleration negative-going zero-crossing, would be sufficient for the purpose of identifying time sequence and contact phase. This technique has not been reproduced here, as it would be indistinguishable in effect from results already presented.

Comparing Fig. 6.14, the P-P acceleration for Subject A-5 at speeds 9-21 kmh\(^{-1}\), with Fig. 6.25, the derived P-P velocity for the same subject across the same range of speeds, the in-stride V-velocity was decreasing as treadmill speed increased and both the AP & ML in-stride velocity increased. The effect on the V-velocity could have been directly estimated from the multi-speed gait-cycle normalised results (Fig. 6.18) and the implications of Eqn.5.2 & Eqn.5.3. In Fig. 6.18 the vertical acceleration, which includes a step-rate squared component, maintained a consistent magnitude and shape across the speeds, given the increasing step rate (Chapter 5, Fig.5.4) as speed increased, the derived V-velocity must decrease.

Fig. 6.25  In-Stride Peak to Peak Velocity (per axis) at speeds 9-21kmh\(^{-1}\) (Subject A-5)
6.3.4.5 In-stride Displacement by Integration

The 2-dimensional in-stride displacement figures view the displacement along each of the three axes. The contact marking in Fig. 6.26 gave the figure an orientation with respect to the stance phase. By identifying the initial foot contact (marked with an asterisk in a circle), the sequence or time series of the contact phase was identifiable.

Analysis of Fig. 6.26 (Treadmill running 18km\(^{-1}\) with in-sole pressure sensors):

- Despite the apparent near symmetry of the V vs. ML velocity graph, the V vs. ML displacement graph showed considerable left/right asymmetry.
- The peak vertical position occurs offset from the accelerometer centreline.
- The initial left foot contact occurred as the trunk was dropping, the contact continued as the trunk reached its lowest point and continued as the trunk rose again. Final contact occurred as the trunk rose past the midpoint of the oscillation. This movement represents the eccentric, isometric and concentric phases as the downward movement of the body is resisted, stopped and reversed.
- This subject tends to only move up and down on the spot, there is very little anterior-posterior movement.

![Fig. 6.26 CoM trunk displacement with left-foot contacts marked. Initial contact identified by circle marker. Treadmill running speed 18km/h. Noise filter 25Hz. 1.0Hz offset removal filters. SR=3.05Hz, Period = 0.328s, contact = 0.192s, flight = 0.135s. Contact estimated as 2 x left contact period. Stance = 30%.

6-37
Analysis of Fig. 6.27 (Treadmill Running 17km$^{-1}$ Subject A-5):

- This subject displayed some left/right asymmetry in the AP and V directions.
- V-displacement peak occurred along the accelerometer centreline.
- AP-peak occurred along the axis in the direction of travel.
- This subject moved backward and forward on the treadmill, not a great distance but considerably more than the subject of Fig. 6.26.

Comparing P-P displacement as treadmill running speed increased (Fig. 6.28), the V-displacement reduced - just as the V-velocity reduced in Fig. 6.25. The increasing ML-displacement strongly supports the belief that a considerable portion the ML-acceleration is pelvic-tilt related. The AP-displacement remained consistent across the speeds from 9-21 km$^{-1}$. This does not match the suggestion in [35] (Fig.10.8 p.266) that as speed increased the AP-displacement during contact would also increase. This could simply an artefact of treadmill running. The two dimensional views of AP/V displacement suggest that during treadmill running the body tends to do more up/down movement than forward/back movement.

![Fig. 6.27 2-Dimensional views of displacement @ 17km$^{-1}$. For this subject, running speeds of 15-21km$^{-1}$ all had similar results. (10x10 Subject A5, No noise filter, multiple 1.3Hz offset removal filters, step-rate 3.1Hz, Stride-rate 1.55Hz.)](image)

![Fig. 6.28 In-Stride Peak to Peak Displacement (per axis) at speeds 9-21kmh-1 (Subject A-5)](image)
To further check the change in AP-displacement with change in treadmill speed, the P-P AP-displacement for all 10x10 subjects was graphed in Fig. 6.29. As in many other areas, the resultant data was unique to the individual. Some subjects started with a high AP-displacement that reduced as speed increased, other subjects started with a low AP-displacement that increased as speed increased. There was no definitive result.

![Fig. 6.29 Comparison of P-P displacement on the AP-axis for multiple subjects across a range of treadmill speeds (9-21 kmh$^{-1}$). Some subjects appear to increase the displacement; others remain consistent while others decrease displacement.](image)
6.4 Overground Analysis (1500m Track Running)

This data was collected from two 1500 metre runners doing trial runs. Both subjects were wired with two CoM accelerometers and in-sole sensors on both the left and right foot. The 1500 metre trials were timed including lap timing. The lap-times, step-rates and hallux pressure are recorded in Fig. 6.30. Immediately prior to the end of each lap, the subject is on curved track with the next lap starting on straight track. The curved and straight sections of track are visible in the effect on the left hallux pressure.

![Graph showing step frequency and left hallux pressure over race time](image)

Fig. 6.30 1500m 'Race' comparison of Left-hallux pressure and step-rate. Subject A ran a faster time with a substantially lower step-rate.

The gait-cycle normalised data consists of sample of strides immediately prior to the end of a lap and immediate after the start of each lap. Therefore the samples include strides on both curved and straight track. Orientation (1Hz filter output) did not appear to change between running on the straight and running on the curve. The only clearly identifiable change was the left hallux pressure Fig. 6.30. Lap timing and average speeds are recorded in Table 6.2 (speeds vary between 20.8 and 22.9 kmh\(^{-1}\))
Subject-A 1500m:

Fig. 6.31 Subject A: 1500m Gait cycle normalised acceleration and in-sole pressure. Samples taken immediately before and after the end of each lap (includes straight and curved running).

Fig. 6.32 Subject A: 1500m Derived In-Stride Velocity alone the three axes (with contact marked). Red indicates left-contact, black indicates right contact, squares indicate initial contact, diamonds indicate final contact.
Analysis 1500m Subject A:

From Fig. 6.31, this subject had relatively small heel-pressure compared to other insole pressures. The dip in AP-acceleration immediately following heel-strike was also relatively small when compared to the other subject. This subject exhibited a distinctive double spike in the positive V-acceleration with the sharp leading edge suggesting high leg stiffness. The positive V-acceleration was also left/right symmetrical. There was some variation in the negative V-acceleration. The AP-acceleration was asymmetrical with the left contact period not attaining the same magnitude or shape as during the right contact period. The ML-acceleration was also asymmetrical with very little ML-acceleration during left contact, followed by a small post-contact acceleration but during right contact, the ML-acceleration was considerable followed by a relatively large post-contact acceleration (possible pelvic-tilt). The combined V-ML-AP acceleration during right contact would suggest a large force being applied to the CoM accelerometer except that the foot contact forces during right contact are lower than those during the left contact. This tends to confirm the hypothesis that the large ML-acceleration is due to pelvic-tilt.

During right foot contact the AP and V-acceleration matched in timing, magnitude and shape. Within the inertial frame of reference this would result in a force and movement upward and forward at 45 degrees - the optimal trajectory. This 45 degree up and forward trajectory also appears in the V-AP graphs of velocity (Fig. 6.32) and displacement (Fig. 6.33). Hunter [25] used the vertical velocity and the CoM angle of velocity at take off, both these can be derived from Fig. 6.32. Calculating the angle requires the lateral velocity (from Table 6.2).
Fig. 6.34  Subject B: 1500m Gait cycle normalised acceleration and in-sole pressure. Stride samples taken immediately before and after the end of each lap (includes straight and curved running).

Fig. 6.35  Subject B: 1500m Derived In-Stride Velocity alone the three axes (with contact marked). Red indicates left-contact, black indicates right contact, squares indicate initial contact, diamonds indicate final contact.
Fig. 6.36 Subject B: 1500m Derived In-Stride Displacement along the three axes (with contact marked). Red indicates left-contact, black indicates right contact, squares indicate initial contact, diamonds indicate final contact.

**Analysis 1500m Subject B:**

In general shape the three accelerometer channels exhibited left/right symmetry although the magnitudes varied between the left and right contact. Compared to subject A, the heel pressure was very large and the sharp dip in AP-acceleration immediately following was also substantial. The ML-acceleration has a matching spike indicating either a short, sharp, lateral movement or a consistent pelvic-tilt.

Another feature of interest is the large oscillation in the ML-acceleration during the flight or non-contact period. This large oscillation (between 90 and 180 degrees and 270 and 360 degrees) indicates either a sharp side to side movement or a complete rocking movement, starting with the pelvis flat at toe-off with the pelvis tilting up and down and returning to flat just at initial contact.

In both the derived velocity (Fig. 6.35) and displacement (Fig. 6.36) figures, the V-AP graphs indicate an in-stride trajectory of around 70 degrees. The velocity at toe-off is approximately $0.63 \text{ms}^{-1}$ at an angle of 71.5 degrees. Combining this with an approximate forward velocity of $5.88 \text{ms}^{-1}$ (Table 6.2, sample from the third lap). The overall CoM velocity at toe-off was $6.11 \text{ms}^{-1}$ at an angle of 5.7 degrees. (Compare Subject-A, velocity at toe-off was $6.45 \text{ms}^{-1}$ at an angle of 4.4 degrees)
Comparison 1500m Subjects:

A number of results for the two subjects are recorded in Table 6.2. From these results it was identified that subject A ran with a step-rate approximately 0.2Hz lower than subject B and at the same time ran with a step length 0.2m longer than subject B. This combination of factors lead to Subject-A completing the 1500m 10 seconds faster then Subject-B but with 78 fewer steps.

Table 6.2 Comparison of 1500m results.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Subject A</th>
<th></th>
<th>Subject B</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Laps</td>
<td>Total</td>
<td>Laps</td>
</tr>
<tr>
<td>Flight/Support (deg.)</td>
<td>80:100</td>
<td></td>
<td>80:100</td>
<td></td>
</tr>
<tr>
<td>Lap length (m)</td>
<td>1500</td>
<td>300, 400, 400, 400</td>
<td>1500</td>
<td>300, 400, 400, 400</td>
</tr>
<tr>
<td>Step Count</td>
<td>764</td>
<td>154, 204, 208, 198</td>
<td>842</td>
<td>164, 222, 230, 226</td>
</tr>
<tr>
<td>Step Length (m)</td>
<td>1.96</td>
<td>1.95, 1.96, 1.93, 2.02</td>
<td>1.78</td>
<td>1.83, 1.80, 1.74, 1.77</td>
</tr>
<tr>
<td>Step Rate (Hz)</td>
<td>3.14</td>
<td>3.21, 3.13, 3.11, 3.14</td>
<td>3.32</td>
<td>3.41, 3.32, 3.32, 3.28</td>
</tr>
<tr>
<td>Lap Time</td>
<td>243.20</td>
<td>48.04, 65.28, 66.84, 63.04</td>
<td>253.32</td>
<td>48.08, 66.92, 69.32, 69</td>
</tr>
<tr>
<td>Speed (ms⁻¹)</td>
<td>6.17</td>
<td>6.24, 6.13, 5.98, 6.35</td>
<td>5.92</td>
<td>6.24, 5.98, 5.77, 5.80</td>
</tr>
<tr>
<td>(also in kmh⁻¹)</td>
<td>22.2</td>
<td>22.5, 22.1, 21.5, 22.9</td>
<td>21.3</td>
<td>22.5, 21.5, 20.8, 20.9</td>
</tr>
</tbody>
</table>

Note: Laps are 300m for the first lap and 400 metres for the next three laps.

From the graphs of Fig. 6.31 & Fig. 6.34 and from analysis of the recorded data, two key observations are made.

- Subject A landed with a much flatter foot, the 3rd Metatarsal-Head sensor (3MTH) detected pressure a moment before the heel-sensor. The heel pressure was very small, the Hallux pressure was relatively large and the sequence of pressure peaks ran: 3MTH, Heel, 1MTH, Hallux. During straight running the Hallux pressure was larger than the 1MTH pressure by 10% or less. This combination suggested a flat foot landing with a roll forward to the front foot with the lift and propulsion coming from the 1MTH and the Hallux.

- Subject B contacted with the heel first and the heel pressure was relatively large. Heel strike magnitude is comparable to the 3MTH and Hallux magnitude. The sequence of pressure peaks occurred as: Heel, 3MTH, 1MTH, Hallux. The 1MTH and Hallux peaks were quite close with the 1MTH having approximately twice the pressure of the Hallux. This combination suggested a rear-foot landing with a roll forward to a more flat foot position. The majority of the lift came from the 1MTH.
As noted previously, for Subject B the heel contact appeared to be related to a substantial braking deceleration in the AP-acceleration. This effect was not as blatant with Subject A. Subject A consistently began the positive AP-acceleration earlier and in conjunction with the sharp positive vertical acceleration (which more quickly stops the downward movement of the trunk), the AP and V-acceleration combined to push the trunk upwards and forwards at the same time. With Subject B, the V-acceleration had a slow rise time (taking longer to halt the downwards movement of the trunk) and the V-acceleration was always substantially larger than the AP-acceleration. Calculation of propulsive angles through the decaying positive acceleration, Subject B consistently generated an angle of around 70 degrees, compared to Subject A, who generated acceleration angles in the 45-55 degrees range. This suggested that Subject B tended to waste energy by generating more up/down movement than was necessary.

**Support Phase:**
From the insole data, for both the 1500m subjects, the support phase was generally less than 100 degrees (or about 28%) and closer to 90 degrees (or 25%), for the treadmill subject, support was generally less than 120 degrees (33%) and closer to 110 degrees (30.5%)(a fraction longer at low speed).

*Was there any significant in-stride acceleration or in-stride velocity or displacement that can explain the longer step-length and faster time of 1500m Subject A compared to Subject B?*

In the absence of any other significant factors, there appeared to be two prominent differentiators:

1. The angle of the applied force during the contact phase. Subject-A consistently generated forces that resulted in an in-stride velocity at toe-off at 45 degrees up and forward.
2. The AP-braking at initial contact. While the acceleration data cannot directly identify the initial or final contact, the significant AP-deceleration during the positive-going V-acceleration appears indicative. Subject-B had a substantially larger AP braking spike and this could be assumed to have a detrimental effect on forward speed.
6.5 Discussion

To assist in discussing the results, the questions raised in the chapter are relisted here. These will be addressed more-or-less in the reverse order to which they appear.

Ad-hoc Question: In the 1500m data, does the data indicate that either runner is suffering from fatigue?

The 1500m Gait-cycle normalised graphs were created using sample points from each lap, both on the straight and on the curve. The only identifiable significant differences between any of the traces were those associated with foot contact forces changing between running on the straight and running on the curve. Excluding the first and last dozen or more steps, samples from very early in the 1500m and towards the end did not indicate any significant difference. It should be noted that these runners were both elite level 1500m runners.

Q.5. Do the accelerometer results indicate greater stance time for treadmill running compared to overground running? Do the treadmill results indicate any difference in step-rate compared to overground results? Is there any indication of lower vertical and horizontal in-stride velocities for treadmill running?

Q5:

Without having the same subjects undertaking both overground and treadmill running, parts of this question cannot be answered with the available data. However, from the results it would appear that stance during treadmill running is consistently longer than during overground running. For all the tests that incorporated in-sole sensors, a clearly defined change in AP-acceleration appeared to indicate the initial foot contact and the final contact appeared to occur approximately at the V-acceleration negative-going zero-crossing. Where a clearly definable marker could be seen in AP-acceleration for other treadmill tests (without in-sole sensors), the timing appeared to agreed with that of the in-sole treadmill testing. As an alternative confirmation, the overground running returned a stance consistently around the 25% value while the treadmill running returned a stance of 33% at all speeds. As an alternative confirmation, the positive half cycle of vertical acceleration for the overground running takes around 70 degrees or
19% of the gait cycle, compared to 90 degrees or 25% of the gait cycle for treadmill running.

**Q.4.** Does the CoM acceleration indicate a reducing V-displacement and increasing AP-displacement as running speed increases? Further, is there any indicator in the CoM acceleration to suggest a relationship to leg-stiffness?

V-displacement clearly decreases as treadmill speed increases. During treadmill running there is no conclusive indication that AP-displacement increases with an increase in speed. Of the sample data, several subjects maintained a static AP-displacement, some had decreasing AP-displacement and some had an increasing AP-displacement. Further testing using multi-speed overground running would be required.

There appeared to be no method of identifying leg stiffness from the acceleration data. A number of possibilities exist, such as the rise-time of the V-acceleration or the variation in V-acceleration rise time between gait-cycle normalised curves from different speeds. Other potential candidates included the AP-breaking after initial contact, the shape of the V-acceleration peak or even the change in V-displacement between initial contact and minimum V-displacement. Isolation of leg-stiffness would require further testing specifically for leg-stiffness.

**Q.2.** Can the CoM acceleration signal be normalised such that a group of like-subjects will produce similar graphical results, and therefore allow subjects to be measured within or outside a standardised group?

Q2
Several attempts were made to normalise the multi-speed acceleration data (and its derivatives) using rudimentary 1st level factors such as leg-length, mass and stride-rate. More complex analysis could possibly uncover additional factors required for normalisation. Individual running style factors, such as the running speed with maximum V-displacement or the speed of maximum V-velocity or…, the numerous other 'individual' traits, all combine to confound simple analysis. To analyse this...
further, many more trials would be required using more homogeneous subject groupings eg. groups of trained male runners with the same leg-length and mass.

Despite this, there are several measures, such as stride consistency or speed dependent lean that may be biomechanically useful and that would be able to be normalised for comparison across a group.

Q.3. Further, regardless of the ability to normalise the CoM acceleration, can this acceleration, and its biomechanical affect, be categorised such that the CoM acceleration signal can be used to partially define a subject's running style?

Q3
The acceleration signal and its derivatives, including the gait-cycle normalised multi-speed representation, did appear to contain directly interpretable information on the subject's running style and therefore allow some categorisation of the subject's running style. Some basic biomechanical or style factors include:

- Step-rate vs. Speed relationship (Chapter 5).
- Lean relationship to running speed.
- Consistency of action, eg did the subject maintain a consistent step-rate, 3D displacement, and orientation etc. through successive strides?
- Reproducibility, eg was the subject's running style consistent at different speeds (see [27])?
- Intra-athlete relationship between V, ML and AP acceleration shape and magnitude (interpreting Gait-cycle normalised multi-speed data):
  - Timing of peak pelvic tilt relative to peak V-acceleration (Assuming ML-acceleration was primary pelvic-tilt as contended).
  - Trajectory resulting from V & AP acceleration combination.
  - Magnitude of AP braking deceleration on initial contact?
  - Measure of left/right asymmetry.
- Inter-athlete relationships:
  - Comparison of signal magnitudes across subjects.
  - Comparison of axis phase relationship (order of axis peak acceleration)
  - Comparisons of asymmetry
  - Etc.
A number of parameters used in journal articles appear extractable. Some require the presence of the in-sole sensors although for some subjects the contact phase could be estimated. The following parameters used and defined in [25] are extractable; in-stride vertical velocity at toe-off, velocity angle at toe-off (when combined with known linear velocity), height of take-off, stance time, stance distance, step-rate, vertical velocity of touchdown and takeoff.

There are numerous measures that could be used to define running style; further study would be required to identify key parameters.

Q1. *Is there useful biomechanical information that can be extracted from Centre-of-Mass (CoM) triaxial accelerometer data collected from elite athletes engaged in running?*

Q1
From this brief review of the CoM accelerometry, in conjunction with the topic coverage in previous chapters, there appeared to be considerable useful extractable information of interest, particularly in the field of sports biomechanics. How useful and how likely this that form of analysis would enter mainstream sports biomechanics is unknown.

Potential exists for this form of analysis to be developed further, particularly with the development of an appropriate accelerometer-biomechanical model incorporating the two ambulatory models (inverted pendulum and bouncing gait) along with the appropriate anthropometric measures, similar to the models developed in [41] for swimming and in [42] & [17] for rowing. An operating model of this form would greatly enhance the usefulness of collected accelerometer data.
6.6 Conclusion

The primary interest of sports scientists studying the biomechanics of running is to understand the human running mechanism so that athletes can be trained to safely run faster. To this end, complex studies (often involving expensive and bulky cinematic equipment located at athletic tracks) monitor dozens of variables over numerous body parts. These studies take considerable time to set-up and then only collect small samples of athletic activity that is painstakingly analysed to possibly determine a few key parameters.

As an alternative to this, monitoring devices that have low cost & low intrusiveness can be attached to dozens of athletes actually engaged to competition. These devices can monitor hours of activity and can be used to directly compare aspects of the biomechanics of different athletes, with a knowledge of which athlete was the fastest under competition conditions. Within minutes of completion of a race, a coach or athlete could obtain feedback that would assist in preparing for the next race.

The above assumes that the data collected from the low cost monitoring devices, in this case the triaxial accelerometers, can be analysed such that useful information is extracted. From the analysis of this chapter, it appeared that accelerometer data could be used to generate directly useful biomechanical information as well as biomechanical information that could be used in conjunction with other monitoring systems. Even given the limited test data available, the scope of extractable information appeared broad. Several 'rules of thumb', based on simple kinematics, enhance the interpretation of the available data.

Further study is necessary to fully develop the available biomechanical information contained in CoM accelerometer data. The development of an acceleration-based model of running biomechanics would be beneficial.
6.7 References


[18] Need internal ref on national rowing project.


7 Compression of Athlete Data

As a preface to the introduction, the question must be asked, *what is so special about the compression of athlete data?* After all, in the internet-multimedia age, data compression is ubiquitous. Even primary school students download compressed music to their personal music players, watch compressed movies on their personal portable DVD players, 'zip' data files for emailing, while conversing with their friends via compressed real time speech on their mobile phone.

What is special about compression of athlete data is the context. There are many tools available in the compression toolbox but it is necessary to identify those tools that best fit this specific problem. Athlete data *looks* similar to audio data but compression tools designed for *aural* perception are not appropriate. Athlete data may appear similar to heartbeat data but compression of heartbeat data does not normally deal with a widely varying offset in conjunction with periods of high intensity signal interspaced with periods of zero activity. Heartbeat data has specific activity signatures of significance to a cardiologist while athlete data has activity signatures significant to a sports scientist. Earlier chapters in this Thesis identify some of the specific activity signatures; the estimated energy expenditure is one case, athlete orientation is another. Extraction of this data is a form of perception-based compression similar to perception-based audio or visual compression techniques.

While the compression of athlete data does not appear to be specifically covered in any technical literature, this is probably because it is currently a question of application of existing techniques, rather than the identification of a new complex problem. As the study of athlete data identifies more significant activity signatures, the problem of athlete data compression will become more clearly defined.

7.1 Introduction

This chapter deals with the analysis of accelerometer based athlete data for the purpose of compressing the data for transmission and storage. The analysis and proposed compression techniques have been considered within a framework consisting of four main objectives:
1. Minimise the bandwidth necessary to transmit or store data,
2. Minimise the power necessary to perform the compression,
3. Maximise the retained or transferred information,
4. Optimise compression for functioning over a low bandwidth noisy wireless link.

The necessity for compression exists both for systems that log data and systems that transfer data in real time. For wireless transfer of data, the volume being transferred must be minimised to reduce both the required bandwidth and the wireless power costs. Logged data must be compressed to minimise the memory requirements and to reduce download times. Without compression, a system monitoring insole sensors on each foot as well as a centre-of-mass triaxial accelerometer, could conceivably take longer to download the data than it took to collect the data.

This chapter covers the following topics:

- Considers the question of information content within athlete data,
- Reviews the background of data compression,
- Proposes a particular implementation of compression,
- Analyses raw accelerometer data for the purpose of compression,
- Identifies interactions between compression and transmission/storage channel,
- Summarizes the chapter.

### 7.2 Athlete Data Information Content

The athlete monitor is designed for flexibility with the ability to interface to a range of sensors. In the context of team-sport, the primary source of athlete data is the triaxial accelerometers. The raw data obtained from the accelerometers contains considerable information but not all the information is important in all circumstances. Because of the limited bandwidth of the data channel, it is necessary to minimise the extraneous data by extracting the key information and encoding it appropriately. The definition of "key information" is sport and circumstance specific.
For many activities the key information is in the form of summary data such as a swimming system that identifies the stroke, counts laps and calculates lap times. A running monitor may calculate step rates, count steps and estimate distance. A football monitor may include the functionality of the running monitor as well as detection of contacts, impacts and orientation changes. The actual information content varies from sport to sport and is dependent on the end use of the information.

Chapter 4 investigated the use of accelerometers and reviewed some of the signal processing techniques available for extracting information content. In particular, this information included:

- Orientation of the athlete,
- Repetition rate of activity,
- Intensity or magnitude of acceleration
- Underlying shape of the acceleration signal.

Each of these conveyed some information regarding the activity of the athlete.

Chapter 5 investigated the extraction of an Energy Expenditure (EE) estimator. This estimator was in the form of combined acceleration intensity / activity repetition rate output. The extracted EE estimator is key physiological information.

Chapter 6 investigated the information conveyed by changes in the underlying shape of the athlete signal during repetitious activity. The shape of the athlete acceleration signal combined with the way the signal changes, conveys information regarding the biomechanical activity of the athlete.

For much athlete data, defining the information content is complex and may it require complex analysis and processing to determine what component of the signal is carrying the required information. In some situations, data must be collected from multiple synchronised sensors and the analysis and data extraction performed after the event. In this case the raw data must be stored and transferred. As indicated previously, it is necessary to compress this raw data for storage and transmission.

One measure of compression is the compression ratio, expressed as a percent.
Using the EE estimator as an example, if the source data was generated with triaxial accelerometers using 10-bit sampling at 150Hz, and the output is represented as a single 8-bit number per second, the resultant compression is 99.8% as given below.

\[
\text{Compression Ratio} = 100 \times \frac{\text{number of input bits} - \text{number of output bits}}{\text{number of input bits}} = 99.82\% 
\]

As shown in Chapter 5, the EE estimator could be extracted from triaxial 8-bit data sampled at 25Hz. This reduces the input data rate but the resultant compression ratio is still considerable (98.7%).

Appropriate information extraction can result in significant compression.

As various forms of information extraction have been covered in previous chapters, and in other sports related projects (as referenced in the introductory chapter and other chapters), the rest of this chapter investigates compression techniques in general and the compression of the raw athlete signal in particular. Although the signal source for this analysis is accelerometers, the compression techniques are applicable to data from other sensor types since the underlying signal source is an athlete subject to the limitations of physics and a human body. To maximise compression, an understanding of the athlete's kinematic parameters and their affect on the sensing system is required.

### 7.2.1.1 Athlete Data: Parameters for Compression

The following two subsections list the parameters related to some aspects of athlete activity and the subsequent sensor system outputs.

**The Athlete**
Across humanity, athletes represent the subset that operates at the fastest physical level. Some approximate\textsuperscript{46} biomechanical limits are listed:

- $6 \text{ms}^{-2}$ maximum horizontal linear body acceleration.
- $13 \text{ms}^{-1}$ maximum horizontal linear body velocity.
- $5-6\text{Hz}$ step frequency (fundamental)
- data up to $3^{\text{rd}}$ harmonic
- $4\text{g}$ peak to peak vertical in-stride acceleration
- $2\text{g}$ peak to peak anterior-posterior in-stride acceleration
- $3\text{g}$ ground reaction forces
- $\pm 1\text{g}$ offset.
- impact acceleration observed to exceed $7\text{g}$.

Other observed values, given as 10 bit Analog-to-Digital Converter (ADC) units, from trial data include:

- maximum value during running (10x10 treadmill trial) +161 units
- minimum value during running (10x10 treadmill trial) -75 units
- maximum difference for any athlete (10x10 treadmill trial) 256 units
- maximum values walking at 3km/h 14-23 units
- minimum values walking at 3km/h -15 to -26 units
- difference between min and max for all athletes 30-49 units.

The above values are after the orientation (gravity offset) has been removed

- Football players can spend $>50\%$ of monitored time at the lowest activity levels.

Other limits apply to various subcomponents in the system. The physiological limits are within the same order of magnitude with a heart-rate maximum of around 4Hz. Although a 6Hz system would suggest a sampling rate of around 12Hz some athletic data of interest is in the timings of events within each biomechanical cycle. Sensor sampling-rates need to be sufficient to capture this information. An example of data that should be conserved is identified in Fig. 7.15. The short spikes indicated in the

\textsuperscript{46} These values are derived from various sources, but are predominately derived from test data supplied by the AIS. Some of these values, such as maximum horizontal acceleration and velocity are derived
figure could be some form of biomechanical noise but gait-cycle normalisation (Chapters 4 & 6) identifies these spikes as components of the athlete's gait.

**The Data Collection System [1].**

The following limits were set by the electronics and the initial requirements.

- The system uses 10-bit sampling giving a maximum symbol set size of 1024.
- The system samples at 150Hz on multiple channels (usually 3)
- One gravity has a value of approximately 60 units, ±7g equates to approximately 840 possible symbols (smaller than the possible set size).
- 32M Byte of memory can store approximately 8.9 million 3-channel x 10-bit packed samples or the equivalent of 16.5 hours of raw data.
- The wireless channel capacity is unknown as channel capacity is a function of the technology and the number of devices required to communicate at once. (Refer Chapter-2)
- Infrared download speed was limited to 38400bps (maximum standard asynchronous communications data rate for the serial port). This could equate to a download-to-collection rate of just over 1:1.25, one hour of data collection takes one hour 15 minutes to download (7 sensors @ 10-bit 500Hz) to the more common 6:1 or six hours of data takes one hour to download (3 sensors @ 10-bit 150Hz). A USB link at 200kbps improves these figures by a factor of approximately 5 (one hour becomes twelve minutes).

**Extractable Information**

Useful extractable information is currently limited to the items identified in this chapter but any data compression should not preclude the extraction of useful information in the future. Obviously, this is an imprecise specification and a combination of qualitative and quantitative approaches is required. Currently identified information:

- step rate for energy expenditure estimates.
- accelerometer counts for energy expenditure estimates.

from 10metre split timing for sub 10 second 100 metre sprinters, arguably the fastest humans.
• orientation data (3 channel low pass filter data)
• fundamental biomechanics (10Hz to 15Hz response appears adequate although no data is available from the fastest sprinters.)
• impact signal and impact direction

Athlete Data Information Content Summary
Compression of raw data is predominately required to minimise on-board storage requirements and to minimise transfer rates to the host system. Athlete data is typical of diffuse data like sampled voice signals and unlike the forms of data that are typically compressed on computer systems (such as text documents, financial information etc.).

The appropriate technique for compression of athlete data is dependent on the form of the data, the transmission channel and the final use of the data. Because of this, it is necessary to examine the athlete data, determine the governing parameters of the data and match these parameters to the channel and the extractable information. A further limitation for low-power mobile platforms is the available processing power. These athlete data parameters must then be matched with the appropriate compression technique. Athlete data analysis, with reference to proposed compression techniques, is described in subsequent sections.

7.3 Data Compression Background
This section briefly covers some of the basic background relating to data compression and in particular, those areas of interest to the compression of athlete data on a system with restricted processing power.

7.3.1 Entropy Coding.
Compression techniques are concerned with the existence of symbols (such as characters and words), the number and type of symbols, their sequencing, combinations, permutations and probabilities. Shannon (1948)[2], in regard to communications, defined the term Entropy as a measure of the information in a signal. The measure of entropy in turn gave the minimum number of binary digits (bits) that could be used to represent a message symbol. Around 1948-1949, Shannon and
Fano[3] independently developed a **Minimum Redundancy Coding** system, consequently referred to as Shannon-Fano coding. In 1952, Huffman [4] published a form of minimum redundancy coding that improved on the Shannon-Fano coding. Since that time, Huffman coding predominates where minimum redundancy coding is relevant. The Shannon-Fano and Huffman coding uses variable length codes where the length of the coding symbol (in bits) is inversely related to the probability of the existence of the message symbol being encoded. Huffman described a minimal-redundancy code as one "... constructed in such a way that the average number of coding digits per message is minimised."[4]. Both the Shannon-Fano and Huffman coding techniques code an input symbol into an integer number of bits. The optimal number of bits, based on the distribution of message symbols, may require a fractional number of bits. As fractional bits do not exist, the optimal compression cannot always be achieved.

Another form of entropy coding is Arithmetic Coding. This form of coding takes a sequence of input symbols and represents the sequence with an integer number of bits. With Huffman coding there is a one-to-one mapping of input message symbol to output code and consequently suboptimal codes can be produced as each message symbol is coded. For example, a symbol may be optimally represented by 2.5 bits but must be coded as 3 bits. With Arithmetic coding, because groups of input symbols are coded into a single output code, the sub-optimal 'fractional bit' is shared across many input symbols. For example, a group of 5 symbols may be optimally represented by 17.5 bits but must be coded as 18 bits. Overall this allows the Arithmetic coding to approach the optimal more closely than either Shannon-Fano or Huffman coding. Numerous aspects of Arithmetic coding are covered by a number of United States patents held by IBM47.

### 7.3.2 Static & Adaptive Systems

For the systems described in Section 7.3.1, the entire message is read and the probability of the presence of any particular symbol calculated, resulting in a

---

47 The Compression FAQ [5] lists at least 19 IBM patents relating to Arithmetic coding. It also identifies other patent holders with Arithmetic coding related patents.
probability density function (PDF). The coding scheme is generated for each symbol based on the PDF and finally the input message is coded (or compressed). To allow correct decoding, the coding scheme must either be already known to the decoder or be stored with the compressed data. This system is a static system. If the distribution of symbols is not uniform through the whole length of the message, sub-optimal coding could occur, even to the extent of expanding, rather than compressing, parts of the message. This is a two-pass system, as the data must be read once to generate the PDF, then read a second time to perform the coding.

To overcome the limitations of static coding, adaptive coding variations have been developed. Adaptive Huffman coding was first proposed by Faller in 1973 [6]. Adaptive coding is applicable to streams of data, where the PDF is unknown, or to files where the PDF for one section is considerably different to the PDF for another section. Adaptive coding assumes no foreknowledge of the PDF but builds up and modifies the coding scheme on-the-fly, recording changes to the coding scheme in the output stream. In this way the decoding system can read the incoming data and extract both the coding scheme and the data. This is referred to as a single pass system as the input data is only read once.

### 7.3.3 System order

The Huffman coding described previously referred to data streams where one symbol from the input message was converted to one output code. This is a basic zero-order system. Higher order systems exist where, for example, the PDF describes the probability density of pairs of symbols (first-order system). This is only relevant if pairing, or higher order grouping, is common, in which case treating the pair of symbols as a single different symbol removes some data redundancy from the system.

### 7.3.4 Dictionary Schemes

The previous systems are based around the probability of some symbol occurring in a data stream and therefore the system can be described to some extent by the PDF. Other systems, known as dictionary schemes, do not rely on the PDF but instead exploit the construction of language. Documents and financial transactions are usually
encoded using combinations of loss-less compression techniques such as dictionary based encoding systems based on a Lempel-Ziv algorithm LZ77 [7] or LZ78 [8] or the later LZW (Lempel-Ziv-Welch)[9] patented algorithm. The output from dictionary-based compression may subsequently be compressed with another form of encoding such as Huffman coding. The dictionary-based techniques are based in language concepts where individual symbols (characters) form up in sequences that get repeated (words or roots of words). These common sequences then become a symbol that replaces the original collection of symbols.

7.3.5 Lossy vs. Lossless Compression

Athlete data is a form of diffuse data in some ways similar to speech. Diffuse data is typically encoded using a combination of lossy and lossless techniques. Lossy techniques do not attempt to exactly reproduce the source data but make approximations that are considered adequate. These approximations are based on an understanding of the process that generates the data and the process that interprets the data. The GSM [10] mobile telephone network utilises a compression technique that is based on an understanding of both the human speech and hearing mechanisms [11]. JPEG, a standard for compression of still photography [12][13], applies transforms and compressions based on the human eye's differing ability to perceive intensity and colours. MPEG [14], a compression technique related to compression of moving pictures, exploits the ability of the human eye to integrate a sequence of complete and partial images and to interpret this as a continuous sequence.

While compression for storage can treat the resultant compressed data as a continuous file, compression for transmission over a noisy medium must take into consideration the characteristics of the transmission channel. This includes the ability to recover from errors in the received data. Compression standards such as JPEG incorporate restart markers that enable decompression to continue in the event of corruption of a section of the compressed data.
7.3.6 Transformation

Transformations are used in compression to move data from one descriptive system to another system that is more amenable to compression. A Cartesian point, or series of points, described by a set of x,y,z co-ordinates could possibly be transformed into polar co-ordinates describing angles and magnitudes (or visa-versa), if the alternate system simplified the compression. JPEG uses the Discrete Cosine Transformation [15] as a step in the image compression.

7.3.7 Patents

United States Patents and Trademarks Office (USPTO) grants patents on software algorithms. This is not possible in many countries but the granting of a software patent in the USA influences the availability and take up of patented techniques worldwide. Unisys (and others) had a patent on the LZW compression technique used in the Graphics Interchange Format (GIF\textsuperscript{48} [16]), UNIX 'compress' and other common programs. When Unisys began enforcing the patent, alternative algorithms were implemented. Katz, the developer of PKZIP\textsuperscript{49} [17], developed a combination LZ77-Huffman compression function known as 'Deflate' (Described in RFC-1951 [18]). To move away from the use of the patent encumbered GIF, the Portable Network Graphics (PNG [19]) image compression was developed. As noted previously, IBM holds many patents associated with Arithmetic coding. Gailly [5] identifies many dozens of patents associated with data compression.

7.3.8 Other Systems

There are many techniques and sub-techniques related to compression. The appropriate compression technique must be determined by the type of data to be compressed. Some forms of compression are designed for a specific purpose and are inappropriate

\textsuperscript{48} No obvious definitive reference appeared to exist for GIF, although the reference given here appears to be the specification written by CompuServ (since purchased by America On Line (AOL) and is hosted by W3C World Wide Web Consortium (www.w3.org)

\textsuperscript{49} Phil Katz implemented a program called PKARC as an improvement on a commonly used 'ARC' compression tool. To avoid legal pressure from the owners of the ‘ARC’ tools, around 1988-89 Katz developed the PKZIP tools, available as shareware, which became the de facto data compression tool.
when used in other circumstances. JPEG was designed for photographic images where many shades of colour blend together. GIF was developed using the previously patented LZW technique and limits the number of colours per frame to 256. GIF is preferred for simple computer generated images while JPEG is preferred for photographic images.

Run Length Encoding (RLE\(^{50}\) [20]) counts continuous runs of a character and then codes the entire run with just one instance of the character along with the character count. RLE was initially used to compress simple images. RLE can be found incorporated in a variety of other compression systems such as Modified Hamming [21], used in fax data transfer, and in JPEG, as well as other systems.

Fingerprint images are routinely stored on computer but the image compression techniques of JPEG, GIF etc are inadequate and fingerprints require their own compression technique (see examples [22][23]). Some forms of compression only exist to augment other forms of compression.

Silence Compression\(^{51}\) [24] is a lossy technique for identifying and removing noise signals. It is not limited to audio but can be applied to many types of files. In essence, silence compression operates as a filter, removing single isolated instances of symbols that do not conform to the surrounding signal or where the intensity of the signal falls below some nominated threshold. A two-dimensional silence compression scheme could be implemented to remove 'fly-specs' from a digital image by identifying isolated pixels that do not conform to those surrounding them. This would benefit coding schemes such as RLE that would otherwise have to encode for a single isolated pixel and then start a new run count. Some other forms of compression also benefit from silence compression.

\(^{50}\) Many references exist to RLE, many suggest CompuServ as the originator. Reference [20] lists some of the common file formats that implement RLE. (ARC also implemented RLE)

\(^{51}\) This reference is not definitive however it is used as a model for the system described later.
7.3.9 Summary

There are many compression schemes, variations on compression schemes, and combinations of compression schemes, each with various technical, commercial or user-base reasons for existing. The loss-less compression techniques divide into those based on probability, like the minimum-redundancy coding, entropy-based methods, and those based on the dictionary model. Lossy techniques are based on 'perception' where the data is graded from most perceptible to least perceptible. The level of compression is determined by the amount of acceptable degradation. To aid the compression, data may be transformed into different formats. Some compression schemes are based on a static description of the data, others adapt the coding to the dynamic content and the coding information is inserted into the data stream along with the compressed data.

7.4 Proposed Athlete Compression

In order to analyse the compressibility of the athlete data, a compression scheme that appeared to match the requirements identified in Section 7.1 was proposed. The compression scheme, depicted in Fig. 7.1, used a combination of lossy and loss-less techniques. These techniques are described below, along with discussion on how they were implemented and how the athlete data was analysed to gain an understanding of the impact of the particular compression scheme.

![Fig. 7.1 Proposed athlete-data compression system](image)

The system of Fig. 7.1 combines a sequence of compatible functions used to build a compression system meeting the four requirements identified in the introduction. The compression system begins with data sampled with the appropriate number of bits and at the appropriate data rate. Single isolated noise spikes are removed by silence compression prior to the transformation of the data by differencing. The differencing greatly reduces the number of symbols present in the source data but the lossy
companding step forces a further reduction to a small handful of symbols. The final step is to encode the data using minimum redundancy encoding, or during periods of low activity, using RLE.

For wireless operation, data packet sizes are determined by the networking protocol. The compression ratio of a data stream is not static, but varies inversely to the information content of the data. Because of this, the size of the compressed data packets are constantly changing and an algorithm is required to ensure that compressed data packets can be correctly matched to wireless packets. A further complication occurs because data processed using differencing cannot be reassembled correctly if intervening data is lost. All wireless packets must incorporate absolute reference points to restart the correct reassembly of compressed data. Another consideration is the effect of data corruption on variable length codes. Any corrupted bit in a stream seriously inhibits the ability of the decoding system to extract any useful data after the corruption. While restart points in a data stream assist in the reconstruction of data in the event of lost packets, if adaptive compression is used the loss of a data packet may mean that no subsequent data can be decoded if the lost packet included adaptive coding information. Each of these points requires consideration in any system transferring compressed data over a wireless link.

Compression of data for logging has fewer limitations. On a small-embedded system the primary limitation is the RAM memory available for packet assembly prior to writing to the non-volatile FLASH memory. Similar to wireless packet assembly, an algorithm is required to track the total size of compressed data from all channels, so that blocks of compressed data are matched as efficiently as possible to FLASH block sizes. As there is no possibility of lost packets, adaptive schemes are permissible.

Descriptions of each of the subsystems identified in Fig. 7.1 follow.

7.4.2 Sampling System

The volume of raw data bits is a function of the sampling system, in particular the sample-rate and the bits-per-sample. Although an immediate 50% compression can be applied by halving the sample-rate, the athlete data requires qualitative and
quantitative analysis to determine the impact of any changes to the sampling system. Modifying the source of the data will have flow-on effects on the downstream compression algorithms due to changes in the redundancy in the data. Table 7.1 gives examples of sampling system output data rates based on various sampling system parameters.

<table>
<thead>
<tr>
<th>Sample Rate</th>
<th>Bits per Sample</th>
<th>Channels</th>
<th>Output Bit Rate (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>10</td>
<td>8</td>
<td>40000</td>
</tr>
<tr>
<td>150</td>
<td>10</td>
<td>3</td>
<td>4500</td>
</tr>
<tr>
<td>150</td>
<td>8</td>
<td>3</td>
<td>3600</td>
</tr>
<tr>
<td>75</td>
<td>8</td>
<td>3</td>
<td>1800</td>
</tr>
<tr>
<td>50</td>
<td>8</td>
<td>3</td>
<td>1200</td>
</tr>
<tr>
<td>25</td>
<td>8</td>
<td>3</td>
<td>600</td>
</tr>
<tr>
<td>25</td>
<td>8</td>
<td>1</td>
<td>200</td>
</tr>
</tbody>
</table>

### 7.4.3 Silence Compression

Silence compression does not of itself improve the compression ratio; it is a precursor to other forms of compression. The two major beneficiaries of silence compression in this scheme are minimum redundancy coding and RLE. Silence compression works by removing short noise spikes that interrupt an otherwise flat signal. The silence compression utilised inspects the time series data and removes single sample point noise spikes. The algorithm used is simple and is depicted in Fig. 7.2. While the benefit for RLE depends on a break-even run length, every noise sample cancelled by silence compression benefits Huffman compression (Fig. 7.3). The value of the benefit depends on the actual encoding scheme.

The low-power silence compression program implementation appears in Appendix 9.5. This code appears complex although only a portion is executed on each sampling. To reduce processing load the code remembers the state of previous comparisons using flags. This reduces the requirement to manipulate array pointers and reduces the comparisons required at each step. If a silence is detected then some comparisons
normally required for subsequent samples are skipped. In the diagram of Fig. 7.2, the current sample under test is identified by the 'A'; the next two samples have already been proved to match therefore the appropriate flags can be preset.

3 samples
  X X X 2 samples
  A X X

Oldest Sample Noise Newest Sample

Fig. 7.2 Silence Compression Algorithm, if three preceding samples and two trailing samples are identical then the sample in question is set to match.

<table>
<thead>
<tr>
<th>Decimal Difference</th>
<th>Code Bits</th>
<th>Example Huffman Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>101</td>
<td>X X X Y X X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0 0 +1 -1 0</td>
</tr>
<tr>
<td>+1</td>
<td>100</td>
<td>0 0 100 101 0</td>
</tr>
</tbody>
</table>

Fig. 7.3 Comparison of effect of silence compression on output of differencing and Huffman encoding. For the Huffman code given, the silence-compressed data results in a Huffman encoded data stream 4 bits shorter than the results from the raw data.

An analysis of the athlete data was performed to determine the usefulness of silence compression within the scheme of Fig. 7.1.

7.4.4 Differencing

Differencing is a transformation that greatly reduces the number of symbols present in the input data and is performed by generating the difference between a sample and the previous sample as indicated in Eqn. 7.2.

Equation (7.2) \[ d_n = x_n - x_{n-1} \]

Where \( d_n \) is the current difference, \( x_n \) is the current sample and \( x_{n-1} \) is the previous sample. The first line of 'C' programming code in Fig. 7.5 implements this equation. Triaxial accelerometers detect gravity and the orientation of the sensor determines if gravity causes a positive or negative offset. A sensor detecting ±2g of activity on top of a ±1g offset, has a ±3g dynamic range. If 1g is represented by 50 discrete levels, a
6g dynamic range requires 300 discrete levels. This is too many levels to manage within a minimum-redundancy coding scheme, particularly where different channels are centred on different offsets. By using differencing, all channels centre on zero difference, therefore enabling a common coding scheme across all channels. Differencing also reduces the overall dynamic range of inputs. An absolute value may move up and down through 300 discrete levels where the size of the change (the difference) may only use 3 or 4 different values. This reduction in the number of symbols used to describe the data is clearly exhibited in Fig. 7.4 where 68 discrete sample values were present in sample data but differencing of the data reduced this to 18 discrete differences. Differencing is a loss-less transformation.

Fig. 7.4 (a) PDF of 8-bit athlete data (b) PDF of Differences. Data from vertical channel of 3-tackle test (Refer Chapter 4). The test activity includes changes in orientation, running, jumping and tackling.

```
diff=silence[s_head][channel] - silence[oldIndex][channel];
totaldiff[channel] = totaldiff[channel] + diff;
totaldiff[channel] = totaldiff[channel] - getHuffCode(totaldiff[channel]);
```

Fig. 7.5 Code Fragment for calculating differences and handling ‘carry-over’ differences from companding.

### 7.4.5 Companding

Companding is traditionally a weighting technique that represents small signals with proportionally more bits than large signals. The purpose is to make the quantising error relative to the signal size. This can be implemented using a non-linear analogue amplifier or an analogue to digital converter using tapered quantisation [25]. In both cases, the signal being quantised must oscillate about the zero axis.
The athlete monitoring system uses single sided accelerometers sampled by single sided analogue-to-digital converters. The accelerometers detect gravity, which can affect the size of the signal offset on one or all channels. This precludes the use of special companding hardware since the location of the mean of the input signal is constantly changing.

Since companding cannot be applied to the raw acceleration signal, any companding must be applied to a differenced signal. The purpose of companding in this instance is to minimise the number of symbols required to represent the athlete signal. Reducing the number of required symbols reduces the bits needed to represent them either directly (eg 64 symbols requires 6 bits, 32 symbols requires 5 bits), or indirectly (minimum redundancy compression schemes can lead to localised data expansion if a group or rarely used symbols coincide).

![Fig. 7.6](image)

The companding algorithm is implemented after differencing by truncating the current difference to a companding level and then *carrying over the remainder* to the next sample. It is important than no difference value is lost since the successful reconstruction of the signal requires the cumulative sum of all differences. This form

---

52 Single sided, in this context refers to the fact that sensor device and the analogue to digital converter are both using only one supply voltage and the signal is in oscillation about an offset that is between zero volts and the supply voltage. The signal will never go below zero.
of companding affects the signal bandwidth by delaying some higher speed transitions. To overcome this filtering effect, a form of predictive\textsuperscript{53} differencing can be implemented. The implementation of the carry-over appears in the programming code fragment of Fig. 7.5. In this code the current difference is added to any previous carry-over with the resulting difference sent for companding (the function call 'getHuffCode'). The compander returns the companded value, which is then used to calculate any new carry-over. An example of the companding algorithm appears in Appendix 9.5. A fragment of the algorithm appears in Fig. 7.7.

\begin{verbatim}
} else if (adpcm<16){
    idx_huffcodes=idx_huffcodes+(sdpcm*4);
    results=8;
}
\end{verbatim}

Fig. 7.7 Companding algorithm code implementation fragment. Values from 8 to 15 are returned as 8.

For non-zero differences, determination of the compander output was controlled by a sequence of 'C' if-else statements. The compander implementation was symmetrical with the compander determining the output based only on the magnitude of the difference ($adpcm$) and returning the compander output as both an absolute value ($results$) and an index pointer ($idx_huffcodes$) to the array of minimum redundancy codes. The 'results' value was multiplied by the sign of the difference ($sdpcm$) before being returned to the calling function. The index pointer, initialised to the mid-point of the array, was modified by either a negative or positive offset to find the correct code.

Predictive differencing was performed by a slight modification to the compander algorithm. The compander code in Appendix 9.5 and the code fragment of Fig. 7.7 describe a compander where the magnitude of the difference was always rounded down toward zero-difference. With predictive differencing, the compander may round the difference magnitude up or down as in the code extract of Fig. 7.8.

\textsuperscript{53} This method rounds large differences to the nearest companding level, rather than rounding down and assumes that if large differences occur, it is likely that they will be followed by more differences in the same direction therefore it is 'predictive'. This is different to the Predictive Differential Pulse Code Modulation of [26] which refers to the prediction of the colour of pixels neighbouring a pixel being sampled ('nearest neighbour prediction').
 elseif absValue < 6       %values 4,5, round down to 4
    value = 4;
 elseif absValue < 9       %values 6,7,8 round to 7
    value = 7;
 elseif absValue < 12      %values 9,10,11, round to 10
    value = 10;
 elseif absValue < 16      %values 12,13,14,15 round to 14
    value = 14;
 else
    value = 18;  %everything 16 & greater round to 18

Fig. 7.8 Matlab code for predictive differencing implemented in the compander.

Optimal compander operation requires analysis of the athlete data to determine an appropriate number of differencing levels, to identify the filtering impact of companding and the effectiveness of different companding techniques. Companding cannot be designed in isolation; it is heavily dependent on the sampling system.

### 7.4.6 Minimum Redundancy Coding

The encoding proposed for this application was zero-order, static Huffman encoding. This implementation used analysis of historical symbol occurrence in a representative data set to generate the PDF. From the PDF, a Huffman encoding table was generated with the most frequently occurring symbols being represented by the least number of bits. This means the data model was known to both ends of the circuit prior to the transmission of any data. If the data content changed significantly at different points of the data stream, static Huffman codes would not effectively match the stream and data expansion could occur. To prevent data expansion, the number of symbols present in Huffman code table was limited so than no input symbol required more than 8 bits to represent it.

Table 7.2 includes a small sample of a set of Huffman Codes from the program code example in Appendix 9.5. There were 18 codes in this set with a maximum code length of 8-bits. For 8-bit input data, while some 8-bit input data could be represented with a
1-bit output, no 8-bit input symbol would require more than 8-bits in the output. This prevents any possibility of data expansion. This code table was not the only possible code table for this data. Although the codes were variable length, they were all stored as 8-bit unsigned chars in an array, arranged symmetrically about the array mid point. This allowed the compander code of Appendix 9.5 to generate an index into the array. The formatting of the codes was designed to allow the program to quickly determine the code length and the code. This was done by making all the codes in the table, except the code for zero difference, start with a zero. The codes are stored to the right of the byte, with the most significant unused bits filled with ones.

Table 7.2 Example Implementation of Huffman Code Table

<table>
<thead>
<tr>
<th>Code (Hex)</th>
<th>Code (Binary)</th>
<th>Variable Length Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xA2</td>
<td>10100010</td>
<td>0100010</td>
</tr>
<tr>
<td>0xEB</td>
<td>11101011</td>
<td>01011</td>
</tr>
<tr>
<td>0xFC</td>
<td>11111100</td>
<td>00</td>
</tr>
<tr>
<td>0x01</td>
<td>00000001</td>
<td>1</td>
</tr>
<tr>
<td>0xFB</td>
<td>11110111</td>
<td>011</td>
</tr>
<tr>
<td>0xE9</td>
<td>11101001</td>
<td>01001</td>
</tr>
<tr>
<td>0xA0</td>
<td>10100000</td>
<td>0100000</td>
</tr>
</tbody>
</table>

The compander function pointed to the appropriate code in the code array and the code fragment of Fig. 7.9 was used to identify the number of bits in the code. This was performed by looking at the most significant bit and testing for a '1'. If '1' was found, this bit was a filler bit and the next bit to the right was tested. This loop repeated until a zero was found. As the loop iterated a bit counter was decremented. The bit counter (huffcount) and the code (huffcode) were used by the Huffman encoder, encodeOutput (Appendix 9.5), to add the appropriate code to the output buffer.

```c
huffmask=128;
huffcount=8;
/* this code finds the first bit of the huffman code eg the first '0' bit */
while (huffmask&huffcode){
  huffcount--;
  huffmask=huffmask>>1;
}
```

Fig. 7.9 Code used to count the length of a variable length code and to identify the variable length code stored within a byte.

Selection of appropriate static Huffman codes was determined by analysis of the distribution of companded differences in the input stream. The Huffman codes include an escape code, to indicate to the receiver that what followed was an instruction to the receiver - not coded data.
7.4.7 Run Length Encoding

RLE may be applicable where the signal does not change for some period or where the signal remains below some identified threshold. In this instance, RLE comes after differencing/companding therefore the most common run is a run of zero differences.

The ability to effectively use RLE requires that the cost of encoding a run of zeros with RLE is less than the cost of encoding a run of zeros with Huffman coding. For some athlete data PDFs, the resultant Huffman codes encode a zero difference in 1-bit. To move from Huffman coding to RLE requires a Huffman escape code, followed by a code indicating 'zero' followed by a count of successive zeroes. Assuming that a Huffman escape code only exists to code for RLE, and that a run length counter consists of fixed number of bits, a breakeven run length can be determined.

Assuming runs of zeros are common, the Huffman escape code may be coded in fewer bits than other less common symbols. Using 5 bits for the escape code and 4 bits for the counter, a minimum of nine successive zero samples would be required to break even. Using RLE for ten or more successive zeros would therefore be beneficial. The existence of the Huffman escape code in a data stream could imply ten successive zeros therefore the counter bits would count zeros in excess of ten. The RLE may also require an escape code. In the above example, it could be assumed that the RLE only codes for runs from 10 to 25 successive zeros. Therefore after the RLE bits are processed the system falls back to Huffman coding. Alternatively, the counter could use an escape code to indicate that another 4-bit run length counter follows. This format allows the RLE to code for 10 to 24 successive zeros at a cost of 10 bits, with each successive 15 zeros costing an additional 4 bits. An example of this approach appears in Fig. 7.11.

With run length encoding of 8-bit data, as in Fig. 7.10, encoding one sample causes data expansion but despite this, even with only short runs present in the input data, there is considerable data compression. Generating a Modified-Huffman stream for bit data, as in the example of Fig. 7.11, the compression benefits are not as great due to the long break-even sequence necessary to start RLE. Using a short escape code may impose a hidden penalty cost since using a short escape code means some other high
probability symbol must be relegated to a longer code (probably an 8-bit code). This may erase any RLE benefit.

| Input: | 5E 5E 5E 5E 5E 5E 5E 5E 5E 5E 5E AF AF 5E AF 5E 5E 5E 5E 5E 5E 5E 5E 5E 5E AF 02 5E AF 01 AF 01 AF 09 |
| Output: | 5E 0B AF 02 5E 01 AF 01 5E 09 |
| Compression Ratio | 58% |

Fig. 7.10 Run Length Encoded Data Bytes. In this figure, the encoded data consists of the sample value (one byte) followed by the run length (one byte). Data represented as Hexadecimal values. Compression ration is the overall compression of the stream.

| Difference Stream: | 10000000000000000-1-10110000000 |
| Huffman Code: | 1000000000000000000000010110100100000000 |
| Modified Huffman: | 100 11111 0110 101101001000000000 |
| Statistics: | 29 samples, 39 bits in Huffman code, 32 bits in RLE output, 18% Compression |

Fig. 7.11 Modified Huffman output comparison to standard Huffman output. Huffman codes using example codes of Fig. 7.3. The difference stream are numerical values, the Huffman and Modified-Huffman codes are bits, spaced to show the code groups. The Modified-Huffman escape used is the sequence 11111 (underlined), with the run-length using four bits for the counter (underlined). Compression is for the RLE encoded output vs. the Huffman encoded stream.

Programming for RLE is constrained by the system limitations of available memory, available processing cycles and particularly by any data packetisation required by the wireless or storage system. To minimise memory requirements, reduce processing latency and spread processing load, it is preferred that data is encoded sample-by-sample. In this case a sequence of zeros would already be encoded in the output buffer before the length of a run was determined.

Assuming zero differences were encoded as zeroed bits, a buffered sequence of ten zeroed bits does not necessarily equate to a sequence of ten zero-difference samples since some of the zeros in the buffer may be the trailing bits of the Huffman code for a previous sample. The same reasoning exists if zero differences were encoded as binary ones. This prevents an RLE system simply counting zero bits in the output stream.

The scenarios for implementing RLE with Huffman coding become complex and so a longer latency between sampling and completion of processing is required to keep complexity of RLE to a minimum. The RLE code must interwork with the other compression components to minimise processing overheads. Analysis of the athlete data for run lengths should be performed to determine the viability of RLE.
7.5 **Accelerometry Data Analysis**

The analysis performed in this section draws heavily on test data discussed in previous chapters. In particular the 10x10 treadmill trial (Chapter 5) provides a source of data that represents activities with a range of intensities and frequencies. The entropy of this data, sampled at two different rates, appears in Fig. 7.12. The presentation of this analysis follows the same order as the description of the compression system:

- Analysis of the effect of changing sample-rate and bits-per-sample.
- Analysis of the impact of silence compression.
- Distribution of Differences. (Probability Density Function)
- Effect of companding on the distribution of differences.
- Matching Minimum Redundancy Coding to the PDF of differences.
- Distribution of Run-Length for Zero Differences.

For some data presented here, to allow direct graphical comparison between various forms of data, where the number of bits is the data is reduced, rather than perform an integer divide-by-two, the data has the least significant bit zeroed. This leaves the magnitude of the signal the same as the original data and the two signals can be directly graphically overlaid. Changes in the shape of the signal due to the reduction in bits can be visually identified.

![Fig. 7.12 Entropy for different sample rates using 8-bit samples, with and without silence compression.](image-url)
7.5.2 Sampling System Analysis

Analysis of the sampling system investigated the effects of both a reduced sample rate and a reduced number of bits per sample. This required a qualitative assessment of the effect of reducing both the temporal resolution and signal amplitude resolution, in isolation and in combination. This qualitative assessment was also based on the assumption that the existing signal was appropriately representative of the activities being monitored. As noted previously, in Chapters 4 & 5, some activities resulted in sensor overload and hardware filters were used to remove this signal. Should the system require the compression of unfiltered data, this analysis, as well as the analysis from subsequent sections would need to be performed again.

7.5.2.1 Trimming Data Bits

The summary of dynamic data from running activities (in section 7.2) resulted in identifying a group maximum dynamic range of 256 ADC units; this value was obtained after offset and gravity removal (using 1Hz high pass filter). Initially this appeared to indicate that a range of ±127 (8 bits) would be an appropriate sample size however gravity is also detected and this affects the offset and the number of necessary bits.

An investigation of the relevance of the two least significant bits indicated that they were of little significance to the final data. In the following graphs, the 10 bit data was compared to a version of itself with the two least significant bits zeroed (to represent 8 bit data). During high intensity activity (Fig. 2.3), visual inspection of both the time domain and frequency domain versions of 10 bit and 8 bit data detected no discernable difference, the two sets of data appeared as a single trace. During low intensity activity (Fig. 7.14) visual inspection identified small variations to the edges of the signal but the fundamental shape remained. The comparison in the frequency domain indicated no discernable difference in vertical and anterior-posterior channels. The mediolateral channel had a reduction in amplitude across all frequencies; this was probably more noticeable since the spectral power density of the mediolateral channel was spread, while that of the vertical and anterior-posterior channels was concentrated in a single frequency. It also suggested that the mediolateral signal had a low signal to noise ratio at this speed.
Fig. 7.13 Time Domain and Frequency Domain comparisons of 10 bit and 8 bit 150Hz data for 21kmh$^{-1}$ running. (a) Comparison of the raw 21kmh$^{-1}$ acceleration data overlaid by same signal with least two significant bits set to zero. (b) FFT of the acceleration signals in (a). From top to bottom, the graphs represent the vertical, mediolateral and anterior-posterior channels. In both the time domain and frequency domain views of the signal, there was no discernable difference between the raw data and the same signal with the 2 least significant bits zeroed.

Fig. 7.14 Time Domain and Frequency Domain comparisons of 10 bit and 8 bit 150Hz data for 3 kmh$^{-1}$ walking. (a) Acceleration traces for 10 bit and 8 bit sampling of walking at 3kmh$^{-1}$, (b) FFT of acceleration data in (a). These graphs depicted the effect of reducing data bits on low intensity activity. From top to bottom, the graphs represent the vertical, mediolateral and anterior-posterior channels. Slight differences in the acceleration signal were perceivable. No difference was observable in the FFT results for the vertical or anterior-posterior channels but the mediolateral recorded a lessening of intensity.

7.5.2.2 Reducing Sample Rate
For maintaining information signal content regarding Per-Second Energy Expenditure, very low sample rates could be used, as it was only necessary to record the input signal frequency. A 25Hz sample rate was compared to a 150Hz sample rate in Fig. 7.15. The 25Hz rate was more than sufficient to capture the athlete step frequency, in this case approximately 3Hz. The 25Hz rate appeared to miss features, which may be relevant to biomechanical analysis. As the test data was only to 21kmh$^{-1}$, 25Hz would appear to be insufficient for recording biomechanical information for sprinters running at 40kmh$^{-1}$. 
Graphical analysis for a sample rate of 75Hz appears in Fig. 7.16, Fig. 7.17 and Fig. 7.18. In these examples the 75Hz sample rate was combined with 7-bit sampling (three least significant bits set to zero). In these examples, the sample rate appeared sufficient but the quantising error introduced by the 7-bit sampling was visible in the high intensity activity (Fig. 7.16) and was completely removing features in the low intensity activity (Fig. 7.17). Comparing tri-axial accelerometer counts (Fig. 7.18) for the 150Hz/10-bit and 75Hz/7-bit sampling, the low intensity activities returned near-identical values but the high intensity activities were noticeably affected. This combination contained sufficient information to regenerate both the low intensity energy estimates (from accelerometer counts) and the high intensity energy estimates (from the step rate).

![Graphical analysis for a sample rate of 75Hz](image)

**Fig. 7.15** Samplerate reduction. Comparison of 21kmh⁻¹ running data sampled at 150Hz and at 25Hz. The dark line represents the 150Hz data; the circle markers and thin line represent 25Hz sample points and reconstructed signal. From top to bottom, the graphs represent the vertical, mediolateral and anterior-posterior channels. Some features of the data were completely lost while others were modified. Most noticeable in this sample, there was a repetitive loss of a small feature in the mediolateral trace as well the truncation of a number of other features.
Fig. 7.16  Comparison of 21kmh⁻¹ accelerometer data sampled at 150Hz/10bit and at 75Hz/7bit. From top to bottom, the graphs represent the vertical, mediolateral and anterior-posterior channels. Small visual differences are appearing, particularly in the mediolateral signal. Removing the 3rd Least Significant Bit (LSB) appears to be degrading high intensity signal.

Fig. 7.17  (a) Comparison, Walking Data (3kmh⁻¹) sampled at 150Hz 10-bit compared to same data sampled at 75Hz and 7 bits (zeroing three least significant bits). (b) FFT comparison for the acceleration data in (a). While 150Hz/10bit and the 75Hz/7bit traces in the vertical and mediolateral channels (top and bottom) are reasonably similar, the loss of the additional bit of resolution severely impacts the low-level mediolateral signal. The FFT results indicate that the 75Hz/7bit data has the same frequency content at about half the magnitude. The mediolateral is more severely impacted with magnitudes dropping considerably.
Fig. 7.18  Change in Triaxial Accelerometer Counts (AC) after halving the sampling rate from 150Hz to 75 Hz and reducing the sample bits from 10 to 7 bits. Left to right the features represent the AC for walking (3kmh$^{-1}$ and 5kmh$^{-1}$) followed by jogging and running (7kmh$^{-1}$ to 21kmh$^{-1}$, 2kmh$^{-1}$ increments). There appears to be no significant change in AC at the low intensity activities but at the higher speeds the AC are reduced for the 75Hz sampling.

**Sampling System Summary**

From the available data, which included hardware filtering to remove impact spikes, reducing the bits per sample from 10-bits to 8-bits did not appear to have a detrimental impact. At very low activity levels, while reducing the bits-per-sample made no apparent significant change in the time domain information, there appeared to be a small loss in the frequency domain (Fig. 7.14) for the mediolateral axis. Inspection of this data suggested that due to the very low intensity on the mediolateral axis this signal might already have a high noise component.

Aggressive reduction in sampling frequency from 150Hz to 25Hz (Fig. 7.15) appeared to retain key features but lose some potentially important (from a biomechanical viewpoint) sub-features. To begin to recapture these features required at least a doubling of the sample rate. The use of a 75Hz sample rate appeared satisfactory although combining 75Hz sampling with 7-bits per sample was unsatisfactory, the combination making small but noticeable changes in the time domain data during high intensity activity and almost erasing the mediolateral signal for low intensity activity.

Overall, for the available data, a reduction to 8-bits per sample combined with a sample rate of 75Hz appeared satisfactory. It should be noted that the available data only includes running up to 21kmh$^{-1}$ while a world-class sprinter can reach nearly twice that speed. A higher sample rate may be necessary for complete data capture.
7.5.3 Silence Compression

The silence compression of Fig. 7.2 was implemented and run on all 10x10 treadmill data sets for two different sample rates (150Hz and 75Hz) and for four different bit rates at each sample rate (10, 9, 8 & 7 bits per sample). The output data was differenced and the percentage of zero differences compared to the percentage of zero differences for the same data sets but without silence compression. The results for each channel were kept separate. The results from this analysis appear in Fig. 7.19.

Silence compression provided the greatest benefit during periods of low activity and at higher sample rates with fewer bits per sample - i.e. 150Hz 7-bit sampling Silence compression appeared to provide a noticeable improvement compared to 75Hz 10-bit sampling. The higher sample-rate also resulted in an overall increase in zero-differences, ranging from a few percent, to close to 20% in some cases.

![Graph showing impact of silence compression on total number of zero-difference samples.](image)

One aspect of the shape of the graphs of Fig. 7.19 was indicative of results from the energy expenditure analysis of accelerometer counts from Chapter 5. Although not
mathematically related, the graphs indicated similar trends. The vertical-channel showed an almost flat response for the running speeds from 9km\textsuperscript{-1} to 21km\textsuperscript{-1} while the mediolateral channel, particularly at 150Hz, had a reasonably linear response across all speeds.

**Silence Compression Summary**

Silence compression, which was applied pre-differencing, improved the post-differencing percentage of zero-differences. This was important for subsequent minimum-redundancy encoding and/or run length encoding. The improvements were most noticeable in the mediolateral channel - due mostly to the lower intensity of activity on this channel. Silence compression made the most impact at 7-bits per sample and the least at 10-bits per sample. As noted earlier, 7-bits per sample results in the removal of small features and when combined with silence compression may result in the removal of other features. It would also appear that for 10-bit sampling, the least significant bits consist mainly of considerable inconsequential noise and silence compression is largely ineffective. Both 9-bit and 8-bit sampling benefited from silence compression.

**7.5.4 Differencing.**

The PDF for differences in 150Hz-sampled, 10-bit data (Fig. 7.20) indicated that the number of symbols present in differenced data had dropped from in excess of 200 symbols present in the raw data (from Section 7.2.1.1) to around 60 significant symbols. The count of zero differences was nearly twice the count of any other difference. The distribution had a noticeable asymmetry with a longer tail to the right. This data was from an athlete running at 21km\textsuperscript{-1} and the larger positive differences were due to the sharp rise time associated with the vertical acceleration immediately after initial foot contact. Fig. 7.21 is the same test data as Fig. 7.20 but with the least two significant bits set to zero. The number of symbols present dropped from approximately 60 symbols down to 17 significant symbols. Even during this high intensity activity, a high proportion of differences were represented by just three symbols (-1, 0, +1).
For low intensity activity, the number of significant symbols present in the differenced 150Hz 10-bit data dropped to around 21 (Fig. 7.22 (a)) and for 8-bit data this dropped further to around 5 (Fig. 7.22 (b)).
Probability Density Function for Multi-Speed Differences.

The multi-speed PDFs of differences of Fig. 7.23 and Fig. 7.24 provided an indication of how the distribution of differences changed as activity intensity, in this case running speed, increased. Both the figures represent 8-bit data. For 150 Hz sampled data, a significant number of samples could be represented with just three symbols (-1, 0, +1). As running speed increased the number of required symbols increased and the weighting of the three key symbols reduced (but was still significant).

At a lower sampling rate of 75Hz, the -1, 0, +1, differences still predominated at walking speeds but at high running speeds these differences were only marginally more important than those surrounding them.
Fig. 7.24 Histogram of sample differences for speeds 3-21km\(^{-1}\) for 75Hz, 8bit data. At this samplerate, zero difference is no longer predominating, from 11 to 21km\(^{-1}\) zero differences are on a par with the adjacent differences. The reduced samplerate has reduced the number of samples to store but has increased the number of symbols necessary to represent them.

**Differencing Summary**

The analysis of Fig. 7.4 identified that loss-less differencing clearly reduced the size of the symbol set necessary to completely describe the data. Fig. 7.21 & Fig. 7.22(b) identified the benefit of reducing the number of bits-per-sample with the concentration of samples into a much smaller set of symbols. For the purpose of developing a static minimum-redundancy coding scheme the PDF of the differencing output was calculated for a range of athletes and a range of activity intensities. The differences for both 75Hz and 150Hz sampling appeared to have a gaussian distribution about a mean of zero-difference. With increasing activity intensity (speed) the probability of the mean reduced while the standard deviation increased. For any particular activity intensity, the higher sample rate (150Hz) results in a higher probability of the mean, along with a smaller standard deviation.

While no single two-dimensional model of the PDF gave optimal minimal redundancy compression, due to the consistency of the shape of the PDF a compromise model should result in reasonable compression.
7.5.5 Companding

The companding step is dependent on the distribution of differences, which in turn is dependent on the output of the sampling system. The measure of entropy of the system gave an indication of the number of bits necessary to represent the data at different running speeds, and the PDFs indicated the distribution of message symbols at different speeds. To determine how many companding levels were necessary, the cumulative sum of PDFs of Fig. 7.23 (150Hz) and Fig. 7.24 (75Hz) were generated as Fig. 7.25 and Fig. 7.26. These cumulative percentages indicate the number of symbols (companding levels) necessary to represent 100% of samples for the particular sampling system.

As the signal is an oscillating wave, and the orientation of the sensors cannot be guaranteed, the differencing output must be treated as symmetrical about zero differences. This in turn means that there are an odd number of output levels from differencing and companding. The following figures give results for an odd number of output levels, using the contiguous differencing levels of Fig. 7.23 and Fig. 7.24.

Fig. 7.25 Distribution of number of symbols necessary to represent differenced athlete data sampled at 150Hz, 8-bits per sample.
At 150Hz sample rate, 15 output levels can virtually completely represent the 21km$^{-1}$ running data, with 3 levels representing approximately 97% of low intensity activity such as 3km$^{-1}$ walking data. For 75Hz sampled data, the larger spacing between samples resulted in a higher probability of changes in the signal magnitude and increased the range of differences appearing in the signal. The companding system is dependent on the sampling system and ultimately related through the system entropy. Entropy cannot be considered an absolute measure due to qualitative nature of the assessment of the signal. The combined effect of the sampling and companding on the regenerated output was analysed using various combinations of sample-rate and companding systems as shown in the following figures.
Fig. 7.27 Reconstructed companded output compared to source data. In this comparison, data from an athlete running at 21kmh\(^{-1}\) is sampled at 8-bit / 75-Hz, then differenced and companded into a maximum of 15 levels. The difference between this data and the source data (150Hz 10bit) was a function of both the sampling and the companding. The 15 levels consisted of: 0, ±1,2,3,4,6,8,16. The overshoot on the reconstructed data was due to the addition of a difference and the remainder from the companding of the previous difference.

Fig. 7.28 Aggressive companding of 150Hz 8-bit data using only seven companding levels consisting of 0, ±1, 2, 4. At 3kmh\(^{-1}\) the difference between raw data and differenced and companded data was not observable. At 21kmh\(^{-1}\) the 7-level companded data was not tracking the fast changes.
Fig. 7.29  Aggressive companding of 75Hz 8-bit data using only seven companding levels consisting of 0, ±1, 4, 8. With fewer output levels available the output cannot track fast changes in acceleration and filtering of the output occurs. Both small and gross detail was lost.

**Compander Summary**

The 7-level companding of 150Hz 8-bit 21kmh$^{-1}$ running data (Fig. 7.28) almost completely represented the data, as did the 15-level, 75Hz 8-bit results of Fig. 7.27. The 7-level companding of 8-bit 75Hz data of Fig. 7.29 lost some fine and gross detail. In these cases the number of bits required to represent the output came close to the entropy level of the system. From Fig. 7.12, the entropy for 150Hz 8-bit data at 21kmh$^{-1}$ was just over three bits. Seven compander levels could be represented in 3 bits but because only 7 levels were used, the compander was using fractionally less than three bits. This difference between compander bits and entropy bits was represented in the lost signal data.

The primary purpose of the compander was to limit or fix the number of symbols present in the differenced data without noticeably affecting the reconstructed output signal. The results from this analysis indicated that companding could produce the desired result but there was an interaction between the sample rate and the number of compander output levels. At 150Hz, 15 levels were satisfactory but 7-levels was just insufficient. At 75Hz, the combination of sample rate and companding suggested that 15-levels was marginally insufficient.
7.5.6 Minimum Redundancy Coding

A number of minimum redundancy coding schemes were developed to match the PDF at different activity intensity levels or for different mixes of activities. These schemes were all developed to give a maximum code length of 8-bits to ensure that data expansion never occurred due to some combination of low probability symbols. The number of code bits for 17 differencing or compander levels were depicted in Fig. 7.30. In some cases additional codes were available; in other cases the 17 codes were the total codes available in the scheme. Compression ratios for each coding scheme were calculated for a variety of activities with the results appearing in Fig. 7.31 and Table 7.3 for 150Hz sampled and silenced data, and Fig. 7.32 and Table 7.4 for 75 Hz sampled and silenced data.

![Comparison of code bits per compander level for various minimum redundancy coding (MRC) schemes.](image)

Code tables can be automatically generated, using programs such as those described in [24], or manually calculated. Automatically generated code tables varied across athletes, data channels and activities. For estimating compression, the actual minimum redundancy codes were not important but the number of bits required to describe any particular differencing or compander output, together with the probability of a particular sample difference determined the overall compression. Comparing the schemes of Fig. 7.30 with the PDFs of Fig. 7.23 and Fig. 7.24, the relationship between the minimum redundancy code bits-per-sample-difference and the distribution of probability of sample differences became obvious.
A scheme designed to maximise compression for low intensity activity (scheme 1) obtained good compression (77%) for low intensity activity with the 1.84 bits per sample approaching the entropy level (compare Fig. 7.12). For high-speed activity this model performed poorly, exceeding the entropy level by nearly 0.5 bits per sample. To generate the high compression, scheme-1 utilised a combination of 1-bit, 2-bit and 3-bit codes to represent the three highest probability differences. This severely limited the number of short codes available for other symbols/differences.

Table 7.3 Percent compression (bits per sample) for 150 Hz 8-bit data using static encoding models

<table>
<thead>
<tr>
<th>Minimum Redundancy Model</th>
<th>Activity Mix</th>
<th>Difference from Best Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LS 3,5,7</td>
<td>All A&amp;I</td>
</tr>
<tr>
<td>Model 1</td>
<td>77 (1.84)</td>
<td>74 (2.09)</td>
</tr>
<tr>
<td>Model 2</td>
<td>75 (1.99)</td>
<td>73 (2.16)</td>
</tr>
<tr>
<td>Model 3</td>
<td>73 (2.17)</td>
<td>71 (2.35)</td>
</tr>
<tr>
<td>Model 4</td>
<td>69 (2.44)</td>
<td>68 (2.54)</td>
</tr>
<tr>
<td>Model 5</td>
<td>62 (3.01)</td>
<td>62 (3.07)</td>
</tr>
</tbody>
</table>

Note: LS 3,5,7 indicates low-speeds of 3, 5 & 7 kmh⁻¹, HS indicates high-speeds, All A&I indicates compression of the whole test sequence including all activity and inactivity. Best compression for each activity type highlighted. 'Difference from best fit' is a simple arithmetic calculation of sum of percentage difference for this model to the model with maximum compression.
When applied to 75Hz-sampled data, the same minimum redundancy coding schemes gave different results. Scheme-1 still achieved the highest compression for low intensity activity but for high intensity activity the performance was very poor with compression of 4.92 bits per sample, approximately 1-bit worse than the entropy value. For 75Hz data, scheme-5, which was designed for high intensity activity gave the best performance, with 4 bits per sample or 50% compression, close to the optimal value. Scheme-5 gave the most consistent result but the optimal scheme depended on the mix of activity intensity. Scheme-5 cannot exceed 62.5% compression of 8-bit data as the shortest codes were 3-bits.

Table 7.4 Percent compression (bits per sample) for 75Hz 8-bit data using static encoding models.

<table>
<thead>
<tr>
<th>Minimum Redundancy Model</th>
<th>Activity Mix</th>
<th>Difference from Best Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LS 3,5,7</td>
<td>All A&amp;I</td>
</tr>
<tr>
<td>Model 1</td>
<td>71 (2.36)</td>
<td>67 (2.62)</td>
</tr>
<tr>
<td>Model 2</td>
<td>69 (2.45)</td>
<td>68 (2.59)</td>
</tr>
<tr>
<td>Model 3</td>
<td>70 (2.43)</td>
<td>66 (2.75)</td>
</tr>
<tr>
<td>Model 4</td>
<td>67 (2.65)</td>
<td>64 (2.84)</td>
</tr>
<tr>
<td>Model 5</td>
<td>62 (3.07)</td>
<td>59 (3.24)</td>
</tr>
</tbody>
</table>

Notes for Table 7.3 also apply for this table.
7.5.7 Run Length Encoding

From the previous sections, it was identified that zero differences predominate, therefore zero differences were the most likely symbols to form into significant runs. This is particularly true when activity intensity levels were low. The analysis from Fig. 7.33 indicated the distribution of different run-lengths of zero-differences at the different running speeds.

![Fig. 7.33 Percentage of Zero-Difference by run-length and athlete walking/running speed. Data is 150Hz / 8bit with silence compression on. The existence of runs of zeros even at the higher activity intensities suggested that some form of run length encoding would be useful.](image)

From the RLE discussion of Section 7.4 and assuming that a Huffman escape code of 5 bits can be made available, runs of 10 or more zeros can be beneficially encoded using RLE. According to Fig. 7.33, for 150Hz sampling and during low intensity activity such as 3kmh\(^{-1}\) walking, approximately 14% of samples occurred in runs of 10 or more zeros. This percentage increased during periods of no activity, and decreased to only one or two percent during periods of high activity. In the case of the minimum redundancy code Scheme-1 of Fig. 7.30 the use of a 5-bit escape code would be prohibitive as only two 5-bit codes were used and these represent the differences of ±2.
Either of these differences represented around 10% or more of samples at any running speed (Fig. 7.23 & Fig. 7.24). Using a 5-bit escape code in this case would further improve low activity-intensity compression at a cost to the high activity intensity compression. The reduction in the compression ratios due to the use of a short code for an escape code is given in Table 7.5. This table gives the reduction in compression percentages from the values given in Table 7.3 & Table 7.4. For comparison, an example of the benefit of RLE for $3\text{kmh}^{-1}$ activities is also presented. This compression gain was only relevant for 150Hz sampled data at $3\text{kmh}^{-1}$ and was for all channels from one athlete. The benefit would vary from athlete to athlete and from channel to channel. The benefit reduced rapidly as walking speed increased and depending on the mix of activities could be detrimental. Scheme-5 showed considerable benefit despite needing to use a 6-bit code as an escape code. This occurred as scheme-5 used 3-bits to represent a zero difference and the cost benefit changeover occurred with a run length of 4 successive zero differences.

Table 7.5 Effect on compression ratio of adding RLE.

<table>
<thead>
<tr>
<th>Minimum Redundancy Model</th>
<th>Compression Loss % for Activity Mix (150Hz, 75Hz)</th>
<th>Low Activity Compression Gain for RLE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LS 3,5,7</td>
<td>All A&amp;I</td>
</tr>
<tr>
<td>Model 1</td>
<td>-1, -2</td>
<td>-1, -1</td>
</tr>
<tr>
<td>Model 2</td>
<td>0, 0</td>
<td>0, -1</td>
</tr>
<tr>
<td>Model 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model 4</td>
<td>0, 0</td>
<td>0, 0</td>
</tr>
<tr>
<td>Model 5</td>
<td>0, 0</td>
<td>0, 0</td>
</tr>
</tbody>
</table>

Notes for Table 7.3 also apply for this table.
Additional Notes: Models 1, 2 & 4 sacrificed the least used 5-bit code for use as an escape code. Model 5 sacrificed the least used 6-bit code. Compression Gain was calculated from sample 150Hz athlete data for $3\text{kmh}^{-1}$ walking activity. During inactivity, the compression ratio for any model could rise as high as 96%.

From Table 7.5 it appeared that in some cases the benefit of RLE was outweighed by the cost. This was dependent on the proportion of inactivity or low-activity to high intensity activity. It could be assumed that for many athletes, training consists of long periods of high intensity activity and the compression trade-offs are insufficient to
justify the added complexity of applying RLE. As RLE for athlete data was only concerned with the compression of zero-differences, a further analysis of the effective compression of zero-differences appears in Table 7.6. This covers different sample rates and includes the effect of implementing silence compression. From this analysis, the combination of silence compression, RLE and a short break-even run length could give considerable percentage improvement for low intensity activities. Higher intensity activities benefited only marginally, if at all, depending on the scheme implemented. For example, for 75Hz sampling and a 9kmh⁻¹ running speed, no RLE scheme resulted in any significant improvement in compression over the MRC. At 150Hz sampling, the best improvement for 9kmh⁻¹ data was less than 3%.

Table 7.6 Comparison of percent compression of zero-differences for 3kmh⁻¹ sample data using combined MRC and RLE (9kmh⁻¹ data bracketed).

<table>
<thead>
<tr>
<th>Percent Compression</th>
<th>150Hz Sample Rate</th>
<th>75Hz Sample Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Data</td>
<td>With silence</td>
<td>Raw Data</td>
</tr>
<tr>
<td>1-bit MRC for zero diff.* No RLE</td>
<td>Maximum Compression Possible 87.5%</td>
<td></td>
</tr>
<tr>
<td>RLE 5-bit escape + 4-bit counter</td>
<td>88.3</td>
<td>90.2 (87.5)</td>
</tr>
<tr>
<td>3-bit MRC for zero diff.* No RLE</td>
<td>Maximum Compression Possible 62.5%</td>
<td></td>
</tr>
<tr>
<td>RLE 6-bit escape + 3-bit counter</td>
<td>75.3</td>
<td>81.2 (65.3)</td>
</tr>
<tr>
<td>RLE 6-bit escape + 4-bit counter</td>
<td>74.4</td>
<td>81.3 (64.7)</td>
</tr>
<tr>
<td>RLE 6-bit escape + 5-bit counter</td>
<td>73.3</td>
<td>80.6 (64.1)</td>
</tr>
</tbody>
</table>

Note: Figures in brackets indicate effective RLE compression at 9kmh⁻¹.

*Runs of 10 or more improve compression for 1-bit zero coding, runs of 4 or more improve compression for 3-bit zero coding. MRC = Minimum Redundancy Code.

For comparison with the byte-oriented RLE of Fig. 7.10 the percentage of zero-difference runs of three or more is identified in Fig. 7.34 In a scheme that does not utilise variable length codes, RLE could provide some compression for up to 55% of samples (during activity - more if stationary). This could provide substantial compression for activities such as football, where the athlete spends considerable time at very low activity intensity levels (refer Fig. 5.12, reproduced here as Fig. 7.35)
Fig. 7.34  Comparison of percentage of zero-differences in runs of three or more for silence compression ON and OFF. Data is from all athletes, all channels and shown for each test speed. Sample rate was 150Hz and sample size was 8 bits. All these samples can benefit from run length encoding.

Fig. 7.35  Histogram of accelerometer detected activity intensity from football athlete. 50% or more of on-field time is spent at lower activity intensity levels.

**Run Length Encoding Summary**

The RLE schemes provide no effective compression benefit for any high intensity activity (i.e. running) and in some cases would reduce compression significantly (3%). RLE's benefit lies in the compression of very-low activity or inactivity data. When combined with 'scheme-5' from the previously assessed minimum redundancy codes, the reduction in compression for high intensity activity was negligible but the improvements in compression for low intensity activity brought this compression scheme up to comparative levels with schemes that were designed specifically for low intensity activity. Silence compression benefited all forms of RLE and for some schemes the improvement exceeded 5%.
7.6 Analysing Combined Compression Techniques

Individual compression techniques and combinations of techniques can be used to obtain particular levels of compression. The achieved levels of compression may be affected by the specific implementation or desired end use. A system required to both wirelessly transmit data in real-time and concurrently log the data may use different compression techniques for the different channel types. From the analysis in the preceding sections, it was apparent that the system must use matched processing components, for example, a system that implements a lower sampling rate but monitors a high proportion of high intensity activity must use companding levels and minimum redundancy codes tuned for that application, in order to maximise compression and minimise data loss. Conversely, a system monitoring low intensity activity will benefit from processing tuned for low intensity activity.

This section presents an analysis of the combination of the discussed compression techniques as well as the interaction between the compression schemes and the implementation of wireless or wired data channels.

7.6.1 Combined Compression Techniques.

Based on the foregoing analysis, implementing silence compression can provide several percent of compression improvement in RLE and minimum redundancy coding. If either of these forms of compression are implemented, silence compression should be included if the processing power is available. Silence compression does not directly benefit differencing or companding.

Analysis of the reconstructed waveforms from 7-level companded 150Hz data (Fig. 7.28) and 15-level companded 75Hz data (Fig. 7.27), along with the calculated entropy levels indicated that both these forms of companding were marginally inadequate and further, that at 75Hz some small features were lost. Alternative sample rates and compander output combinations, along with the resultant bandwidth requirement are tabled in Table 7.7. These combinations either increase the sample rate, to reduce the range of differences, or increase the companding bits to cater for the wider range of differences that exist at lower sampling rates. While the 75Hz sampling provides the
lowest bit rate, the marginal increase in cost of 100Hz sampling is offset by the benefit of improved signal tracking.

Table 7.7 Data rates based on sample rate and bits-per-sample.

<table>
<thead>
<tr>
<th>Sample Rate - Sample bits</th>
<th>Bits per Sample After Differencing &amp; Companding</th>
<th>Channels</th>
<th>Output Bit Rate (bps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 - 8</td>
<td>3</td>
<td>3</td>
<td>1800</td>
</tr>
<tr>
<td>100 - 8</td>
<td>4</td>
<td>3</td>
<td>1200</td>
</tr>
<tr>
<td>75 - 8</td>
<td>5</td>
<td>3</td>
<td>1125</td>
</tr>
</tbody>
</table>

Using 100Hz 8-bit data differenced and companded to 15 levels (or 4-bits per sample), the output was subject to minimum redundancy encoding using a fixed encoding scheme (scheme-2 of Fig. 7.30). The resultant compression is given in Table 7.8.

Table 7.8 Compression of silenced / differenced / companded data using minimum redundancy encoding

<table>
<thead>
<tr>
<th>Speed as a measure of activity intensity</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 kmh⁻¹</td>
</tr>
<tr>
<td>Channel</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>All</td>
</tr>
</tbody>
</table>

Note: Input data consisted of 15 seconds of 100Hz sampled data for athlete 2 (from 10x10 trial), differenced and companded into 15 levels (4 bits), input bits = 6000 bits per channel.

Minimum redundancy encoding of the entire test file (inactivity + speeds 3-21kmh⁻¹) used in the analysis of Table 7.8 resulted in 41.6% compression of compander output or 70.8% compression of the sampled data (or 2400bps reduced to 700bps). Using RLE compression of the compander output resulted in 25.4% compression, the equivalent of 62.7% compression of the sampled data (or 895bps). Note that for RLE of the 15-level, 4-bit compander output, the 16th four-bit code is the escape code and runs of three or more zero differences benefited from RLE. Fig. 7.34 is indicative of the number of samples that could benefit from RLE of the compander output.
Different Data Sets:
For further analysis of the effectiveness of the proposed compression techniques, data from different tests sets comprising many different sporting activities was analysed individually and in combination. This data was compressed using different techniques for comparison. In some cases a single fixed minimum coding table was used to compress data from a number of different test data sets. This data was compared to compression achieved using standard Huffman compression on each data set individually, and on compression using PKZIP, an adaptive system combining dictionary and Huffman schemes. From Table 7.10 & Table 7.11, the benefits of an adaptive scheme, in this case PKZIP, were as high as 6% additional compression.

Table 7.9 Compression ratios for combined 150Hz test data using combined compression techniques.

<table>
<thead>
<tr>
<th>Data Format</th>
<th>Huffman Compression</th>
<th>Huffman plus Silence</th>
</tr>
</thead>
<tbody>
<tr>
<td>9-bit Diffuse Data</td>
<td>25%</td>
<td>26%</td>
</tr>
<tr>
<td>8-bit Diffuse Data</td>
<td>46%</td>
<td></td>
</tr>
<tr>
<td>Above + Differencing</td>
<td>60%</td>
<td></td>
</tr>
<tr>
<td>Above + Companding</td>
<td>76%</td>
<td>78%</td>
</tr>
</tbody>
</table>

Table 7.10 Compression ratio statistics for combined 150Hz 8-bit test data.

<table>
<thead>
<tr>
<th>Compression Scheme</th>
<th>Percent Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>41 symbol companding &amp; fixed Huffman code table.*</td>
<td>Mean 69.4, Min 60.1, Max 80.8, SD 5.9</td>
</tr>
<tr>
<td>Standard Huffman encoding (2-pass)</td>
<td>Mean 71.2, Min 63.6, Max 82, SD 5.4</td>
</tr>
<tr>
<td>PKZIP [18]</td>
<td>Mean 73, Min 66, Max 83, SD 4.6</td>
</tr>
</tbody>
</table>

Note: Test data comprised sampled data from 23 different sporting activities.

*The Fixed Huffman code table was developed from the combined test data but modified for a maximum code length of 8-bits.

Table 7.11 Compression of data from athlete engaged in football training (150Hz 8-bit).

<table>
<thead>
<tr>
<th>Compression Scheme</th>
<th>Percent Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>41 level companding &amp; fixed Huffman code table</td>
<td>71</td>
</tr>
<tr>
<td>Standard Huffman coding (2-pass)</td>
<td>71.6</td>
</tr>
<tr>
<td>PKZIP [18]</td>
<td>74</td>
</tr>
</tbody>
</table>

Note: This test used the same compander and Huffman code table as Table 7.10.

7-48
Reducing the number of compander levels (from the 41 used in Table 7.10 & Table 7.11) to 15, improved the compression results as indicated in Table 7.12. As indicated in earlier results, silence compression could be used to improve compression another one or two percent. The use of a fixed Huffman coding table, as opposed to a Huffman scheme generated for a specific activity, appeared to cost around zero and two percent compression (from Table 7.10, Table 7.11 & Table 7.12).

From the results of Table 7.8 & Table 7.12, differences in compression across the channels were observed. This was expected given the percentage of zero differences in Fig. 7.19 In the small number of examples given in this chapter, up to 8% variation in compression across channels was observed.

Table 7.12 Compression techniques compared by sport and data channel.

<table>
<thead>
<tr>
<th>Test activity</th>
<th>Running &amp; turning</th>
<th>Basketball</th>
<th>Football defence</th>
<th>Running up stairs</th>
<th>Running &amp; tackling</th>
<th>Treadmill various speeds</th>
<th>Rowing various speeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huffman Fixed Coding</td>
<td>Ch1 80 75 74 78 78 78 86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ch2 77 76 73 74 74 73 85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ch3 79 75 75 78 78 75 84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total 79 75 74 77 77 76 85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Huffman Total</td>
<td>Total 80 77 76 77 78 77 85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Data is 150Hz, 8-bit differenced data using 15-level compander. For the Huffman - fixed coding table - the table of codes was generated from the compander output using the combined test data. For the standard Huffman compression, each file was compressed individually and therefore used a scheme optimised for that test file.

**Combined Compression Techniques Summary**

The proposed combination of compression techniques extracted reasonable levels of compression without apparent detriment to the reconstituted data signal. The use of a fixed, minimum redundancy coding table, as opposed to a data specific code table, appears, in the worst cases, to decrease the potential compression by a couple of percent. The use of silence compression appears to benefit minimum redundancy based compression by a one or two percent, rising to 7% when combined with RLE (for some data models).
7.6.2 Compression for wireless transmission on an unreliable link.

Data transmission on an unreliable link imposes a number of constraints on the compression of data. In particular, three components of the compression system analysed in this chapter need special consideration:

- Differentiating.
- Adaptive Schemes (including any scheme that 'escapes' to another scheme.
- Minimum redundancy coding using variable length codes.

In conjunction with the above aspects, two other factors relating to the packetisation of data also require analysis:

- Instantaneous compression rates impact effective data packing.
- Interleaving or non-interleaving of channel data affects data packing.

The discussion of these points is independent of any implementation of forward error correction or error detection codes.

7.6.2.1 Effect of Differentiating

The use of differentiating requires that any packet must include an absolute reference value to ensure that loss of a data packet does not affect subsequent data packets. Differentiating is affected by intra-packet corruption, but provided the remains of the packet can be decoded, the corruption of a difference may cause a transient dynamic signal and a subsequent change in absolute level but retain other information of interest. Intra-packet errors may be recoverable depending on subsequent data packets and a combination of error correction/detection and an understanding of the information content.

7.6.2.2 Limitation of Adaptive Schemes

As stated previously, the use of adaptive schemes is inappropriate for wireless data where there is no \textit{guaranteed} transfer of data. The loss of any packet containing adaptive coding changes will corrupt subsequent data. The use of escape codes to insert RLE data falls into the category of adaptive schemes as an errored escape code or another code incorrectly decoded as an escape code changes the interpretation of following data.
7.6.2.3 Minimum Redundancy Codes

Minimum redundancy codes suffer severely from intra-packet corruption. The corruption of a single data bit can completely corrupt the rest of the data packet, as variable length codes do not have defined symbol boundaries. Examples of the effect of bit errors are described using Table 7.13, Fig. 7.36 & Fig. 7.37 below. This form of error does not affect fixed length codes where a single bit error does not propagate through subsequent codes (except in the case of adaptive codes).

### Table 7.13 Differences mapped to variable length codes (for use in Fig. 7.36)

<table>
<thead>
<tr>
<th>Difference</th>
<th>Variable Length Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>+3</td>
<td>0100010</td>
</tr>
<tr>
<td>+2</td>
<td>01011</td>
</tr>
<tr>
<td>+1</td>
<td>00</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>011</td>
</tr>
<tr>
<td>-2</td>
<td>01001</td>
</tr>
<tr>
<td>-3</td>
<td>0100000</td>
</tr>
</tbody>
</table>

Partial table only, not all mappings displayed.

INPUT:

Differences: 0 0 +1 +1 +2 +1 0 -1 -2  (total = +2 in 9 samples)
Huffman Code: 1 1 00 00 01011 00 1 011 01001

**Example 1**

After Error: 1 1 01 00 01011 00 1 011 01001

^ errored bit

As decoded: 1 1 0100010 1 1 00 1 011 01001
Differences: 0 0 +3 0 0 +1 0 -1 -2  (total = +1 in 9 samples)

**Example 2**

After Error: 1 1 10 00 01011 00 1 011 01001

^ errored bit

As decoded: 1 1 1 00 00 1 011 00 1 011 01001
Differences: 0 0 0 +1 +1 0 -1 +1 0 -1 -2  (total = -1 in 11 samples)

Fig. 7.36 Examples of impact of bit errors on the decoding of variable length codes. Codes used are from Table 7.13. In example one, the error results in decoding the same number of samples but different data. The error of example two results in both different data and a different count of samples. Spaces in the bit streams are not part of the data but are for ease of viewing.
Fig. 7.37 Input signal compared to output signals after single bit error. This figure was derived from the data of Fig. 7.36 and for the generation of the graph assumes a start of zero offset. Depending on the sequence of bits and the location of the errored bit, the output of an errored packet could completely change the information content.

7.6.2.4 Instantaneous Compression Rates

Many forms of wireless network are bandwidth constrained (refer Chapter 2). Where a network is bandwidth constrained and using, for example, time division multiplexing, the data must be arranged in fixed size packets. Because compressed data is of a variable density, the fixed packet size will not exactly match the current compressed data bit-rate. Packet sizes designed for 60% compression will be under utilised at higher compression and, even when combined with additional sensor buffering, may be inadequate when handling prolonged high intensity activity (with subsequent lower compression).

This mis-matching of compressed data to packet size exists even where the subsystems have been designed to prevent instantaneous data expansion. An example of the reconstructed compressed data and the instantaneous compression ratio for different packet sizes (by count of samples) appears in Fig. 7.38, Fig. 7.39 & Fig. 7.40 These figures compare source data with compressed data and the instantaneous compression ratios for blocks of 8 or 16 samples.
Fig. 7.38  Comparison of 150 Hz 8-bit decompressed data and source data from 3-tackle sequence test data (refer chapter 4). This data is compressed using differencing and 15 companding levels. At this viewing level there is no discernable difference between source data and the compressed-decompressed data.

Fig. 7.39  Instantaneous compression ratio using blocks of 8 samples. This data matches the section of data from Fig. 7.38. Because the codes are limited to 8-bits, data expansion cannot occur. The maximum compression possible is 87.5%.

Fig. 7.40  Instantaneous compression ratio using blocks of 16 samples. This data matches the section of data from Fig. 7.38 & Fig. 7.39. Because the block size is larger than that of Fig. 7.39 there is less variation in the resultant block compression.
7.6.2.5 Channel interleaving.

The use of channel interleaving (Fig. 7.41) simplifies the encoding and decoding of data into a single data stream by removing the necessity to track the filling of separate buffers for each channel. Due to the use of variable length codes and the difference in compression ratios between channels, maintaining a separate buffer for each channel is akin to the problem described in the previous subsection i.e. packing the variable length data into separate packets for each data channel and subsequently combining these packets into a single packet of a set length.

Against interleaving is the problem identified in Fig. 7.36, any one-bit error has the potential to totally corrupt subsequent codes. In the case of interleaved codes, a single-bit error could not only corrupt the codes but also cause the interleaving to get out of sequence with decoded data incorrectly allocated to the various channels.

![Interleaving of channel data including absolute reference values and variable length codes.](image)

Combining RLE compressed variable length codes from multiple channels increases the complexity of channel interleaving. With RLE, the run length is unknown until the first non-zero difference is encountered or, sufficient zero-differences to fill one counter have occurred. Therefore the encoding with variable length codes requires a latency of at least one run counter (a run of 24 for the first 4-bit counter). To minimise the complexity of encoding a counter, the count should be complete since the counter is encoded into a bit-stream and may be stored anywhere within a memory word or across memory word boundaries (making it complex to update an already bit-stream encoded counter.

If interleaving is used then the interleaving will be complicated by the absence of data from one channel. Further, while one channel may have detected an encode-able run,
the count of bits from the other channels may reach the packet boundary with fewer represented samples.

The complexity of using RLE with interleaved data in fixed size packets would appear to exceed the benefit of using RLE. If runs of zero-differences cross packet boundaries, there may be no compression benefit in using RLE.

The value of interleaving data must be weighed against the possibility of intra-packet corruption. If intra-packet corruption is a real possibility then variable length data should not be interleaved but encoded into sub-packets that map to byte boundaries within the data packet. In this way corruption of variable length codes may be isolated to a single channel. This may result in additional inefficiencies, as the data encapsulation protocol must track the sub-packet boundaries and each channel may have wasted bit storage at the end of its sub-packet.

**7.6.2.6 Alternative compression for wireless transmission.**

A number of the problems of transmitting compressed data are related to the non-deterministic nature of the compression schemes. The varying compression ratio makes data encapsulation more complex. The variable length codes make recovery in the case of error difficult. Data packet loss makes using adaptive schemes risky.

As an alternative, by extracting the output of the compander, a fixed compression ratio can be guaranteed and a deterministic method of data encapsulation implemented. If the compander generates a 4-bit output (15-levels) then two samples can be packed into each 8-bit byte. This system benefits by elimination of the error propagation that occurs with variable length codes. There is no benefit in using interleaving nor is there any penalty. In the event of an intra-packet bit error, the decoding error is isolated to the particular 4-bit value. Since this value represents a signal difference, there will be a signal level error that propagates through the reconstructed signal.

Implementing RLE of the compander output is of relatively low complexity as both the escape code and counter can use 4-bits. This ensures the counter is always aligned with either the upper or lower nibble in a byte. There is low latency for encoding RLE as the breakeven point is two samples and, as the count increases it can easily be
incremented in-situ. RLE in this situation suffers from similar problems to adaptive schemes and makes the packing of data more complex. This complexity is much more manageable than in the case of variable length codes as all the codes used are 4-bit and always pack two codes per byte.

7.6.2.7 Compression for Wireless Transmission Summary

Any form of unreliable transmission of compressed data can cause errors that will propagate to a greater or lesser extent through the reconstructed signal. An error in the reception of an adaptive code could propagate through all subsequent data. An error in an escape code indicating a run-length counter is following, will lead to misinterpretation of subsequent data. Conversely, the existence of an adaptive scheme or RLE means that a corrupted code could be incorrectly decoded as an escape code leading to the incorrect decoding of subsequent data as adaptive content or as an RLE counter (again corrupting subsequent data). A single bit error in a bit stream of variable length codes could propagate across all channels and corrupt all subsequent data in a packet. A single error in differenced (fixed length) data will cause an instantaneous dynamic error and propagate a signal level error to the end of the packet (for one channel only).

Using a 4-bit compander output results in a fixed compression level but overcomes many problems associated with transmission of variable length codes. The quality of the wireless link and the available bandwidth should be considered in the development of the packetisation of compressed data.

7.6.3 Compression for storage or reliable transmission.

Where data is compressed for on-board storage and subsequent download via a reliable wired or wireless connection, the constraints of an unreliable real-time connection do not apply. It can be assumed that the probability of bit errors in the storage medium is so low that they can be ignored. Adaptive variable length codes, interleaving and RLE are all candidates for compression implementation. Restart points can be widely separated, amortising the cost of the restart data over many samples and minimising the effect on overall compression cost.
7.7 **Implementation on sensor platform.**

The compression functionality described in this chapter was implemented using Matlab for analysis, as well as Borland Turbo-C (for Personal Computer use). The C program code was cross-compiled to the target H8 based microprocessor.

To test the correct operation of the compression system, the target system was preloaded with previously collected data. This data was processed by the operating system and compression programs in the normal manner. The output of the data compression stages was recoded with Hamming [27] Forward Error Correction codes (FEC) and output via the sensor platform serial port for capture by a personal computer. The compressed data was decoded and reconstructed for comparison with the original data. An example, using data compressed and transmitted by the target system, showed that the implemented compression & decompression correctly reconstructed the signal and was effective for low and high intensity activity with changing athlete orientation (see Fig. 7.38).

This processing was repeated in various ways and for different combinations of compression. The details of the code size and processing load related to the compression (and FEC) functionality appears in Section 3.4.2. The originally proposed compression was implemented in a small memory footprint and within a modest processing power budget.
7.8 Summary

Accelerometer based athlete data from a variety of activities was analysed for information content and compressibility of the raw signal using traditional compression techniques. The analysis included qualitative and quantitative assessment of the effectiveness of a combination of lossy and lossless compression techniques.

Using the original 150Hz 10-bit data as a comparison baseline, it was identified that, for the test data used, a 100Hz 8-bit sampling system could reproduce the data with sufficient accuracy. Combining this sampling system with differencing and companding to 15-levels, the bits per sample could be reduced to 4-bits, reducing the overall data rate for a 3-channel system from 4500bps to 1200bps with negligible loss of signal quality.

Combining a 4-bit compander output with Huffman minimum redundancy encoding or run-length encoding (or both) provided a further increase in compression of the signal. The compression of both of these methods could be improved by pre-processing the sampled data using silence compression.

The compressibility of the athlete data using minimum redundancy encoding was a function of the intensity of the athlete activity. High intensity activity was less compressible than low intensity activity. To optimally compress a stream of data consisting of blocks of high and low intensity requires an adaptive compression scheme but adaptive schemes are not suitable for wireless transmission on unreliable links. Instead, several encoding schemes were compared across a range of activity intensities. Using a single fixed, minimum-redundancy coding scheme, known to both the data-compressing source and the decompressing receiver, reasonable levels of compression could be achieved.

A number of considerations relating to the transmission of compressed data on an unreliable link were analysed.

Each of the four main objectives identified in the introduction have been achieved.
7.9 Athlete Data Compression Postscript

This chapter used a mixed qualitative / quantitative approach to determining the appropriate operating parameters for a series of connected compression processing steps. The compression steps were chosen for their apparent suitability to the problem of athlete data collection on a low-powered self-contained sensor platform.

Technical difficulties associated with the different compression steps were analysed and different options proposed.

While this chapter covered the technical implementation of compression, due to the patent situation, the implementation of this compression in an exported commercial product may not be advisable without considerable further study. Even though many early compression patents have expired, patents on minor variations of techniques could still be in force. Patents on apparently unrelated algorithms could also be construed to cover some particular implementation. Unfortunately there is no 'quick-reference' chart identifying patent coverage, as the patents are often complex and open to interpretation.

This patent situation is not unique to the question of compression but also impinges on other aspects of the athlete data sensor platform.
7.10 References


8 Concluding Comments

In-situ Athlete Monitoring: Data Collection, Interpretation & Feature Extraction

This thesis tackled a relatively new area of research, the in-situ monitoring of elite athletes. This research was broken into two main areas of investigation, (a) the in-situ athlete monitoring including the data collection and, (b) the data interpretation and feature extraction. These two core components interacted at a number of levels and were not independent of each other.

In-situ Athlete Monitoring: Data Collection

The development of a low-power, wireless network ready, real-time operating system was foundational in the subsequent development of a variety of athlete sensor devices. A large number of other sports monitoring projects were developed on the back of this system. The simplicity of the operating system allowed the easy implementation of the operating system on different hardware platforms, allowing transitions to improved or alternate hardware devices and the application of this system to other projects.

The fusion of data from multiple intra-athlete or inter-athlete sensors was investigated and some rule-of-thumb fusion timing constraints identified. These constraints were predominately associated with other monitoring technologies such as video. These timing requirements fed the design requirements for the implementation of a synchronised, low power wireless network. The ability to minimise power consumption while maximising bandwidth and maintaining synchronisation was investigated for several technology types. Each technology had various impediments to the implementation of a high bandwidth, synchronised wireless network. A general strategy across all technologies involved moving the power-cost of data transfer away from the athlete-based wireless nodes to the static, venue-based wireless nodes.

A technology independent, wireless network topology was designed. This topology required multiple static nodes giving maximum geographic coverage by providing
receiver diversity and maximising the probability of at least one static node capturing a complete, uncorrupted or correctable message from a wireless node.

Athlete data from a wide range of sporting activities was analysed for the purpose of tuning the sampling system. This analysis identified the appropriate sampling system parameters. It also traced the combined impact of the athlete data and the sampling system on various signal compression schemes. The effectiveness of these schemes was analysed in the contexts of data logging and data transfer via wireless networking. Variations of these compression schemes have been implemented in various sports monitoring projects.

**Future Research:**
With the market dominance of the WiFi technology, there has been pressure on both device price and performance. Combining reduced component cost with the large available bandwidth makes this technology appear to be best able to meet the requirements of the wireless sensor network for monitoring teams of athletes. Modelling and analysis of the performance of a statistically based master-controlled synchronisation scheme (as identified in Section 2.6.3 with the sub-heading "Low Power Master-Controlled WiFi Synchronisation"), should be performed. This scheme has the potential to reduce both wireless power-costs and network complexity.

**Data Interpretation and Feature Extraction**

A variety of athlete derived triaxial accelerometer data was analysed and the signal-processing techniques for extracting information of interest assessed against various criteria. For extracting information of particular interest, such as athlete estimated energy expenditure, both traditional accelerometer-based physiological measures (counts) and normal signal processing techniques such as the FFT were inadequate.

**Estimated Energy Expenditure**
The identification of the step-rate as the parameter for estimating energy expenditure of fit athletes during running, simplifies the documentation of fitness training for
athletes and enhances the data toolbox available to sports physiologists. This is beneficial across a range of sporting activities from swimming to rowing to athletics to team sports, as running plays a major part in fitness training. By combining the three factors of orientation, accelerometer signal power and signal frequency, the categorisation of activity in some sports, such as rugby football, is possible. Extraction of these factors using low-power signal processing techniques allowed the implementation of this activity categorisation on an embedded platform operating at a reduced frequency and power.

**Future Research:**
For sports such as the rugby codes and other contact sports, other activities still need quantification. For example, the physiological energy expenditure involved in a rugby tackle requires quantification. For some playing positions, tackles occur frequently and the energy expenditure is unknown. Similarly, in the sport of Australian Football some players are involved in a large number of ball contests in the form of 'marks' or 'rucks'. This generally involves a short run and a vertical leap. The physiological energy expenditure of this activity requires quantification, at least to identify if it requires further consideration. If these activities are identified as important to the total energy expenditure then signal-processing techniques need to be developed to extract this information.

**Running Biomechanics**
The combination of in-sole sensors and triaxial centre-of-mass accelerometers enhanced the ability to interpret the acceleration data and the derived velocity and displacement data. The acceleration data and derived data was highly suggestive of how the athlete's body was moving but due to the athlete's six degrees of freedom, it was not possibly to definitively categorise the kinematic activity. The multi-speed treadmill data further identified the complexities that occur when attempting to categorise the biomechanical activities of runners. For instance, for the speeds tested, the subjects generally showed decreasing vertical displacement as running speed increased but the peak vertical displacement occurred at different speeds for different subjects. When assessing horizontal displacement, some subjects increased horizontal displacement with increased speed, for others the horizontal displacement decreased.
This did not accord with some references on this topic however these differences may be artefacts related to the use of the treadmill.

**Future Research**

Further kinematic-biomechanical analysis is required, particularly using experienced runners in multi-speed overground running. Other related investigation is also required. Monitoring of the running movement using combined accelerometer - gyroscopic and magnetic sensor devices is required to clarify the mediolateral signal and determine the proportion of that signal attributable to pelvic tilting as against actual lateral movement. This research would also identify the range of pelvic tilt and twist associated with the running movement.

Using the above multi-sensor platform would also allow further research into the actual orientation of the sensor platform and the attached athlete. By improving the identification of biomechanical activity, improved isolation of the gravity component from the total accelerometer signal would be possible. This would improve the extraction of the athlete orientation signal.

**Summary**

The research underlying this thesis and the practical implementations of that research has provided additional investigative tools for the sports scientist. The totality of the research embodied here was combined to implement a research tool for a specific field of sporting endeavour, particularly any training or competition activity that is predominately walking or running. Components of this research, particularly the in-situ athlete monitoring components have been combined in other fields of elite sports research. A number of clearly defined areas of future research have been identified
9 Appendices

9.1 Appendix to Chapter 2 (Wireless Networking)

This appendix details licensing information relating to a specific band in various jurisdictions, the Australian Class License conditions for Low Interference Potential Devices (LIPDs) and the various ISM bands available in different jurisdictions.

9.1.1 A Summary of 433.92 MHz License Conditions,

The situation regarding the use of 433.92 MHz has been investigated for a number of countries/regions as shown below.

**Australia:**

Class License for Low Interference Potential Device (LIPD)  (Ref: Ch.2-[61] )

25mW Effective Isotropic Radiated Power (EIRP)

No Duty Cycle, No Channel Restriction.

**Europe**

Table 9.1 European 433MHz License Conditions

<table>
<thead>
<tr>
<th>Freq. Band</th>
<th>Power</th>
<th>Antenna</th>
<th>Ch. Spacing</th>
<th>Lic. Req</th>
<th>Duty Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>433.050-434.790MHz</td>
<td>10mW</td>
<td>Internal or Type Approved External</td>
<td>N/A (1)</td>
<td>Not Req.</td>
<td>10% (2)</td>
</tr>
</tbody>
</table>

NOTES: (1) No set frequency channelling - can use anywhere in the band segment. (2) Duty Cycle should equate to 10%. Maximum recommended TX on time is 36 seconds, minimum TX off time is 3.6 seconds. For example, 10 transmissions of 36 seconds within one hour. (These limits are advisory with a view to facilitating sharing between systems in the same frequency band).

**Country Specific Requirements (Europe):**

Table 9.2 European, country specific 433MHz license conditions. (Annex 1 for Band E  (Ch.2-[62]))

<table>
<thead>
<tr>
<th>Administration</th>
<th>Remark/Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Denmark</td>
<td>Audio and Voice Signals only if the e.r.p is below 100 microwatts.</td>
</tr>
<tr>
<td>Finland</td>
<td>Audio &amp; voice Signals not allowed.</td>
</tr>
<tr>
<td>France</td>
<td>No duty cycle Limit</td>
</tr>
<tr>
<td>Hungary</td>
<td>Two-way speech not allowed</td>
</tr>
<tr>
<td>Italy</td>
<td>Audio and voice signals not</td>
</tr>
<tr>
<td>Latvia</td>
<td>Voice, audio, video is not allowed</td>
</tr>
<tr>
<td>Luxembourg</td>
<td>Audio and voice signals not allowed.</td>
</tr>
<tr>
<td>Norway</td>
<td>Audio and voice signals are not allowed.</td>
</tr>
<tr>
<td>Sweden</td>
<td>25mW is allowed. No duty cycle limitation.</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>Voice not allowed.</td>
</tr>
</tbody>
</table>
United States of America
Federal Communications Commission (FCC) Code of Federal Government Regulations, Title 47, Part 15 (section 231) (Ch.2-[63])

The following table refers to **Intentional Radiators** (systems) operating on frequencies operating between 216 and 960 MHz. It excludes operation on channels used by broadcast services (e.g. Television)

Table 9.3 FCC Regulations for Intentional Radiators

<table>
<thead>
<tr>
<th>Type of System</th>
<th>Radiated Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Continuous</strong></td>
<td>200 microvolts/metre measured at 3 metres</td>
</tr>
<tr>
<td><strong>Non periodic</strong></td>
<td>2750-12500* microvolts/metre at 3 metres</td>
</tr>
<tr>
<td>Burst radiators</td>
<td>* linear interpolation</td>
</tr>
<tr>
<td>Maximum on time 5 seconds.</td>
<td></td>
</tr>
<tr>
<td>Maximum <strong>periodic</strong> signaling at this level, one second at intervals of one hour</td>
<td></td>
</tr>
<tr>
<td><strong>Periodic Radiators</strong></td>
<td>1500-5000* microvolts/metre at 3 metres</td>
</tr>
<tr>
<td>Max on time one second</td>
<td></td>
</tr>
<tr>
<td>Maximum duty cycle 3.33%</td>
<td>* linear interpolation</td>
</tr>
</tbody>
</table>

**Other:**
Major land masses/countries not identified

Table 9.4 Existence of 433MHz Band in Africa, Asia and Americas (not USA)

<table>
<thead>
<tr>
<th>Country / Continent</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>433.92 MHz is a designated ISM band (part of Region 1 which includes Europe)</td>
</tr>
<tr>
<td>Asia</td>
<td>No ISM band at 433.92MHz (Region 3 as is Australia)</td>
</tr>
<tr>
<td>South America, Canada</td>
<td>No ISM band at 433.92MHz (Region 2 as is the USA)</td>
</tr>
</tbody>
</table>

The above countries, which do not have an ISM band centred at 433.92 MHz, may (or may not) have Short Range Device (SRD) class licenses as in the Australian LIPD case.
9.1.2 Australian Class License Conditions

The following is an extract from the *Radiocommunications (Low Interference Potential Devices) Class Licence* (June 2000)(Ch.2-[61]). It summaries the conditions under which the Class License operates. Similar conditions, although worded differently, exist for LIPD or SRD devices operating in other jurisdictions.

**NOTE**

A radiocommunications device supported under this Class Licence can be expected to be operating in radiofrequency spectrum also used by other radiocommunications devices (that is, it shares the spectrum with them). Devices supported under this Class Licence are typically used for communications over short distances. By placing appropriate limits on parameters such as device type, radiated power levels and frequencies of operation, the interference potential of a low interference potential device (LIPD) may be held to a sufficiently low level that enables sharing the spectrum with other radiocommunications devices on an uncoordinated basis in most circumstances.  
It is recognised that interference arising from the operation of a LIPD is still possible, although under less likely circumstances. As an aid to interference resolution in those circumstances, it is a condition of the operation of a device under this Class Licence that the device not cause interference to other radiocommunications devices; as well, a device will not be afforded protection from interference caused by other radiocommunications services (see paragraph 4 (1) (b) and Note 1 after section 4 of this Class Licence).
9.1.3 Class Licence and ISM Frequencies

Some of the frequencies and transmitter classes covered by the Radiocommunications (Low Interference Potential Devices) Class Licence (June 2000) (as amended) (Ch.2- [61]) are extracted and reproduced below.

Table 9.5 Australian Class License transmitter types, operating frequencies and maximum EIRP

<table>
<thead>
<tr>
<th>Item</th>
<th>Class of transmitter</th>
<th>Permitted operating frequency band (MHz) (lower limit exclusive, upper limit inclusive)</th>
<th>Maximum EIRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>All transmitters</td>
<td>40.66–41</td>
<td>1 W</td>
</tr>
<tr>
<td>14</td>
<td>All transmitters</td>
<td>54–56</td>
<td>2.5 mW</td>
</tr>
<tr>
<td>15</td>
<td>All transmitters</td>
<td>1. 70–70.24375  2. 77.29375–77.49375  3. 150.7875–152.49375  4. 173.29375–174</td>
<td>100 mW</td>
</tr>
<tr>
<td>17</td>
<td>All transmitters</td>
<td>433.05–434.79</td>
<td>25 mW</td>
</tr>
<tr>
<td>18</td>
<td>All transmitters</td>
<td>915–928</td>
<td>3 mW</td>
</tr>
<tr>
<td>19</td>
<td>All transmitters</td>
<td>2400–2463</td>
<td>10 mW</td>
</tr>
<tr>
<td>25</td>
<td>Telecommand or telemetry transmitters</td>
<td>472.0125–472.1125</td>
<td>100 mW</td>
</tr>
<tr>
<td>26</td>
<td>Telecommand or telemetry transmitters</td>
<td>1. 2400–2450  2. 5725–5795  3. 5815–5875</td>
<td>1 W</td>
</tr>
<tr>
<td>27</td>
<td>Telecommand or telemetry transmitters</td>
<td>5795–5815</td>
<td>2 W</td>
</tr>
<tr>
<td>39</td>
<td>Aquatic animal tracking transmitters</td>
<td>48–49</td>
<td>10 mW</td>
</tr>
<tr>
<td>44</td>
<td>Radio Local Area Network transmitters (indoors)</td>
<td>5150–5250  5250–5350</td>
<td>200mW  200mW</td>
</tr>
<tr>
<td>45A</td>
<td>Digital Modulation Transmitters</td>
<td>915–928  2400–2483.5  5725–5850</td>
<td>1W  4W  4W</td>
</tr>
<tr>
<td>45B</td>
<td>Digital Modulation Transmitters</td>
<td>915–928 (min 20 hopping freqs)  2400–2483.5 (min 15 hopping freqs)  5725–5850 (min 75 hopping freqs)</td>
<td>1W  500mW  4W</td>
</tr>
</tbody>
</table>

Frequencies and transmitter types of interest highlighted.
### ISM Frequencies, extracted from (Ch.2:[64])

Table 9.6 ISM Bands designated in the Australian Spectrum Plan

<table>
<thead>
<tr>
<th>Band</th>
<th>Centre Frequency</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>6,765 - 6,795 kHz</td>
<td>6,780 kHz</td>
<td></td>
</tr>
<tr>
<td>13,553 - 13,567 kHz</td>
<td>13,560 kHz</td>
<td></td>
</tr>
<tr>
<td>26,957 - 27,283 kHz</td>
<td>27,120 kHz</td>
<td></td>
</tr>
<tr>
<td>40.66 - 40.70 MHz</td>
<td>40.68 MHz</td>
<td></td>
</tr>
<tr>
<td>433.05 - 434.79 MHz</td>
<td>433.92 MHz (Region 1 only)</td>
<td>Covered by Aust Class License</td>
</tr>
<tr>
<td>902 - 928 MHz</td>
<td>915 MHz (Region 2 only)</td>
<td>Partially covered by Aust Class License</td>
</tr>
<tr>
<td>2,400 - 2,500 MHz</td>
<td>2,450 MHz</td>
<td></td>
</tr>
<tr>
<td>5,725 - 5875 MHz</td>
<td>5,800 MHz</td>
<td></td>
</tr>
<tr>
<td>24 - 24.25 GHz</td>
<td>24.125 GHz</td>
<td></td>
</tr>
<tr>
<td>61 - 61.5 GHz</td>
<td>61.25 GHz</td>
<td></td>
</tr>
<tr>
<td>122 - 123 GHz</td>
<td>122.5 GHz</td>
<td></td>
</tr>
<tr>
<td>244 - 246 GHz</td>
<td>245 GHz</td>
<td></td>
</tr>
</tbody>
</table>

Region 1 = Europe & Africa  
Region 2 = North & South America & Greenland  
Region 3 = Australia, Asia & Pacific Islands
9.2 Appendix for Chapter 4, Athlete Data Processing Toolbox

To enable efficient handling of the datasets, and to perform the desktop processing, a number of processing routines were written using the Matlab package. Data downloaded from the triaxial accelerometer sensor device as text files are converted by Matlab functions into a standard athlete format known as 'athdata'. Athdata is a structure containing the accelerometer data as well as information on the athlete, the sample rate, the sensor serial number, the person collecting the data and the type of test. The following are some, but not all, of the functions written for data processing. All the function names start with 'ADAT' to indicate that they belong to the Athlete Data Toolbox.

Some functions, such as ADATfilter, truncate the dataset. Other functions, such as ADATadd, ADATsubtract etc, operate on the assumption that a truncated dataset is the result of filtering and these functions align the dataset based on this assumption.

Generally the use of the functions is in the form:
result=ADATfunction(input, parameter list,..);

for example:
\texttt{cv=adatCombinedVector(cal,-1,1,2,3)};

Where \texttt{cv} is the variable holding the results from the \texttt{ADATCombinedVector} function, \texttt{cal} is the input data and \texttt{-1,1,2,3} are the parameters.

<table>
<thead>
<tr>
<th><strong>Data Visualisations</strong></th>
<th><strong>Processing</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ADAT3Dtrace</td>
<td>Create a 3D plot of the data and animate by running a bead along from the start to the end of the trace.</td>
</tr>
<tr>
<td>ADATEnergyHistogram</td>
<td>Read an energy file and generate a histogram</td>
</tr>
<tr>
<td>ADATview</td>
<td>View single or multi-channel data. Each channel on its own graph. Graphs in a column.</td>
</tr>
<tr>
<td>ADATview2</td>
<td>Compare graphs from two data sets side by side</td>
</tr>
<tr>
<td>ADAToverlay</td>
<td>Create a graph where one data set overlays another. Will adjust for truncated data sets. Can overlay data sets with different sample rates.</td>
</tr>
<tr>
<td>ADATplot3D</td>
<td>Create a 3D plot of data.</td>
</tr>
<tr>
<td>ADATplotall</td>
<td>Plot all traces in a data set on the one graph.</td>
</tr>
<tr>
<td>ADATplotPolar</td>
<td>Generate Polar Plots of data.</td>
</tr>
<tr>
<td>ADATSetGraphForPrinting</td>
<td>Set parameters of graph for simplified import or printing.</td>
</tr>
<tr>
<td><strong>Data Set Manipulation</strong></td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ADATclip</td>
<td>Display data, zoom if required, clip out a piece of data of interest.</td>
</tr>
<tr>
<td>ADATcombine</td>
<td>Combine two data sets into one.</td>
</tr>
<tr>
<td>ADATInterpolate</td>
<td>Use linear interpolation to generate data with different sample rates</td>
</tr>
<tr>
<td>ADATsingletag</td>
<td>Tag a point in a data set.</td>
</tr>
<tr>
<td>ADATtag</td>
<td>Tag multiple points in a data set.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Data Processing</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADATadd</td>
<td>Add the matching elements from two data sets.</td>
</tr>
<tr>
<td>ADATmultiply</td>
<td>Multiply the matching elements from two data sets.</td>
</tr>
<tr>
<td>ADATsubtract</td>
<td>Subtract one data set from another</td>
</tr>
<tr>
<td>ADATAlanLaiCalibrate</td>
<td>Calibrate a data set using the Alan Lai Method.</td>
</tr>
<tr>
<td>ADATangle</td>
<td>Calculate the trigonometrical angle between two data sets</td>
</tr>
<tr>
<td>ADATcombinedvector</td>
<td>Generate a vector magnitude from multidimensional data.</td>
</tr>
<tr>
<td>ADATderivative</td>
<td>Calculate the derivative of a data set.</td>
</tr>
<tr>
<td>ADATenergy</td>
<td>Use the vector magnitude to generate the Power Temporal Density</td>
</tr>
<tr>
<td>ADATfft</td>
<td>Use a FFT to generate a Power Spectral Density</td>
</tr>
<tr>
<td>ADATfilter</td>
<td>Generate low pass data using a Hamming Window FIR Filter</td>
</tr>
<tr>
<td>ADAThamming</td>
<td>Sub function of ADATfilter to generate the Hamming filter elements</td>
</tr>
<tr>
<td>ADATintegrate</td>
<td>Mathematical integration of the data set.</td>
</tr>
<tr>
<td>ADATLag</td>
<td>Generate a first order lag.</td>
</tr>
<tr>
<td>ADATManualRotate</td>
<td>Display data and rotate data set based on a user selected point.</td>
</tr>
<tr>
<td>ADATpolar</td>
<td>Convert cartesian data to polar data.</td>
</tr>
<tr>
<td>ADATRollingFFT</td>
<td>Perform a second-by-second FFT and plot the significant peaks.</td>
</tr>
<tr>
<td>ADATRotate</td>
<td>Rotate a data set using a given input rotation matrix.</td>
</tr>
<tr>
<td>ADAT_Rxx</td>
<td>Calculate the unbiased autocorrelation coefficients of a data set.</td>
</tr>
<tr>
<td>ADAT_Rxy</td>
<td>Calculate the unbiased crosscorrelation coefficients of two data sets.</td>
</tr>
<tr>
<td>ADATSmooth</td>
<td>Apply a rectangular FIR filter (rolling average).</td>
</tr>
<tr>
<td>ADATStrideDetector</td>
<td>Detect positive going zero-crossings in a data set.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Data Compression</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADATcompand</td>
<td>Apply weighting factors to differences.</td>
</tr>
<tr>
<td>ADATcompandRLE</td>
<td>Perform run-length encoding on a companded data set.</td>
</tr>
<tr>
<td>ADATdifference</td>
<td>Calculate differences between successive samples</td>
</tr>
<tr>
<td>ADATDropOneBit.m</td>
<td>Drop the least significant bit of the current data set</td>
</tr>
<tr>
<td>ADATRegenerateDPCM</td>
<td>Regenerate data from a DPCM compressed data set.</td>
</tr>
<tr>
<td>ADATRLE</td>
<td>Perform Run Length Encoding on a data set.</td>
</tr>
<tr>
<td>ADATRunLength</td>
<td>Calculate a histogram of detected run-lengths of a particular value.</td>
</tr>
<tr>
<td>ADATsilence</td>
<td>Apply a silence compression algorithm to the data set. (see Ch.7)</td>
</tr>
</tbody>
</table>
9.3 Appendix for Chapter 5 (Estimated Energy Expenditure)

9.3.1 10x10 Treadmill Trial Data

Table 9.7 Athlete Anthropometric Measures

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Height (cm)</th>
<th>Mass (kg)</th>
<th>Skinfolds</th>
<th>Troch-Lat. (cm)</th>
<th>Lat. Floor (cm)</th>
<th>Leg Length (cm)</th>
<th>Billocristal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>185.7</td>
<td>81.3</td>
<td>45.6</td>
<td>47.9</td>
<td>50.3</td>
<td>99.4</td>
<td>29.2</td>
</tr>
<tr>
<td>2</td>
<td>184.5</td>
<td>78.6</td>
<td>40.5</td>
<td>44.5</td>
<td>48.2</td>
<td>93.9</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>192.2</td>
<td>90.9</td>
<td>49.1</td>
<td>43.4</td>
<td>49.9</td>
<td>94.5</td>
<td>30.7</td>
</tr>
<tr>
<td>4</td>
<td>181.6</td>
<td>79.9</td>
<td>56.4</td>
<td>46.1</td>
<td>48.7</td>
<td>96</td>
<td>29.8</td>
</tr>
<tr>
<td>5</td>
<td>183.6</td>
<td>80.6</td>
<td>84</td>
<td>45.4</td>
<td>48.5</td>
<td>95.1</td>
<td>30.5</td>
</tr>
<tr>
<td>6</td>
<td>187.2</td>
<td>88.45</td>
<td>40.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>186.5</td>
<td>73.5</td>
<td>37</td>
<td>46.9</td>
<td>50.4</td>
<td>98.5</td>
<td>29.6</td>
</tr>
<tr>
<td>8</td>
<td>193</td>
<td>92.4</td>
<td>63.5</td>
<td>50.7</td>
<td>50.7</td>
<td>102.6</td>
<td>31.3</td>
</tr>
<tr>
<td>9</td>
<td>183</td>
<td>84.6</td>
<td>47</td>
<td>43.3</td>
<td>46.8</td>
<td>91.3</td>
<td>29.3</td>
</tr>
<tr>
<td>10</td>
<td>180.9</td>
<td>81.4</td>
<td>60.9</td>
<td>44.8</td>
<td>46</td>
<td>92</td>
<td>29.7</td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking.

Table 9.8 Step Frequency Results

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Speed (km/h)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>1.47</td>
</tr>
<tr>
<td>2</td>
<td>1.47</td>
</tr>
<tr>
<td>3</td>
<td>1.52</td>
</tr>
<tr>
<td>4</td>
<td>1.45</td>
</tr>
<tr>
<td>5</td>
<td>1.61</td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.56</td>
</tr>
<tr>
<td>8</td>
<td>1.5</td>
</tr>
<tr>
<td>9</td>
<td>1.6</td>
</tr>
<tr>
<td>10</td>
<td>1.39</td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking.
### Table 9.9 Triaxial Accelerometer Counts

<table>
<thead>
<tr>
<th>Speed (km/h)</th>
<th>Athlete</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>13</th>
<th>15</th>
<th>17</th>
<th>19</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>60</td>
<td>182</td>
<td>1734</td>
<td>3096</td>
<td>3571</td>
<td>3736</td>
<td>3877</td>
<td>3648</td>
<td>3694</td>
<td>3709</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>98</td>
<td>239</td>
<td>457</td>
<td>4501</td>
<td>4737</td>
<td>4894</td>
<td>4795</td>
<td>4813</td>
<td>4744</td>
<td>5008</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>139</td>
<td>369</td>
<td>2565</td>
<td>3867</td>
<td>4364</td>
<td>4751</td>
<td>4851</td>
<td>4841</td>
<td>5142</td>
<td>4951</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>83</td>
<td>228</td>
<td>536</td>
<td>2190</td>
<td>3003</td>
<td>3466</td>
<td>3680</td>
<td>3743</td>
<td>3959</td>
<td>4179</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>104</td>
<td>221</td>
<td>374</td>
<td>3883</td>
<td>4798</td>
<td>5128</td>
<td>5240</td>
<td>5319</td>
<td>5254</td>
<td>5354</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>145</td>
<td>341</td>
<td>693</td>
<td>4985</td>
<td>5349</td>
<td>5528</td>
<td>5614</td>
<td>5559</td>
<td>5676</td>
<td>5492</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>127</td>
<td>317</td>
<td>3077</td>
<td>4507</td>
<td>4715</td>
<td>4784</td>
<td>4779</td>
<td>4899</td>
<td>4756</td>
<td>4764</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>133</td>
<td>241</td>
<td>562</td>
<td>4175</td>
<td>4967</td>
<td>5567</td>
<td>5906</td>
<td>6173</td>
<td>6281</td>
<td>6414</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>147</td>
<td>1393</td>
<td>3066</td>
<td>3704</td>
<td>4042</td>
<td>4419</td>
<td>4868</td>
<td>5115</td>
<td>5166</td>
<td>4802</td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking.

### Table 9.10 Uniaxial Accelerometer Counts

<table>
<thead>
<tr>
<th>Speed (km/h)</th>
<th>Athlete</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>13</th>
<th>15</th>
<th>17</th>
<th>19</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>24</td>
<td>116</td>
<td>1566</td>
<td>2871</td>
<td>3294</td>
<td>3328</td>
<td>3384</td>
<td>3202</td>
<td>3218</td>
<td>3091</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>30</td>
<td>68</td>
<td>144</td>
<td>3598</td>
<td>3786</td>
<td>3930</td>
<td>3835</td>
<td>3883</td>
<td>3712</td>
<td>4021</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>44</td>
<td>95</td>
<td>1513</td>
<td>2290</td>
<td>2595</td>
<td>2878</td>
<td>2786</td>
<td>2907</td>
<td>3160</td>
<td>2782</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>26</td>
<td>42</td>
<td>199</td>
<td>1593</td>
<td>2223</td>
<td>2509</td>
<td>2622</td>
<td>2482</td>
<td>2475</td>
<td>2528</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>38</td>
<td>92</td>
<td>104</td>
<td>3293</td>
<td>3945</td>
<td>4058</td>
<td>4080</td>
<td>4084</td>
<td>3998</td>
<td>4055</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>62</td>
<td>147</td>
<td>335</td>
<td>4241</td>
<td>4611</td>
<td>4793</td>
<td>4909</td>
<td>4857</td>
<td>4861</td>
<td>4592</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>49</td>
<td>100</td>
<td>2089</td>
<td>3119</td>
<td>3258</td>
<td>3346</td>
<td>3376</td>
<td>3474</td>
<td>3318</td>
<td>3257</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>47</td>
<td>75</td>
<td>168</td>
<td>3359</td>
<td>3990</td>
<td>4471</td>
<td>4835</td>
<td>5025</td>
<td>4969</td>
<td>4924</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>53</td>
<td>985</td>
<td>2183</td>
<td>2738</td>
<td>2891</td>
<td>3156</td>
<td>3427</td>
<td>3514</td>
<td>3479</td>
<td>3225</td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking.
9.3.2 SF vs $\dot{V}O_2$ based EE Treadmill Trial Data

Table 9.11 Athlete Anthropometric Measures

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Gender</th>
<th>Age (years)</th>
<th>Height (cm)</th>
<th>Mass (kg)</th>
<th>Upper leg (cm)</th>
<th>Lower leg (cm)</th>
<th>Total leg (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>25.9</td>
<td>172.6</td>
<td>60.2</td>
<td>41.1</td>
<td>46.5</td>
<td>87.6</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>25.9</td>
<td>179.6</td>
<td>86.2</td>
<td>43.1</td>
<td>49.3</td>
<td>92.4</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>32.0</td>
<td>168.2</td>
<td>58</td>
<td>40.1</td>
<td>44.6</td>
<td>84.7</td>
</tr>
<tr>
<td>4</td>
<td>M</td>
<td>22.4</td>
<td>176.1</td>
<td>61.6</td>
<td>49.4</td>
<td>44.6</td>
<td>94.0</td>
</tr>
<tr>
<td>5</td>
<td>M</td>
<td>39.6</td>
<td>166.9</td>
<td>60</td>
<td>42.3</td>
<td>42</td>
<td>84.3</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>37.0</td>
<td>180.8</td>
<td>72</td>
<td>45.3</td>
<td>48.8</td>
<td>94.1</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>35.9</td>
<td>185.4</td>
<td>94.2</td>
<td>44.4</td>
<td>48.9</td>
<td>93.3</td>
</tr>
<tr>
<td>8</td>
<td>M</td>
<td>39.1</td>
<td>179.8</td>
<td>81.0</td>
<td>41.5</td>
<td>49.0</td>
<td>90.5</td>
</tr>
<tr>
<td>9</td>
<td>M</td>
<td>26.9</td>
<td>178.4</td>
<td>67.7</td>
<td>45.4</td>
<td>46.7</td>
<td>92.1</td>
</tr>
<tr>
<td>10</td>
<td>M</td>
<td>30.0</td>
<td>187.5</td>
<td>80.6</td>
<td>43.4</td>
<td>51.6</td>
<td>95.0</td>
</tr>
</tbody>
</table>

Table 9.12 Step Frequency Results

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Speed (km/h)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>1.85</td>
</tr>
<tr>
<td>2</td>
<td>2.11</td>
</tr>
<tr>
<td>3</td>
<td>1.62</td>
</tr>
<tr>
<td>4</td>
<td>2.02</td>
</tr>
<tr>
<td>5</td>
<td>1.99</td>
</tr>
<tr>
<td>6</td>
<td>1.87</td>
</tr>
<tr>
<td>7</td>
<td>1.99</td>
</tr>
<tr>
<td>8</td>
<td>1.89</td>
</tr>
<tr>
<td>9</td>
<td>2.05</td>
</tr>
<tr>
<td>10</td>
<td>1.97</td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking
Table 9.13 Triaxial Accelerometer Counts

<table>
<thead>
<tr>
<th>Speed (km/h)</th>
<th>Athlete 1</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athlete</td>
<td></td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>154</td>
<td>556</td>
<td>4625</td>
<td>5490</td>
<td>5960</td>
<td>6090</td>
<td>5980</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>503</td>
<td>3326</td>
<td>4208</td>
<td>4879</td>
<td>5733</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>170</td>
<td>606</td>
<td>4362</td>
<td>5534</td>
<td>6114</td>
<td>6538</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>467</td>
<td>1363</td>
<td>4377</td>
<td>4784</td>
<td>5150</td>
<td>5060</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>867</td>
<td>1887</td>
<td>3471</td>
<td>4440</td>
<td>4870</td>
<td>4821</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>403</td>
<td>1275</td>
<td>3438</td>
<td>4465</td>
<td>5200</td>
<td>5335</td>
<td>5670</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>393</td>
<td>2687</td>
<td>3647</td>
<td>4464</td>
<td>5017</td>
<td>5261</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>390</td>
<td>1161</td>
<td>3865</td>
<td>4780</td>
<td>5315</td>
<td>5355</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>540</td>
<td>3485</td>
<td>4354</td>
<td>5200</td>
<td>5335</td>
<td>5670</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>372</td>
<td>1138</td>
<td>3850</td>
<td>4400</td>
<td>4780</td>
<td>5170</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking

Table 9.14 Volume of Oxygen used (litres)

<table>
<thead>
<tr>
<th>Speed (km/h)</th>
<th>Athlete 1</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athlete</td>
<td></td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>0.42</td>
<td>0.78</td>
<td>1.60</td>
<td>2.18</td>
<td>2.58</td>
<td>2.90</td>
<td>3.02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.46</td>
<td>2.04</td>
<td>2.72</td>
<td>3.20</td>
<td>3.74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.52</td>
<td>0.72</td>
<td>1.42</td>
<td>1.48</td>
<td>2.24</td>
<td>2.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.80</td>
<td>1.36</td>
<td>2.04</td>
<td>2.72</td>
<td>2.86</td>
<td>3.44</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.66</td>
<td>1.50</td>
<td>1.98</td>
<td>2.44</td>
<td>2.68</td>
<td>2.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.14</td>
<td>1.74</td>
<td>2.48</td>
<td>2.62</td>
<td>3.30</td>
<td>4.04</td>
<td>4.26</td>
<td>4.62</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.32</td>
<td>2.22</td>
<td>3.04</td>
<td>3.54</td>
<td>4.06</td>
<td>4.26</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.86</td>
<td>1.74</td>
<td>2.46</td>
<td>3.02</td>
<td>3.38</td>
<td>3.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1.32</td>
<td>1.56</td>
<td>2.20</td>
<td>2.54</td>
<td>3.06</td>
<td>3.22</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.18</td>
<td>1.88</td>
<td>2.78</td>
<td>3.16</td>
<td>3.26</td>
<td>3.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking

Table 9.15 Energy Expenditure (EE) estimated from VO2

<table>
<thead>
<tr>
<th>Speed (km/h)</th>
<th>Athlete 1</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athlete</td>
<td></td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>8.80</td>
<td>16.38</td>
<td>32.86</td>
<td>45.78</td>
<td>54.98</td>
<td>63.45</td>
<td>67.78</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>31.49</td>
<td>43.58</td>
<td>58.11</td>
<td>69.68</td>
<td>82.98</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>10.84</td>
<td>14.71</td>
<td>29.09</td>
<td>30.78</td>
<td>47.97</td>
<td>58.33</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16.97</td>
<td>28.63</td>
<td>43.05</td>
<td>58.11</td>
<td>61.54</td>
<td>75.62</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14.27</td>
<td>31.81</td>
<td>42.81</td>
<td>53.38</td>
<td>56.01</td>
<td>66.89</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>24.24</td>
<td>37.08</td>
<td>52.98</td>
<td>55.56</td>
<td>69.82</td>
<td>86.93</td>
<td>93.86</td>
<td>103.70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>27.79</td>
<td>46.97</td>
<td>65.88</td>
<td>77.81</td>
<td>91.13</td>
<td>96.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>18.99</td>
<td>37.53</td>
<td>53.95</td>
<td>66.38</td>
<td>76.39</td>
<td>85.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>28.13</td>
<td>32.28</td>
<td>45.75</td>
<td>54.13</td>
<td>66.79</td>
<td>72.11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>24.48</td>
<td>40.36</td>
<td>60.39</td>
<td>69.95</td>
<td>73.68</td>
<td>85.88</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Grayed cells indicate athlete is walking
Table 9.16 Trunk Lean Angles (degrees), Angle Correlation with Speed and change in Angle

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Angle</th>
<th>Speed (km/h)</th>
<th>Delta-Y</th>
<th>Delta-Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>A1</td>
<td>Y</td>
<td>22.9</td>
<td>25.8</td>
<td>30.6</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>5.7</td>
<td>5.1</td>
<td>5.2</td>
</tr>
<tr>
<td>A2</td>
<td>Y</td>
<td>37.9</td>
<td>40.2</td>
<td>41.7</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>4.2</td>
<td>5.7</td>
<td>5.9</td>
</tr>
<tr>
<td>A3</td>
<td>Y</td>
<td>47.3</td>
<td>46.8</td>
<td>47.3</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>6.9</td>
<td>6.9</td>
<td>6.7</td>
</tr>
<tr>
<td>A4</td>
<td>Y</td>
<td>31.0</td>
<td>32.1</td>
<td>34.5</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>6.4</td>
<td>6.6</td>
<td>7.9</td>
</tr>
<tr>
<td>A5</td>
<td>Y</td>
<td>33.4</td>
<td>34.7</td>
<td>36.3</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>3.4</td>
<td>3.6</td>
<td>3.9</td>
</tr>
<tr>
<td>A7</td>
<td>Y</td>
<td>29.6</td>
<td>29.6</td>
<td>30.7</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>4.4</td>
<td>4.0</td>
<td>3.4</td>
</tr>
<tr>
<td>A8</td>
<td>Y</td>
<td>41.0</td>
<td>42.2</td>
<td>43.3</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>3.7</td>
<td>3.3</td>
<td>2.7</td>
</tr>
<tr>
<td>A9</td>
<td>Y</td>
<td>36.5</td>
<td>36.7</td>
<td>37.7</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>2.9</td>
<td>2.3</td>
<td>3.0</td>
</tr>
<tr>
<td>A10</td>
<td>Y</td>
<td>35.1</td>
<td>35.6</td>
<td>34.7</td>
</tr>
<tr>
<td></td>
<td>Z</td>
<td>1.6</td>
<td>1.9</td>
<td>2.6</td>
</tr>
</tbody>
</table>
Table 9.17 Non-rotated data: Peak to Peak (p-p) acceleration for averaged signal for each athlete for each running speed and axis, with Correlation Coefficients for p-p acceleration against running speed

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Axis</th>
<th>Running Speed (kmh⁻¹)</th>
<th>Correlation Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>A1</td>
<td>V</td>
<td>24.89</td>
<td>28.26</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>5.66</td>
<td>6.86</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>7.13</td>
<td>8.79</td>
</tr>
<tr>
<td>A2</td>
<td>V</td>
<td>27.40</td>
<td>29.35</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>6.53</td>
<td>8.84</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>13.60</td>
<td>15.58</td>
</tr>
<tr>
<td>A3</td>
<td>V</td>
<td>23.22</td>
<td>23.69</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>14.07</td>
<td>17.83</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>17.01</td>
<td>18.49</td>
</tr>
<tr>
<td>A4</td>
<td>V</td>
<td>20.65</td>
<td>22.59</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>10.95</td>
<td>12.46</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>12.23</td>
<td>12.47</td>
</tr>
<tr>
<td>A5</td>
<td>V</td>
<td>27.57</td>
<td>30.92</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>8.67</td>
<td>12.87</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>11.75</td>
<td>13.71</td>
</tr>
<tr>
<td>A7</td>
<td>V</td>
<td>32.13</td>
<td>32.45</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>8.08</td>
<td>8.83</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>12.17</td>
<td>12.66</td>
</tr>
<tr>
<td>A8</td>
<td>V</td>
<td>28.55</td>
<td>27.48</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>8.68</td>
<td>10.40</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>16.76</td>
<td>16.36</td>
</tr>
<tr>
<td>A9</td>
<td>V</td>
<td>32.75</td>
<td>33.00</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>8.87</td>
<td>7.93</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>13.04</td>
<td>14.00</td>
</tr>
<tr>
<td>A10</td>
<td>V</td>
<td>25.03</td>
<td>27.76</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>10.63</td>
<td>12.74</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>15.55</td>
<td>16.86</td>
</tr>
</tbody>
</table>
Table 9.18 Rotated data: Peak to Peak (p-p) acceleration for averaged signal for each athlete for each running speed and axis, with Correlation Coefficients for p-p acceleration against running speed.

<table>
<thead>
<tr>
<th>Athlete</th>
<th>Axis</th>
<th>Running Speed (kmh⁻¹)</th>
<th>Correlation Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>A1</td>
<td>V</td>
<td>24.22</td>
<td>27.67</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>3.90</td>
<td>5.33</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>10.62</td>
<td>11.41</td>
</tr>
<tr>
<td>A2</td>
<td>V</td>
<td>28.06</td>
<td>30.68</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>6.42</td>
<td>7.25</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>12.89</td>
<td>13.66</td>
</tr>
<tr>
<td>A3</td>
<td>V</td>
<td>27.89</td>
<td>28.82</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>13.41</td>
<td>16.83</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>9.05</td>
<td>9.04</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>9.27</td>
<td>12.19</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>6.21</td>
<td>7.96</td>
</tr>
<tr>
<td>A5</td>
<td>V</td>
<td>27.30</td>
<td>32.20</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>8.47</td>
<td>12.90</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>10.50</td>
<td>11.46</td>
</tr>
<tr>
<td>A7</td>
<td>V</td>
<td>32.43</td>
<td>32.88</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>7.99</td>
<td>8.80</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>11.49</td>
<td>11.88</td>
</tr>
<tr>
<td>A8</td>
<td>V</td>
<td>29.69</td>
<td>29.60</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>9.38</td>
<td>11.06</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>14.83</td>
<td>15.46</td>
</tr>
<tr>
<td>A9</td>
<td>V</td>
<td>32.58</td>
<td>33.46</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>7.91</td>
<td>7.86</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>13.52</td>
<td>13.57</td>
</tr>
<tr>
<td>A10</td>
<td>V</td>
<td>28.49</td>
<td>30.79</td>
</tr>
<tr>
<td></td>
<td>ML</td>
<td>10.68</td>
<td>12.75</td>
</tr>
<tr>
<td></td>
<td>AP</td>
<td>9.46</td>
<td>10.26</td>
</tr>
</tbody>
</table>
9.5 Appendix for Chapter 7 (Data Compression)

Companding Code

```c
int getHuffCode(int dpcm)
{
    /* this subroutine performs two functions.
       (1) companding. It takes a dpcm value, compands it and
           returns a companded value (compdiff).
       (2) huffman code finding. It returns a
           huffman code, count of huffman bits and a bit mask.

       input parameters:
       dpcm: the PCM difference between two samples.

       output values (GLOBAL VARIABLES)
       huffmask: a bit mask pointing to the first huffman bit
                in the huffman code byte
       huffcount: count of bits in a huffman code
       huffcode: a byte containing a huffman code
                (left filled with ones)

       output values (RETURN VALUE)
       compdiff: the companded difference (compander output)

       note: the static huffman codes are in a 16 value array

       dpcm = differential pulse code modulation */

    int sdpcm=1;   /*sign of the input difference */
    int adpcm=0;   /*absolute value of the difference*/
    int results=0; /*initialise*/
    int compdiff=dpcm; /*initialise*/

    idx_huffcodes=8; /* set to the 0 value of the huffcodes array*/

    /*handle differences of 0, -1 and +1 */
    if (dpcm == 0) {
        huffmask=1;huffcount=1;huffcode=1;
        /*eg huffman code is 1 (one bit in bit 0)*/
        /*idx_huffcode is already set above*/
        return(compdiff);
    }else if(dpcm == -1) {
        huffmask=2;huffcount=2;huffcode=0;
        /*eg huffman code is 011 (three bits in bits b2,b1,b0)*/
        idx_huffcodes--;
        return(compdiff);
    }else if(dpcm == 1) {
        huffmask=4;huffcount=3;huffcode=3;
        /*eg huffman code is 010 (three bits in bits b2,b1,b0)*/
        idx_huffcodes++;
        return(compdiff);
}
```
adpcm=abs(dpcm);       /* get magnitude of difference */
if (dpcm<0){          /* get sign of difference*/
    sdpcm=-1;
}

if (adpcm<4){
    idx_huffcodes=idx_huffcodes+(sdpcm*2);
    results=2;
} else if (adpcm<8){
    idx_huffcodes=idx_huffcodes+(sdpcm*3);
    results=4;
} else if (adpcm<16){
    idx_huffcodes=idx_huffcodes+(sdpcm*4);
    results=8;
} else if (adpcm<32){
    idx_huffcodes=idx_huffcodes+(sdpcm*5);
    results=16;
} else if (adpcm<64){
    idx_huffcodes=idx_huffcodes+(sdpcm*6);
    results=32;
} else if (adpcm<256){
    idx_huffcodes=idx_huffcodes+(sdpcm*7);
    results=64;
} else {
    /*values greater than 255 are treated as esc code! */
    huffcode=huffcodes[(sizeof huffcodes)-1];
    results=256; sdpcm=1;
}

huffcode=huffcodes[idx_huffcodes];
compdiff = results * sdpcm;

huffmask=128;
huffcount=8;
/* this code finds the first bit of the huffman code
eg the first '0' bit */
while(huffmask&huffcode){
    huffcount--;
    huffmask=huffmask>>1;
}

return(compdiff);
/*
one possible combination of variable length Huffman codes
This table has 18 codes including the escape (Maximum 8 bits each)
Note all codes in the table do not need to be used.
The initialisation of the pointer should point to the 0x01 value
(middle of the table)
The sign of the difference (sdpcm) will determine if the
pointer moves up or down through the table.
*/

const unsigned char huffcodes[]={
    /* value   byte-huffcode actual variable length huffcode*/
    0x46,     /* 01000110     01000110 */
    0x47,     /* 01000111     01000111 */
    0x50,     /* 01010000     01010000 */
    0x51,     /* 01010001     01010001 */
    0x52,     /* 01010010     01010010 */
    0xA2,     /* 10100010     0100010 */
    0xEB,     /* 11101011     01011 */
    0xFC,     /* 11111100     00 */
    0x01,     /* 00000001     1 */
    0xFB,     /* 11111011     011 */
    0xE9,     /* 11101001     01001 */
    0xA0,     /* 10100000     0100000 */
    0xA1,     /* 10100001     0100001 */
    0x53,     /* 01010011     01010011 */
    0x54,     /* 01010100     01010100 */
    0x55,     /* 01010101     01010101 */
    0x56,     /* 01010110     01010110 */
    0x57     /* 01010111     01010111 ESC */
};

void calcDPCM(int oldIndex, int channel){
    /* (1) calculate the current difference for the current channel
        (2) add in any 'left over' from previous companding
        (3) compand (and setup huffman code)
        (4) update the 'left over'
        
        totaldiff is global (holds the carry over between samples)
        diff is local
        silence array index s_head is global
        getHuffCode returns the compander output
        (as well as sets a number of Huffman coding global variables)
        */
    int diff;

    diff=silence[s_head][channel] - silence[oldIndex][channel];
    totaldiff[channel] = totaldiff[channel] + diff;
    totaldiff[channel] = totaldiff[channel] -
                         getHuffCode(totaldiff[channel]);
}

9-17
**Silence Compression**

/* global silence variables */
unsigned char silence [SILENCE_DEPTH][AD_CHANNELS];
int s_size=SILENCE_DEPTH; /* size of the buffer */
int s_head; /* buffer head */
int s_tail; /* buffer tail */
BOOL s_flag_1[AD_CHANNELS]; /* one silence match */
BOOL s_flag_2[AD_CHANNELS]; /* two successive silence matches */

void silenceEncoding (int s_next){
    /* assumptions: the data is currently being loaded into an array
     silence) at the 'tail' by an interrupt service routine. The
     current data is being pointed to by 'head'. Data prior to this
     has been silence processed.

    parameters:
    s_head, s_tail (global ints) current head & tail subscripts.
    s_next index for head plus one
    (precalculated, saves two recalculations every sample).

    processing:
    Data is loaded into the silence array by an ISR. Silence is
    tested for by comparing the previous three samples and the two
    post samples if these are identical the current sample is made
    the same as these. Numerous short cuts are available to reduce
    processing. Because the silence array is treated as circular
    there are some nasty calculations happening with subscripts
    during silence compression. To minimise these the silence
    compression aborts at the first missed comparison.

    processing current iteration
    test flag, calculate index (addition plus modulus),
    retrieve values, update flag

    processing next iteration
    test two flags, calculate index (addition plus modulus),
    retrieve values, update flag

    processing next next iteration
    perform two index calculations (addition plus modulus) and
    three comparisons, update value, update flags.(will always
    do
    two complex indexes, may do three if the second comparison
    fails) Seems reasonably low powered compared to maintaining
    a
    separate shuffle array and performing 6 data copies per
    sample (per channel) plus the silence testing.
*/

9-18
int channel;
/* do all channels */
for (channel=0;channel<AD_CHANNELS;++channel){
    /* a (compare head to head+1) fail will reset both flags */
    if(!s_flag_1[channel]){  
        if (silence[s_head][channel]==silence[s_next][channel]){  
            /* one successful match */  
            s_flag_1[channel] = TRUE;
        }
    } else if(!s_flag_2[channel]){  
        if (silence[s_head][channel]==silence[s_next][channel]){  
            /* two successful matches */  
            s_flag_2[channel] = TRUE;
        }else{
            /* failed to match therefore no matches */  
            s_flag_1[channel] = FALSE;
        }
    } else {
        /* already two successful matches (three identical values)
        now need to compare to samples two & three away */
        if (s_flag_2[channel]==
            silence[(s_head+2)%s_size][channel]) &&
            (silence[s_head][channel]==
            silence[(s_head+3)%s_size][channel])){
            /* silence condition met, update*/
            silence[s_next][channel]=silence[s_head][channel];
            /* reset flags to simplify next two processing steps
            we don't require this complex step since we already
            know the two successive values are already 'silent'.
            The next value that may require updating is the
            current head+4 so we can postpone this complex step
            until necessary.*/
            s_flag_2[channel]=FALSE;
            s_flag_1[channel]=FALSE;
        }else if (s_flag_1[channel] !=
            silence[s_next][channel]){  
            /* reset flags since no current forward matches */
            s_flag_2[channel]=FALSE;
            s_flag_1[channel]=FALSE;
        } /* end if (silent) */
    }/* end if (0th, 1st, 2nd silence) */
}/* end for (all channels) */
Huffman Encoding

void encodeOutput(int channel){
    /* input parameters */
    outcount   current bit position for the output bit
    huffcount  current bit position for the huffman code
    outmask    a one bit mask initialised to 128 (B7=1) and
                continuously right shifted.
    huffmask   a one bit mask initialised to 128 (B7=1) and
                continuously right shifted.
    outcode    the output data byte (originally initialised to 0)
    huffcode   the huffman code getting added to the output data

    input values
    outcount   indicates last bit updated in output data byte.
    huffcount  indicates number of bits in huffman code.
    huffmask   set to align with first bit of huffman code.
    */

    /* while both the output and the input still have bits to process,
       test if there is a one to add to the output, if so add it. 
       eg the huffmask logically ANDed with the huffcode tells us if the 
       huffcode is one or zero in the position being tested. If it is one 
       we add a 'one' to the appropriate position in the output code. 
       There is no necessity to add a zero since the output byte is 
       initialised to zero 
    */

    while (outcount[channel] >0 && huffcount >0){
        if(huffmask & huffcode){
            /* only update the output if input is '1' */
            outcode[channel]=outcode[channel]|outmask[channel];
        }
        outcount[channel]--;huffcount--;
        huffmask=huffmask>>1;
        outmask[channel]=outmask[channel]>>1;
    }

    /* after exiting while loop - possible conditions 
       (1)output byte is full, more huffman bits require encoding 
           (outcount == 0, huffcount > 0) 

    9-20
write byte, clear byte, reset mask, reset count, continue

(2) output byte is full, no more huffman bits to encode
    (outcount == 0, huffcount == 0)
    write byte, clear byte, reset mask, reset count, return

(3) output byte still has space, no more huffman bits to encode
    (outcount > 0, huffcount == 0)
    return. (eg do nothing)

*/

if (outcount[channel] == 0){  /* covers cases (1) & (2) above */
    writeByte(outcode[channel],channel);
    outcode[channel]=0;
    outmask[channel]=128;
    outcount[channel]=8;
    if (huffcount>0){    /* specifically covers case (1) above. */
        encodeOutput(channel);  /* only one level of recursion */
    }       /* can occur since all huffcodes are 8bits or less */
}   /* 9-21 */
10 Glossary and Acronyms

10.1 Terms associated with Wireless Networking (Ch.2)

Access Point  WiFi base station usually connected to wired infrastructure.

ACK/NAK  Acknowledge or Negative acknowledge of data receipt.

ACL  Bluetooth Asynchronous Connection-Less link.

ARQ  Automatic Repeat reQuest protocol for obtaining lost data.

Bluetooth  Wireless cable-replacement technology for PANs

bps  data rate in bits per second, Mega-bps (Mbps) and kilo-bps (kbps)

CDMA  Code Division Multiple Access, spread spectrum CMTS technology (competing technology to GSM).

CMTS  Cellular Mobile Telephone Service

CSMA/CA  Carrier Sense Multiple Access - Collision Avoidance

CTS/RTS  Clear To Send - Ready To Send, a handshaking protocol.

DSSS  Direct Sequence Spread Spectrum. Each bit of data is transmitted on a number of different frequencies. If some frequencies are subject to interference the receiver may still correctly extract the data bit.

EIRP  Equivalent (or Effective) Isotropic Radiated Power. EIRP(dBm) = (Power of Transmitter (dBm)) + (Losses in transmission line (dB)) + (Antenna Gain(dBi))

FCS  Frame Check Sequence, a value generated from the transmitted data and send with the data. The receiver can use the same algorithm on received data to check if the data was received correctly.

FEC  Forward Error Correction. Encoding to allow the detection of data errors or the correct decoding of data in the presence of errors.

FHSS  Frequency Hopping Spread Spectrum. The RF spectrum is divided into specific frequency division channels and the logical data channel hops between frequency channels on a known sequence.

GSM  Global System for Mobile Communications. CMTS based on a combination of FDM (Frequency Division Multiplexing) and TDM.

IEEE 802.11  Institute of Electronic and Electrical Engineers, Wireless Network Protocol specifications (802 is the networking base)

LIPD  Low-Interference-Potential Device
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSI</td>
<td>Large Scale Integration - semiconductor device containing a mix of complex devices integrated onto the same chip.</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control - protocols handling access to the transmission media such as a wired or wireless link</td>
</tr>
<tr>
<td>MCU</td>
<td>MicroController Unit, microprocessor with specialised subsystems.</td>
</tr>
<tr>
<td>MCXO</td>
<td>Microprocessor controlled crystal oscillator</td>
</tr>
<tr>
<td>MPU</td>
<td>Micro Processor Unit</td>
</tr>
<tr>
<td>PAN</td>
<td>Personal Area Network, networking of personal devices such as laptop computer, PDA, mobile telephone, digital camera, headset.</td>
</tr>
<tr>
<td>RBS</td>
<td>Reference Broadcast Scheme</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency IDentifier</td>
</tr>
<tr>
<td>RSSI</td>
<td>Received Signal Strength Indicator, measure of RF signal strength.</td>
</tr>
<tr>
<td>SCO</td>
<td>Bluetooth Synchronous Connection-Oriented link.</td>
</tr>
<tr>
<td>SN</td>
<td>Sensor Network</td>
</tr>
<tr>
<td>SS</td>
<td>Service Set, group of related WiFi stations</td>
</tr>
<tr>
<td>SSID</td>
<td>Service Set IDentifier, the name of the Service Set (SS)</td>
</tr>
<tr>
<td>Station</td>
<td>WiFi node</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol (associate with Internet Protocol, IP), a connection oriented ARQ protocol.</td>
</tr>
<tr>
<td>TDM</td>
<td>Time Division Multiplexing, combining multiple digital channels onto a single media by allocating each channel a time slice in which to transmit data.</td>
</tr>
<tr>
<td>TDMA</td>
<td>Time Division Multiple Access</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver Transmitter</td>
</tr>
<tr>
<td>VCXO</td>
<td>Voltage Controlled crystal Oscillator</td>
</tr>
<tr>
<td>WiFi</td>
<td>Wireless Fidelity - WLAN technology (a pun based on HiFi), designed for wireless networking of computers.</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WPAN</td>
<td>Wireless Personal Area Network</td>
</tr>
<tr>
<td>WSN</td>
<td>Wireless Sensor Network</td>
</tr>
</tbody>
</table>
## 10.2 Terms associated with Operating Systems (Ch.3)

<table>
<thead>
<tr>
<th>Term</th>
<th>Basic Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atomic</td>
<td>Usually refers to a microprocessor instruction, that may involve many phases but is executed in as a single, indivisible, uninterruptible step.</td>
</tr>
<tr>
<td>bps</td>
<td>Bits per second, measure of communications speed.</td>
</tr>
<tr>
<td>Buffer</td>
<td>The section of memory used to implement a pipe or queue. A pipe would usually be implemented as a circular buffer.</td>
</tr>
<tr>
<td>Clock cycles</td>
<td>Term used to count the internal MCU clock cycles necessary to execute a particular MCU instruction.</td>
</tr>
<tr>
<td>Clock-tick</td>
<td>The fundamental component of the real-time clock system (as opposed to the MCU internal clocking). The clock-tick is generated by a timer and increments the system clock.</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Two processes attempt to access and/or modify the same resource. If allowed, invalid processing will result.</td>
</tr>
<tr>
<td>Context Switching</td>
<td>Contexts refer to the information relating to a particular process and include the process ID, the process memory, the registers in use etc.</td>
</tr>
<tr>
<td>Deadlock</td>
<td>Where processes are blocked from operating because, in the simplest form, a process holding one resource is waiting on another process to release a resource, but that process is waiting on the resource held by the first process.</td>
</tr>
<tr>
<td>FLASH</td>
<td>A form of very low cost electrically erasable, programmable memory</td>
</tr>
<tr>
<td>Granularity</td>
<td>A measure of the size of program blocks, particularly the non-interruptible blocks. An operating system may be described by its granularity, eg. a micro-kernel or a monolithic kernel.</td>
</tr>
<tr>
<td>Heap</td>
<td>An area of memory set aside for ad-hoc memory allocation at the programmer's request. In small systems it may be preferable to not allocate heap space but to use static memory declarations and allocate all remaining memory as stack space.</td>
</tr>
<tr>
<td><strong>ISR</strong></td>
<td>Interrupt service routine, the program designed to respond to MCU interrupts.</td>
</tr>
<tr>
<td><strong>JSR</strong></td>
<td>Jump to Sub-Routine</td>
</tr>
<tr>
<td><strong>kbps</strong></td>
<td>kilo-bps or 1000 bps</td>
</tr>
<tr>
<td><strong>Kernel</strong></td>
<td>The component of the operating system that manages the scheduling, memory management, context switching etc.</td>
</tr>
<tr>
<td><strong>Kernel space</strong></td>
<td>See user space</td>
</tr>
<tr>
<td><strong>Laxity</strong></td>
<td>The time difference between deadline time and the time it takes to do the required processing. Eg. If a process must be completed within 50ms of the trigger event and if the required processing takes 30ms then the laxity is 20ms.</td>
</tr>
<tr>
<td><strong>Memory management</strong></td>
<td>Allocating memory to processes so that the memory owned by one process is protected from access by other processes.</td>
</tr>
<tr>
<td><strong>MULTICS</strong></td>
<td>Multiplexed Information and Computing Service, joint development of MIT, General Electric and Bell Labs. Forerunner of UNIX.</td>
</tr>
<tr>
<td><strong>Near-atomic</strong></td>
<td>In the context of this OS, very short functions that consume very few processing cycles eg, the function may read or write a register or input-output port or even perform two or three such actions.</td>
</tr>
<tr>
<td><strong>Pipe</strong></td>
<td>Inter Process Communications mechanism. One process can write data to a pipe, where the data is buffered, then another process can read the data from the pipe. Also see buffer and queue.</td>
</tr>
<tr>
<td><strong>Queue</strong></td>
<td>A queue can refer to a process-queue, where process waiting to be executed are queued in the order they are to be executed. Can also refer to a pipe where data is queued by one process for use by another.</td>
</tr>
<tr>
<td><strong>RTI</strong></td>
<td>Return from Interrupt</td>
</tr>
<tr>
<td><strong>Semaphore</strong></td>
<td>Mechanism for restricting or allowing access to a resource to enforce mutual exclusion, more often implemented as a mechanism for counting and registering processes that are waiting on a resource</td>
</tr>
<tr>
<td><strong>Stack</strong></td>
<td>When interrupts occur, MCU registers are saved to locations in memory referred to as the stack. The stack is managed by a stack-pointer. In Stack-Based Architecture, temporary variables are created, stored and passed 'on the stack'.</td>
</tr>
</tbody>
</table>
UNIX | A computer operating system developed at AT&T Bell Labs in the 1960s-1970s. Many UNIX-like OSes exist today.
---|---
User space | General purpose OSes differentiate between user space, where preemptable user processes run, and kernel space, where the processes are not preemptable. There are also other differences.
UVEPROM | Ultra-Violet Erasable Programmable Read Only Memory
### 10.3 Terms associated with Signal Processing & Biomechanics

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anterior-Posterior</td>
<td>The axis lying in the normal direction of human ambulation.</td>
</tr>
<tr>
<td>Autocorrelation</td>
<td>The creation of autocorrelation coefficients by multiplying a signal by itself. Each successive coefficient is generated by multiplying the signal by an increasingly delayed version of it self.</td>
</tr>
<tr>
<td>Unbiased Autocorrelation</td>
<td>The autocorrelation coefficients have been normalised using a method that removes the effect of the lengths of the data set and delay.</td>
</tr>
<tr>
<td>Cadence</td>
<td>Measure of rate of ambulatory movement eg step frequency.</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processor, A Large Scale Integrated (LSI) device that is designed specifically to perform digital signal processing.</td>
</tr>
<tr>
<td></td>
<td>Depending on the device, this may include calculating the Power Spectral Density on multiple channels, filtering an incoming data stream and producing multiple output streams containing various filtered outputs.</td>
</tr>
<tr>
<td>EE</td>
<td>Energy Expenditure. Usually referred to as estimated EE since all methods of calculating energy expenditure are only estimates.</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform, A processing algorithm that substantially reduces the processing required to perform a Fourier Transform. FFT is often used in the text as a short hand abbreviation for the Power Spectral Density although the output of the FFT actually requires additional processing to create the Power Spectral Density.</td>
</tr>
<tr>
<td>Finite Impulse</td>
<td>The output of the FIR filter is not dependent on any previous output but only on the finite set of input samples. Therefore an input impulse can only affect a finite portion of the output.</td>
</tr>
<tr>
<td>Gait</td>
<td>The manner in with a person (or animal) walks or runs.</td>
</tr>
<tr>
<td>Gait Cycle</td>
<td>One complete cycle of the Gait, for a biped this would be one complete left-step - right-step combination (two successive steps).</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hamming Windowed Filter</td>
<td>There are a number of different ways in which a filter can be constructed. A Hamming Windowed Filter is one where the symmetrical filter elements ($\omega_m$) are described by the equation: $$\omega_m = 0.54 + 0.46 \cos\left(\frac{m}{M}\right) \quad for \ m \leq M, \ 0 \ elsewhere.$$</td>
</tr>
<tr>
<td>Infinite Impulse (IIR)</td>
<td>The output of an IIR system contains a component based on the previous output therefore an impulse in the input continues to have an effect on the output for some time.</td>
</tr>
<tr>
<td>Mediolateral</td>
<td>The horizontal axis perpendicular to both the Anterior-Posterior and dorsal-ventral (Vertical) axes ie. the axis that is horizontal and runs sideways across the body.</td>
</tr>
<tr>
<td>Power Spectral Density</td>
<td>Refers to the graph representing the power in a signal at different frequencies.</td>
</tr>
<tr>
<td>Power Temporal Density</td>
<td>Refers to the graph representing the power in a signal at different points in time.</td>
</tr>
<tr>
<td>Step Rate / Frequency</td>
<td>The frequency at which the athlete takes individual steps.</td>
</tr>
<tr>
<td>Stride Rate</td>
<td>The frequency at which the athlete completes one gait cycle or stride (comprising both a left and a right step). Stride Rate is half the step rate.</td>
</tr>
<tr>
<td>'Vertical'</td>
<td>The axis that aligns with gravity when the athlete is standing.</td>
</tr>
</tbody>
</table>